WANG

VS

System Utilities Reference
Release 7 Series

VS
System Utilities Reference
Release 7 Series

2nd Edition — June 1986

715-0421A

WANG

WANG LABORATORIES, INC., ONE INDUSTRIAL AVENUE, LOWELL, MA 01851 • TEL. (617) 459-5000, TWX 710-343-6769, Telex 94-7421

Disclaimer of Warranties and Limitation of Liabilities

The staff of Wang Laboratories, Inc., has taken due care in preparing this manual. However, nothing contained herein modifies or alters in any way the standard terms and conditions of the Wang purchase, lease, or license agreement by which the product was acquired, nor increases in any way Wang's liability to the customer. In no event shall Wang or its subsidiaries be liable for incidental or consequential damages in connection with or arising from the use of the product, the accompanying manual, or any related materials.

Software Notice

All Wang Program Products (software) are licensed to customers in accordance with the terms and conditions of the Wang Standard Software License. No title or ownership of Wang software is transferred, and any use of the software beyond the terms of the aforesaid license, without the written authorization of Wang, is prohibited.

Warning

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device, pursuant to Subpart J of Part 15 of FCC rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

PREFACE

Wang Laboratories, Inc., provides a variety of programs that perform routine data processing operations with the VS Operating System. These programs are referred to as "utilities" and extend the VS Operating System support of the programming environment. The VS System utilities are documented in a set of five manuals, each describing a different phase of the data processing environment. You can use any one of the manuals separately or in combination with other manuals.

VS System Utilities Reference (715-0421)
This manual describes the utilities that perform the support tasks involved in program and file processing. The support tasks are categorized as follows: copy support, program support, and system support. The VS System utilities can copy, sort, display, and print files, and they can initialize or analyze system storage media. This manual is intended for all levels of programmers. Introductory concepts and brief descriptions of all system utilities are included in Chapter 1. The utility chapters are alphabetized to facilitate access to the information. The utilities in this manual are as follows:

CIP	COPYWP	FASTLINK	IBMCOPY	OISCART	SORT	TAPEINIT
COPY	DISKINIT	FLOPYDUP	IOELOG	PATCH	SORTINT	TRANSL
COPYOIS	DISPLAY	FONTCNTL	IOTRACE	POOLSTAT	TABLEDIT	VERIFY
COPY2200	EZFORMAT	FORMCNTL	LISTVTOC	SHRSTAT	TAPECOPY	

You should be familiar with the VS environment, as described in the *VS System User's Introduction* (715-0417). In addition, topics treated in the following manuals are helpful to the discussions of individual system utilities.

VS Program Development Tools Reference (715-0384)
This manual describes program development tools that are provided with the VS Operating System, which includes the following utilities:

EDITOR LINKER SYMBOLIC DEBUGGER

VS File Management Utilities Reference (800-1308)
This manual describes the utilities that facilitate the data entry and retrieval process. You can construct, modify, and report on data files through the VS File Management utilities. The utilities in this manual are as follows:

CONTROL CONDENSE DATENTRY EZFORMAT INQUIRY REPORT

VS System Administrator's Reference (715-0420)
This manual describes the utilities that enable a system administrator to create configuration files and to protect system resources against unauthorized use. It also describes the Shared Subroutine Library (SSL) utility. Theutilities in this manual are as follows:

GENEDIT SECURITY SSL

VS System Operator's Guide (715-0418)
This manual describes the BACKUP and VOLCOPY utilities, which enable you to back up or restore files, libraries, or volumes. It also describes other system operation functions. The utilities in this manual are as follows:

BACKUP VOLCOPY

Topics treated in the following manuals may also be helpful to the discussions of individual utilities:

Department of Defense Industrial Security Manual (DoD #5220.22-M, Section 13, Paragraph 116).

Operating Procedures Guide (800-6212)

OIS BASIC Language Reference (800-5650)

Word Processing Reference Manual (700-7611)

VS BASIC Language Reference (800-1202)

VS COBOL 74 Reference (800-1201)

VS DMS/TX Reference (800-1128)

VS Data Management System (DMS) Reference (800-1124)

VS FORTRAN 77 Language Reference (800-1208)

VS Operating System Release 7.10 Software Bulletin (715-0060)

VS Operating System Services Reference (715-0423)

VS PL/I Language Reference (800-1209)

VS Principles of Operation (715-0422)

VS Procedure Language Reference (800-1205)

VS Programmer's Guide to VS/IIS (800-1304)

VS RPG II Language Reference (800-1203)

VS Operating System Release 7.10 Customer Software Release Notice (715-0200)

VS Model 2529V Cartridge Tape Drive Operating Procedures Guide

VS System User's Guide to VS/IIS

CONTENTS

CONTENTS (continued)

CHAPTER 4 THE COPYOIS UTILITY

CHAPTER 5 THE COPY2200 UTILITY

CHAPTER 6 THE COPYWP UTILITY

CHAPTER 9 THE EZFORMAT UTILITY

CHAPTER 10 THE FASTLINK UTILITY

CONTENTS (continued)

FIGURES

FIGURES (continued)

FIGURES (continued)

FIGURES (continued)

TABLES

CHAPTER 1
INTRODUCTION TO THE VS SYSTEM UTILITIES

1.1 THE VS SYSTEM UTILITIES

The VS System utilities enable you to perform many of the routine tasks in the programming cycle. The utilities are categorized into three functional groups: copy support, program support, and system support. Table 1-1 shows in which category each VS System utility is classified. The following sections in this chapter summarize the categories and their particular utilities.

This reference manual is organized alphabetically by utility name, to assist you in locating the specific information about the VS System utilities. All utilities reside in the system library @SYSTEM@ on the system volume. The file name of each utility is the same as the name of that utility. For example, the COPYWP utility is located in the file COPYWP in the @SYSTEM@ library on the system volume.

Table 1-1. Classification of VS System Utilities

Copy Support	Program Support	System Support
COPY COPYOIS COPY2200 COPYWP FLOPYDUP IBMCOPY OISCART TAPECOPY TRANSL	DISPLAY EZFORMAT SORT SORTINT TABLEDIT	CIP [a] DISKINIT FASTLINK FONTCNTL FORMCNTL IOELOG IOTRACE LISTVTOC PATCH POOLSTAT SHRSTAT TAPEINIT VERIFY

[a] Compress-in-Place

1.1.1 The Copy Support Utilities

The copy support utilities transfer files, libraries, or volumes between one device and another. Some of the copy utilities also convert data formats of other computer systems to VS data format and from VS data format to the data formats of other computer systems, so that you can transfer data from one computer system to another. You specify what you want copied, where you want it copied to, and in many cases, how you want the copy to be arranged. The copy support utilities are summarized as follows:

Utility	Description
COPY	The COPY utility creates a copy of an input file, a library, or a volume. Optionally, COPY also modifies the file organization, or, for indexed files, rebuilds the index structure when a file is copied. (Refer to Chapter 3 for more information.)
COPYOIS	The COPYOIS utility transfers Office Information System (OIS) files to and from the VS. This utility allows you to convert OIS BASIC source programs to VS BASIC source programs and OIS data files to VS data file format. (Refer to Chapter 4 for more information.)
COPY2200	The COPY2200 utility provides a convenient method for exchanging information between the VS and the Wang 2200 Series systems. COPY2200 transfers data to and from 2200 diskettes and converts the files to and from the 2200 file formats. (Refer to Chapter 5 for more information.)
COPYWP	The COPYWP utility converts VS word processing documents to VS files and converts VS files to VS word processing documents. This function enables you to transfer information between the VS and Word Processing Systems. In addition, COPYWP performs filing functions for VS word processing documents and libraries. (Refer to Chapter 6 for more information.)
FLOPYDUP	The FLOPYDUP utility duplicates diskettes, creates diskette image files (files containing byte-by-byte copies of entire diskettes), and transfers diskette image files to diskettes. (Refer to Chapter 11 for more information.)
IBMCOPY	The IBMCOPY utility copies and converts IBM-format diskette files to and from the VS file format, with optional translation to and from the ASCII and EBCDIC character sets. IBMCOPY requires you to use a diskette drive that accepts soft-sectored diskettes. (Refer to Chapter 14 for more information.)
OISCART	The OISCART utility enables you to transfer large amounts of word processing (WP) documents from the Wang Office Information System (OIS) to the Wang VS. OISCART can retrieve WP documents that were created from an OIS and placed on 1/4-inch cartridge tape. OISCART fully preserves the structure of the document throughout the transfer. (Refer to Chapter 18 for more information.)
TAPECOPY	The TAPECOPY utility copies tape or disk files to other tape or disk files. TAPECOPY is most appropriate for copying data to a tape volume. (Refer to Chapter 25 for more information.)
TRANSL	The TRANSL utility translates files to and from the ASCII and EBCDIC character sets. TRANSL also translates files according to a translation table that you specify. (Refer to Chapter 27 for more information.)

1.1.2 The Program Support Utilities

The program support utilities facilitate programming tasks, either by generating source code sections or by performing a common programming task. The program support utilities are summarized as follows:

Utility	Description
DISPLAY	The DISPLAY utility displays the contents of any file in block or record format at a workstation and enables you to locate, view, and print the data in that file. (Refer to Chapter 8 for more information.)
EZFORMAT	The EZFORMAT utility generates source code that reproduces a screen format that you design. From your screen design, you can enter data into a program or run other utilities from a menu that you created. (Refer to Chapter 9 for more information.)
SORT	The SORT utility sorts records in data files according to key values. It can also merge two or more sorted files. (Refer to Chapter 22 for more information.)
SORTINT	The SORTINT utility sorts up to 20 files into a single, ordered output file according to standard ASCII or an external collating sequence. SORTINT can also reformat the output record. (Refer to Chapter 23 for more information.)
TABLEDIT	The TABLEDIT (Table Edit) utility enables you to create or modify table files that define a collating sequence (sort order) or case flip table. (Refer to Chapter 24 for more information.)

1.1.3 The System Support Utilities

The system support utilities initialize, analyze, and manipulate VS disks, diskettes, and tapes. The system support utilities are summarized as follows:

Utility	Description
CIP	The CIP (Compress-in-Place) utility consolidates free extents on nonsystem disks without doing a full volume backup and volume restore. (Refer to Chapter 2 for more information.)
DISKINIT	The DISKINIT utility initializes (or reinitializes) a disk or a diskette so that you can use it for storing data on the VS. DISKINIT can also create an identifying volume label and a Volume Table of Contents (VTOC). DISKINIT can perform read and write verification, reformat, relabel, replace bad blocks, and erase data. You can also allocate a dump file and build a page pool through the DISKINIT utility. (Refer to Chapter 7 for more information.)
FASTLINK	The FASTLINK utility allows you to specify the program files that you want the system to keep open indefinitely. If frequently-run application and utility program files are kept open, your system's performance improves because there is a reduction in Volume Table of Contents (VTOC) input/output (I/O) and control block allocations. (Refer to Chapter 10 for more information.)

Utility	Description
FONTCNTL	The FONTCNTL utility allows you to specify and manage print fonts for font-loadable printers. A font is a complete set of type of one size and one face. With FONTCNTL, you can install, delete, make inquiries, and assign numbers to Wang-supplied fonts. (Refer to Chapter 12 for more information.)
FORMCNTL	The FORMCNTL utility creates a form control definition, which is placed in a Forms Definition file, for VS serial forms-loadable printers. The forms control definition consists of information to generate formatted printed output. (Refer to Chapter 13 for more information.)
IOELOG	The IOELOG utility examines the contents of an I/O error log file. You can use this utility to evaluate the history and condition of equipment that is configured on the system. (Refer to Chapter 15 for more information.)
IOTRACE	The IOTRACE utility monitors the I/O trace table and records I/O information for a range of devices, based on criteria that you define. (Refer to Chapter 16 for more information.)
LISTVTOC	The LISTVTOC utility analyzes the contents and accuracy of a volume's table of contents (VTOC). The analysis can include a dump of the VTOC control blocks. (Refer to Chapter 17 for more information.)
PATCH	The PATCH utility enables you to display and modify at your workstation the hexadecimal notation of any file in 16-byte displays. You can also produce a dump of an entire file or a specified portion of a file. (Refer to Chapter 19 for more information.)
POOLSTAT	The POOLSTAT utility monitors the use of up to three page pools on VS15, VS25, VS45, VS65, VS85, VS90, VS100, and VS300 systems. (Refer to Chapter 20 for more information.)
SHRSTAT	SHRSTAT enables you to display statistics that the sharer collects as it processes user requests. The sharer is a dedicated system task that coordinates access to shared files. (Refer to Chapter 21 for more information.)
TAPEINIT	The TAPEINIT utility initializes (or reinitializes) a tape by writing one of three types of labels and an end-of-tape marker, so that you can use the tape to record data. (Refer to Chapter 26 for more information.)
VERIFY	The VERIFY utility tests the primary index, the alternate indexes, and the data chain of indexed files to disclose file structure problems. (Refer to Chapter 28 for more information.)

1.2 RUNNING THE UTILITIES IN THE VS ENVIRONMENT

You can run the VS System utilities interactively from the Command Processor menu (Figure 1-1), interactively from a user-developed menu, or as a background task through a procedure that is created from the VS Procedure language.

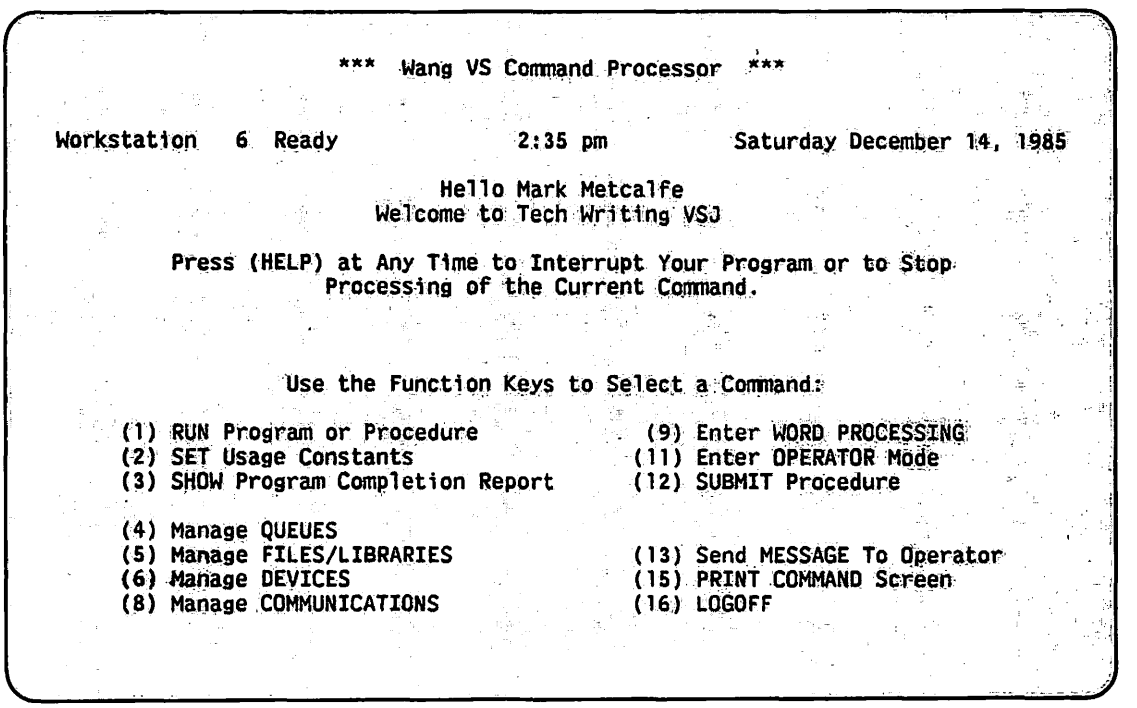


Figure 1-1. Command Processor Menu

1.2.1 The Command Processor Menu

The Command Processor menu is discussed in full in the *VS System User's Introduction* but is reproduced in Figure 1-1 because it is referred to throughout this reference manual. All utilities reside in the system library, @SYSTEM@, and therefore do not require you to specify the Library or Volume fields. If you run a program that is not in @SYSTEM@, specify the library and volume on which it resides before pressing ENTER from that screen. When ENTER is pressed from the Run Program or Procedure screen, the utility or program that you specified begins processing. (If you made an error in specifying the name of a utility or program, or no such program exists, a message will indicate this and allow you to respecify the input.)

1.2.2 Running Utilities Through the Command Processor Menu

To run a utility through the Command Processor menu, perform the following steps:

1. Press Program Function (PF) key 1 (RUN Program or Procedure) from the Command Processor menu and the Run Program or Procedure screen appears (Figure 1-2).

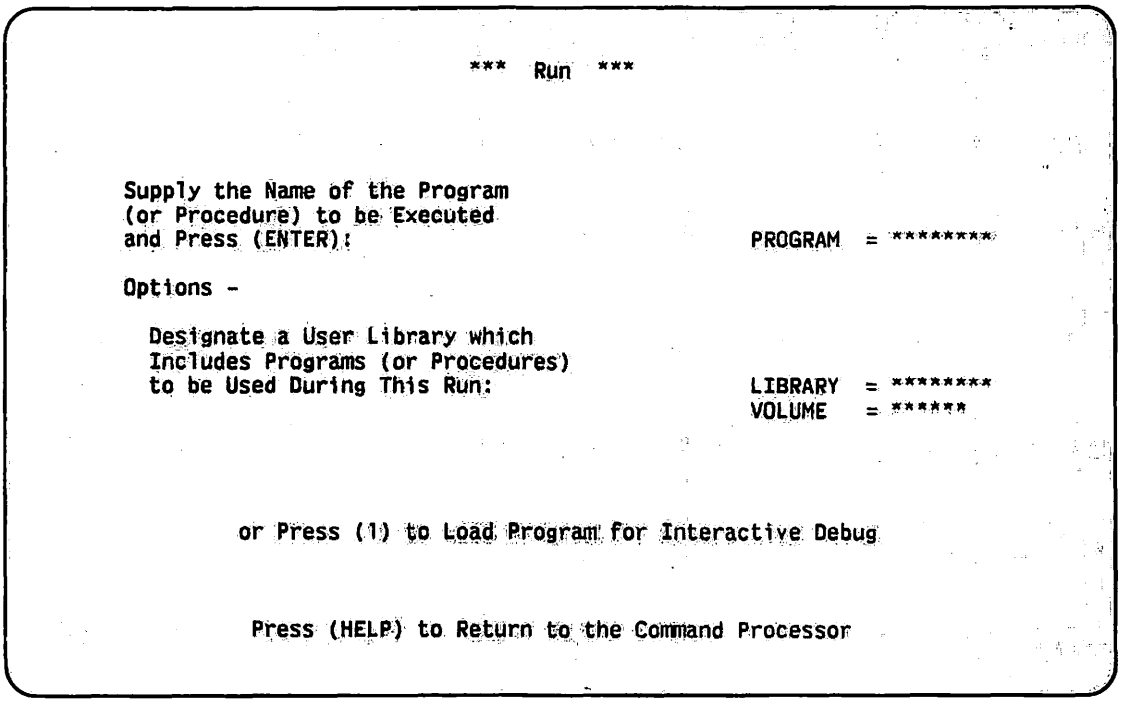


Figure 1-2. Run Program or Procedure Screen

2. Specify the utility name in the Program field and press ENTER.

   You may omit the library and volume names (or leave the default values in the Library and Volume fields). The VS always searches the @SYSTEM@ library on the System Program volume if the specified utility name cannot be located in the library and volume location that is indicated in the Library and Volume fields.

   Type blanks in the Library field if it contains a file with the same name as the utility. If you incorrectly specify a utility, respecify the correct utility name and press ENTER.

3. The "Program (UTILITY-NAME) in Progress" message appears briefly before the utility's main menu appears and user operations begin.

1.2.3 Running Utilities from User-Developed Menus

You can dynamically create a workstation screen display and automatically generate an Assembler, BASIC, COBOL, or RPG II source code program that reproduces the screen display that you created. From these screen displays, you can run any VS utility. User-developed screen displays are created through the EZFORMAT utility (refer to Chapter 9).

1.2.4 Running Utilities Through VS Procedure Language

If you use a utility regularly with the same input or output options and fields (or both input and output options and fields), you may find it helpful to write a procedure to automatically provide the repeated fields. If the utility is run from a procedure, specify only the utility name that follows the Procedure language RUN statement. Most chapters in this reference manual include a sample procedure to help you write a customized procedure. (Some utilities are strictly interactive and therefore cannot be run through a procedure.)

VS System utilities are designed so that workstation interaction can be controlled through the VS Procedure language. Most utility information requests are processed through GETPARMs, which are the VS interface to the Procedure language. A brief description of GETPARMs and the VS Procedure language is in Appendix A. The *VS Procedure Language Reference* provides a comprehensive description of procedure processing requirements.

1.3 THE OPERATOR'S CONSOLE MENU

The Operator's Console menu (Figure 1-3) provides information about your system that aids in analyzing system performance. Some utilities, while run through the Command Processor menu, need input from information obtained through the Operator's Console menu.

For example, to analyze or print the error log through the IOELOG utility, you must first copy the input/output error log file into a separate file from the Operator's Console menu. You must copy the error log because you cannot use the active error log file as input. You can purge the active error log file from the same menu after you have copied it. If you use the purge option, the system then creates a new error log file.

The Operator's Console menu is discussed in full in the *VS System Operator's Guide*, but is reproduced in Figure 1-3 because it is referred to throughout this reference manual.

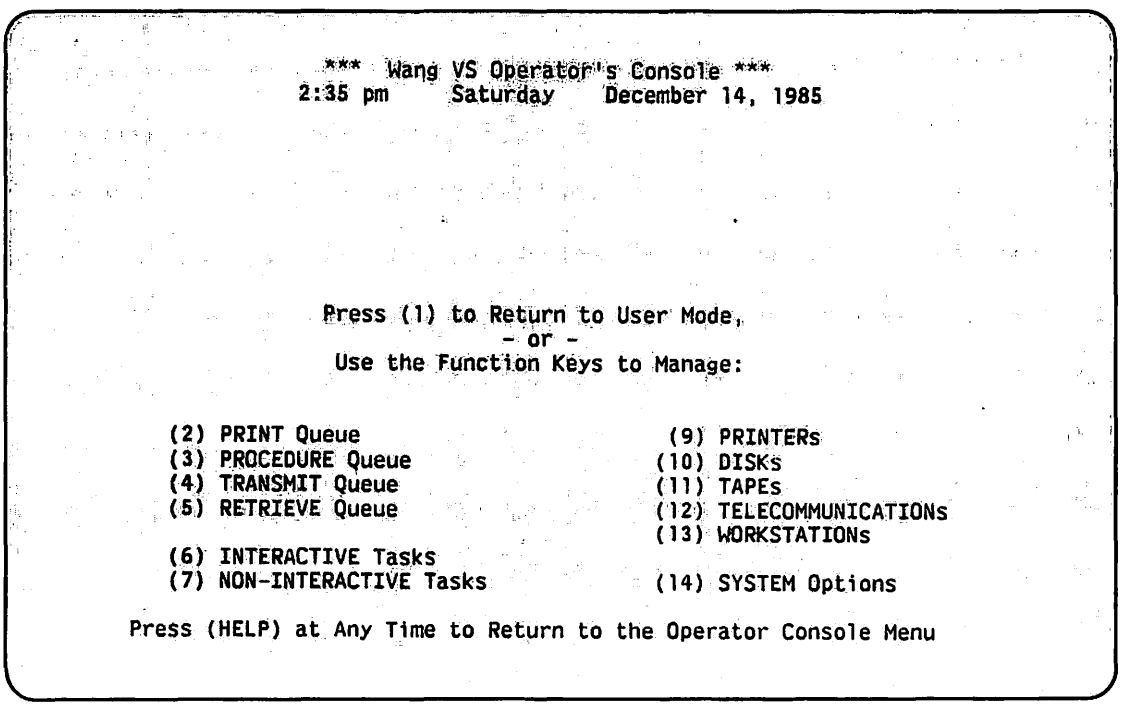


Figure 1-3. Operator's Console Menu

1.4 VOLUME SETS AND MULTIVOLUME FILES

The VS Operating System supports files that span more than one volume. These files are known as multivolume files and they reside on volume sets. A volume set contains one or more volumes that are identified by the name of the volume to which they belong. A volume set identification (VSID) number uniquely differentiates one volume from another in a volume set. Volume sets allow very large files that cannot reside on one disk volume to span up to 255 disk volumes. You can create a volume set through the DISKINIT utility.

For example, if you create a volume set, called Alpha, that contains four volumes, each disk volume is also called Alpha. To differentiate between the four Alpha volumes, you assign a VSID value (1 – 255) to each volume. One member of Alpha (the root) must have VSID 1 assigned to it.

Various utilities in this reference manual have options that are specific to the use of a volume set and multivolume files. For the most part, the use of volume sets and multivolume files are transparent to you; there is little difference between the way volumes and volume sets are treated.

1.4.1 VSID Assignments

One of the volumes in a volume set must be the root volume (VSID 1). The rest of the volumes in a volume set, called secondary volumes, can have any number from 2 to 255 and they need not be in ascending order. In the previous example, your root volume must be VSID 1, but the other three can be VSIDs 3, 15, and 127. Secondary volumes (any volume except the root volume) take no precedence over each other.

CAUTION
You should assign VSID numbers sequentially within a volume set (although it is not necessary), especially when adding volumes to the volume set. If you try to assign an existing VSID number to a new member when that existing member is mounted, an error message appears. However, you can create a new member with a VSID that already exists in the volume set as long as the member with the duplicate VSID is not mounted at the same time that you are creating the new member. This situation, though, can cause unpredictable results. Therefore, you must be extra careful to assign each new member a unique VSID.

1.4.2 The Root Volume

The root volume of a volume set contains the volume set table of contents, also known as the root VTOC. The root VTOC provides indexes that are used by the secondary volumes to find files in a volume set. When you request information on a volume set, the root volume narrows the data location to a particular member of the volume set. If the data is on an unmounted member, the system operator is prompted to mount that member. If the data is on a mounted secondary volume, the data is then accessed through the secondary volume's VTOC. The data is accessed from a volume set in much the same way as it would be accessed from a single volume.

The system uses information in the root VTOC to locate data that is contained on all the volumes of the volume set. Each secondary volume also has its own VTOC, which the system uses to locate specific data on that volume.

1.4.3 Volume Set Impacts on the Utilities

If you use a volume set, the only change that you see occurs when you request data that resides on an unmounted secondary volume. When this occurs, the system pauses and the system operator receives a mount request. Since the root VTOC knows where all data on a volume set is stored, you do not have to mount volumes on a trial basis; the system specifies which volume has the required data to continue your operation. If the root volume is not mounted when you attempt to access a file, a "volume not mounted" error is returned.

1.4.4 Renaming a Multivolume File

When you rename a file on a volume set, the file's entry in the root VTOC is the only entry that is updated. File naming information on secondary VTOCs remain unchanged. Consider the following example:

The root VTOC has information that the file CUSTOMER is on VSID 7 in the address range of 1100 through 5700. If you decide to rename file CUSTOMER to CLIENT, only the root volume VTOC is updated; all references on all volumes that the file CUSTOMER spans are now linked to the file name CLIENT. CUSTOMER no longer exists (although the contents of CUSTOMER exists under the name CLIENT.) The secondary VTOC information remains the same. When information from the file CLIENT is required, the root volume indicates that the file CLIENT is on VSID 7, Addresses 1100 through 5700. The root volume checks for a mounted VSID 7. If mounted, the VTOC on VSID 7 verifies that the file segment exists in Address Range 1100 through 5700 at which point the information is accessed.

CHAPTER 2
THE COMPRESS-IN-PLACE (CIP) UTILITY

2.1 INTRODUCTION

The Compress-in-Place (CIP) utility enables you to consolidate free extents on nonsystem disks (disks mounted for exclusive use) without performing a full volume backup and volume restore. You can run CIP only on volumes that have no open files.

Disk volumes are divided into a number of physical records called blocks. Each block is 2048 bytes (2 KB) in length. When a file is created, the operating system allocates enough space on the disk for the file by reserving a number of contiguous blocks. A group of contiguous blocks allocated for use by a particular file is called an extent. A group of contiguous blocks that is not allocated to any file is called a free extent. Extents and free extents can be fragmented on a disk as files expand and contract. The CIP utility can significantly improve system performance by consolidating extents and free extents on the disk.

The CIP utility uses the disk space more efficiently by gathering the nonfree extents to the outer portions of the disk where they can be more quickly accessed by a disk drive's Read/Write heads. The free extents are gathered to the inner disk which results in disk seek time savings. Allocation of space for new files is made easier for the operating system. When the free extents are consolidated, the operating system does not have to search for an appropriate space in which to allocate a new file because it knows the starting address of the free extent area. System performance improves because the seek time decreases. For more information about extents, refer to the *VS Operating System Services Reference*. Figure 2-1 shows an overview of CIP processing.

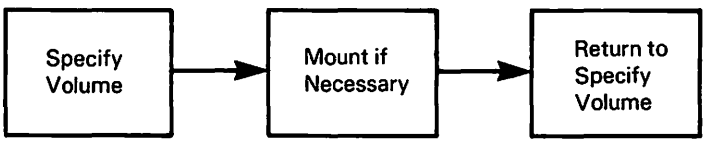


Figure 2-1. CIP Processing

2.2 CIP RESTRICTIONS

Observe the following restrictions when you run CIP:

- Only volumes that are available for mounting for exclusive use can be compressed. A shared volume cannot be compressed until all users are locked out of processing, since CIP updates and moves data blocks. (This creates an exclusive environment).
- CIP must be run by a task that has a modifiable data area size of at least 512 KB. A version of CIP for standalone VS systems is also available. For more information about the VS standalone utilities, refer to the individual processor handbook for your system.
- If the volume is not crash- or media-tolerant and the CIP utility is terminated abnormally, run LISTVTOC to verify that the VTOC is intact. If the VTOC is damaged, run the Backup and Restore options of the BACKUP utility to restore the disk.
- If an I/O error occurs, CIP displays a message that includes the error I/O status word (IOSW) in hexadecimal notation. You must press ENTER to acknowledge the error and to terminate CIP processing.

CAUTION

Before you run the CIP utility, you should run LISTVTOC to ensure that the VTOC is intact. LISTVTOC checks the VTOC for bad blocks (damage).

If the VTOC is damaged, do not use the CIP utility. Running the CIP utility with a damaged VTOC may cause the system to write access information to bad blocks in the VTOC. Instead, run the Backup and Remove options of the BACKUP utility to remove any bad blocks from the VTOC. (For more information about LISTVTOC, refer to Chapter 17. The BACKUP utility is described in the VS System Operator's Guide.)

2.3 CIP PROCESSING

When CIP processing begins, the CIP Input Definition screen (Figure 2-2) prompts you to specify the volume (and the VSID, for volume sets) that you want to compress. The CIP utility automatically mounts the volume for exclusive use. A message appears on the screen if you have any open files; CIP does not continue processing until the volume is freed for exclusive use.

NOTE

You cannot compress an entire volume set as a whole, but you can compress each member of a set separately. You do not need to have the root volume mounted to compress a volume set member. Compressing each disk in the volume set improves system performance.

After you specify the volume name (and VSID), press ENTER. If the volume that you specified is not mounted, a Mount screen appears. The Mount screen prompts you to specify the address of the device on which the disk is to be mounted, and the type of platter (REMOVABLE or FIXED) that is to be compressed. Press ENTER after you have specified the fields on the Mount screen.

```
***  MESSAGE I000 BY CIP

                INFORMATION REQUIRED BY  PROGRAM  CIP
                          TO DEFINE INPUT
                      ACTIVE SUBPROGRAM IS CIP


        *** Wang VS Compress-in-Place Program -- Version  x.xx.xx ***

This program will consolidate the free extents on a disk for the purpose
of providing useful free space on the disk and improving performance by
reducing the disk seek time.

Please specify the name of the volume to be compressed and press (ENTER):

                                VOLUME   = ******

                                VSID     = ***


                        Press (16) to terminate the program
```

Figure 2-2. CIP Input Definition Screen

When CIP processing is complete, CIP displays the CIP Input Definition screen again. You can then mount another volume and run CIP again, or you can exit the CIP utility.

The CIP utility can be run on any volume of a volume set. It is especially practical to run CIP on volumes of a volume set that are not currently in use because CIP can run independently of other processes. If certain volumes of a volume set are used almost constantly, then periodically close all files and lock out users to run CIP. Regular rotation of volumes on a volume set for CIP maintenance can help ensure better system performance.

You should not use CIP if a volume contains a large number of files. If the disk pack is nearly full of files on the inner and outer portions of the disk, the compression of the relatively "few and far between" free extents may not be worth the down time of the disk. Also, it would not significantly improve seek time because nearly full volumes have data packed on both the inner and outer portions of the disk.

2.4 CIP VERSUS BACKUP

The advantage of CIP versus BACKUP is weighed by the amount of time that is spent on compressing free extents against the need for a single extent. The decision to use either CIP or BACKUP to enhance system performance depends on the use of your system disk and the amount of time that has elapsed since the last system BACKUP to the disk. Both CIP and BACKUP perform similar functions on disk data (that is, compressing file extents). BACKUP is the optimum extent compressor because it creates a single, larger free extent from all the free extents on the volume. However, CIP is much faster to run.

The advantage of CIP is speed. CIP compresses extents and free extents on the disk quickly, but CIP does not consolidate the separate free extents into a single, larger free extent. Instead, the portion of the disk where the free extents are collected is fragmented into any number of **contiguous** free extents. When space allocation is required for a new file (or for the expansion of an old file), the operating system searches the free extents for an extent size that "best fits" the allocation size.

The BACKUP utility is more efficient than CIP for consolidating free extents because only one free extent remains on the disk when you run BACKUP. Space allocation using a single free extent occurs by taking only the amount of space that is required for the allocation from the outer portion of the free extent. This leaves a smaller, but still single, free extent.

You must determine whether disk seek times for free extents are slowed down enough to warrant the time it takes to run a full system BACKUP. When the number of free extents causes the system to spend too much time searching for a "bestfit" candidate, you should run BACKUP to consolidate all the free extents into a single free extent and then use CIP on a regular basis. This minimizes disk fragmentation and maintains an efficient disk. Regularly scheduled BACKUPs should be implemented for disk efficiency and integrity. BACKUPs are also crucial for recovering lost data in case of disk crash or natural disaster. Consult the *VS System Operator's Guide* for a complete description of BACKUP procedures and disk maintenance schedules.

2.5 A SAMPLE CIP PROCEDURE

You can control CIP processing through the VS Procedure language. You can specify all CIP options and mount operations through a procedure. Appendix A provides a complete list of CIP GETPARMs. For detailed information about the syntax of the VS Procedure language, refer to the *VS Procedure Language Reference*.

The following procedure mounts and compresses a volume. After the volume is compressed, the procedure exits the CIP utility and dismounts the volume. The procedure dismounts the volume after the completion of CIP processing because VS Procedure language DISMOUNT statements cannot be embedded in a RUN, DISPLAY, or ENTER sequence. The CIP utility Mount operation can be embedded in a procedure because it does not use the VS Procedure language MOUNT statement. The procedure totally automates CIP processing; you only physically mount and dismount the volume.

```
PROCEDURE
    RUN CIP
        ENTER INPUT VOLUME = ZENITH, VSID = 004
        ENTER MOUNT DEVICE = 018
        ENTER INPUT 16
    DISMOUNT DISK ZENITH
RETURN
```

CHAPTER 3
THE COPY UTILITY

3.1 INTRODUCTION

The COPY utility is used to copy files, libraries, and volumes from one secondary storage location on the system to another. It is also used to manipulate file organization. For more information about file structure and characteristics, refer to the *VS Data Management System (DMS) Reference* and the *VS DMS/TX Reference*.

When a library or a volume is copied, either a new output library or volume is created, the copied information is added to an existing library or volume, or the new data replaces the existing data. The COPY utility leaves the input files, library, and volume intact.

When a single file is copied, an output file is created to contain the copied information. If the file to be copied is NOT a print file, WP document, or a program, you can modify the output file (the copy) in the following ways:

- Select consecutive, indexed, or relative file organization
- Select fixed-length or variable-length records
- Allow the utility to compress records
- Reorganize a file
- Select the key length and key position of indexed files
- Change the packing density for index blocks and data blocks
- Rebuild indexed files by copying files one record at a time instead of one block at a time
- Create a copy of DMS/TX files with or without the recovery blocks
- Convert an indexed file to a DMS/TX file by creating recovery blocks
- Attach the copy of a DMS/TX file to the same data base to which the original file is attached

Your security access classification (assigned by a system administrator) allows you to copy any information of an equal or subordinate access class. Your access class is given to the information that you copy through the COPY utility. You become the owner of the copied files.

Figure 3-1 shows an overview of COPY processing.

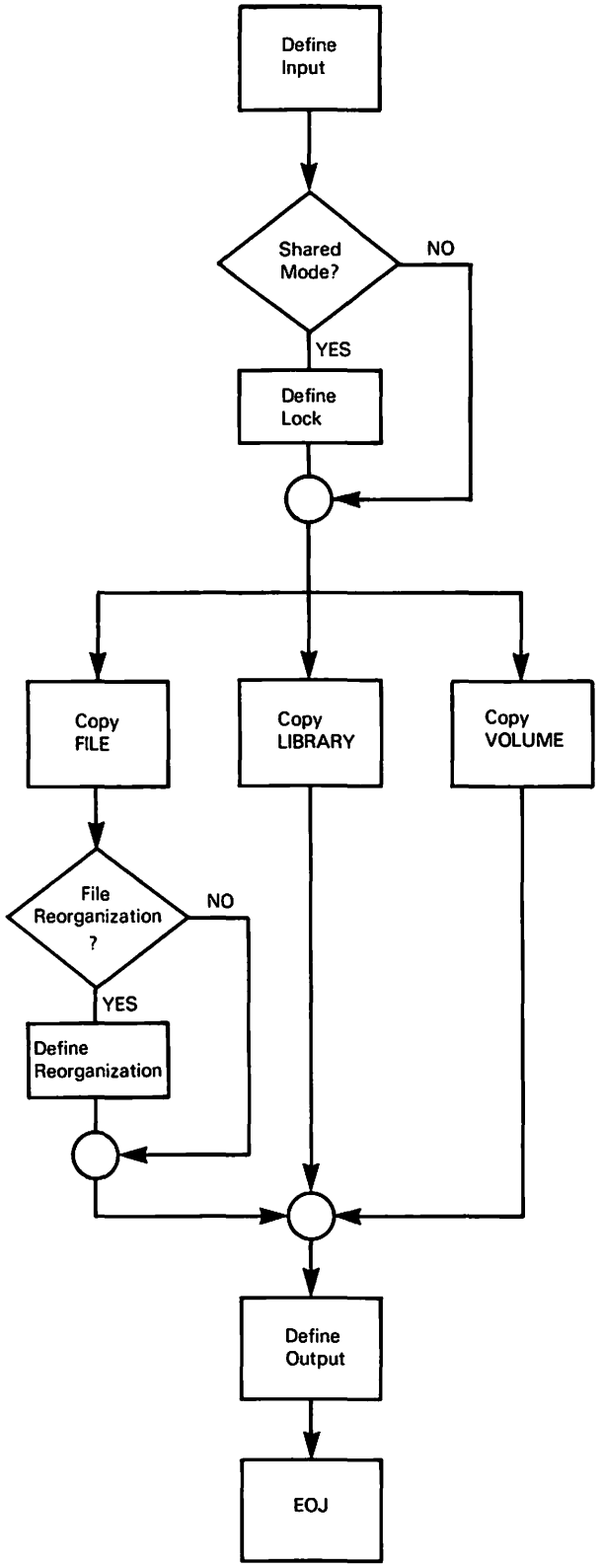


Figure 3-1. COPY Processing

The following procedure illustrates the three basic operations for the COPY utility:

1. For input, specify the file, library, and volume that you want to copy.
2. If necessary, specify any changes that you want in a file's organization and recovery capability.
3. For output, specify the file, library, and volume where you want the copied information to be placed.

NOTE
The COPY utility should not be used for system backup. If the copied data is a system file, or other file with special security access rights, the file's rights can be inadvertently changed. (The BACKUP utility is more appropriate for backup purposes because it copies file characteristics exactly as they are on the input version (disk to disk; not disk to tape). For more information about the BACKUP utility, refer to the VS System Operator's Guide.)

3.2 DEFINING THE INPUT

The first task to be done in the COPY utility is to define what is to be copied. The first screen of the COPY utility is the COPY Utility Input Definition screen (Figure 3-2). In the sample, the entity to be copied is a single file, designated by the COPY = FILE***. The file to be copied is named PATRON, in library ACCOUNTS on volume SYSTEM. The Mode field is specified on the screen as INPUT.

```
***  MESSAGE 0001 BY COPY

                INFORMATION REQUIRED BY  PROGRAM  COPY
                                TO DEFINE INPUT


WANG VS COPY PROGRAM - VERSION  x.xx.xx

PLEASE ASSIGN "INPUT"           (TO BE USED AS INPUT BY THE PROGRAM)

PLEASE SPECIFY THE COPY OPTION DESIRED:
    COPY     = FILE***          (OPTIONS = FILE, LIBRARY, OR VOLUME)

PLEASE SPECIFY THE FILE INFORMATION NEEDED:
    FILE     = PATRON** IN LIBRARY  = ACCOUNTS ON VOLUME    = SYSTEM

PLEASE SPECIFY THE MODE IN WHICH TO OPEN THE FILES TO BE COPIED:
    MODE     = INPUT*           (OPTIONS = INPUT OR SHARED)


                        Press PF16 to terminate COPY.
```

Figure 3-2. Sample COPY Utility Input Definition Screen

The fields on the COPY Utility Input Definition screen are described as follows:

Field	Description
COPY	Specify what you want to copy. The options are FILE, LIBRARY, or VOLUME. Any abbreviation of FILE, LIBRARY, or VOLUME is permitted. FILE is the default value for the Copy field.
FILE	Specify the name of the file to be copied. This entry is made only when copying a single file. If you are copying a library or volume, leave this field blank. The File field has no default value.
LIBRARY	If you are copying a single file, specify the name of the library where the file resides. If you are copying a whole library, specify the name of the library to be copied. If you are copying a volume, leave this field blank. The Library field has a default value of the INLIB (Input Library field) value which is set through the SET Usage Constants function (PF2) of the Command Processor menu.
VOLUME	Specify the name of the volume that you want to copy. The Volume field has a default value of the INVOL value that is set through the SET Usage Constants function (PF2) of the Command Processor menu.
MODE	Specify the mode (INPUT or SHARED) in which the COPY utility opens indexed and consecutive files. The Mode field has a default value of INPUT. If you specify SHARED for a library or volume copy, the COPY utility opens all indexed and consecutive files in Shared mode; it opens all other files in Input mode.

3.2.1 Copying Files in Shared Mode

If you specify SHARED in the Mode field for a library or volume copy, the COPY utility displays the COPY Utility Lock screen (Figure 3-3).

```
***  MESSAGE 0057 BY COPY

                INFORMATION REQUIRED BY  PROGRAM  COPY
                          TO DEFINE LOCK



PLEASE ENTER THE FOLLOWING OPTIONS TO PROCESS A FILE IN SHARED MODE.

SHOULD A LOCK BE PLACED ON THE FILE BEFORE COPYING?    LOCK     = YES (YES, NO)
  IF YES, NO CHANGES TO THE FILE CAN BE MADE DURING
  THE COPY.  IF NO, CHANGES TO THE  FILE CAN BE MADE.

IF LOCK=YES, SPECIFY TIMEOUT AND BYPASS:

HOW MANY SECONDS SHOULD THE TIMEOUT BE?                TIMEOUT  = 10* SECONDS

IF THE TIMEOUT EXPIRES, SHOULD THE FILE BE BYPASSED?   BYPASS   = NO* (YES, NO)
```

Figure 3-3. Sample COPY Utility Lock Screen

The COPY Utility Lock screen fields are described as follows:

Field	Description
LOCK	Specify (YES or NO) whether you want to suspend updates to a file that you are displaying. If you lock a file (YES), no changes to the file can occur while you are displaying it. If you specify NO, no lock is placed on the file, and there is no need to specify the Timeout and Bypass options. The default value is YES.
TIMEOUT	Specify a Timeout value (0 – 255 seconds) for a file if Lock = YES. If another user currently holds the file for update, the Timeout field specifies the length of time that the COPY utility will wait to open the file in Shared mode with a lock. The default value is 10 seconds.
BYPASS	Specify (YES or NO) whether you want to skip a file if the Timeout value expires. If BYPASS = YES and the timeout expires, the COPY utility skips the file. If BYPASS = NO and the timeout expires, the Lock screen reappears with the message "FILE filename IN libname ON volname IS HELD BY USER xxx." The default value is NO.

When timeout expiration occurs, you can continue with the copy operation; redefine the Lock, Timeout, and Bypass options, and press ENTER. The COPY utility also displays a message that informs you of the following options:

- You can skip the particular file on which the timeout occurred by pressing PF1.
- You can terminate the copy operation if you are copying a library or volume, by pressing PF16.

If a timeout expires with a Bypass value of NO during a background task, the copy operation is automatically cancelled. If a timeout expires with a Bypass value of YES for either foreground or background tasks, the copy operation proceeds, but a copy of the file that was bypassed is not contained in the output.

NOTE
The Record Access Method (RAM) is always used to copy shared files. For more information about RAM, refer to the VS Data Management System (DMS) Reference.

3.3 COPYING A FILE

Files can be copied from within a library or from one library to another. The original and output libraries can exist on the same volume or on different volumes. Original files are never altered during a copy operation. Depending on the options that you choose when you complete the COPY Input Definition screen, COPY processing gives you the option of changing the name and organization of the file's output copy.

- If the file to be copied IS NOT a print file, WP document, or a program file (executable object file), then the COPY Utility Options screen (Figure 3-4) is displayed after the COPY Utility Input Definition Screen (Figure 3-2). After the options have been selected, the COPY Utility Output Definition screen (Figure 3-5) appears.
- If the file to be copied IS a print file, WP document, or a program file, then a COPY Utility Output Definition screen (Figure 3-5) is displayed after the COPY Utility Input Definition Screen (Figure 3-2).

3.3.1 Defining Options

The file organization of the output file is the same as the input file, unless you change the organization of the output file through the COPY Utility Options screen (Figure 3-4).

```
***  MESSAGE 0030 BY COPY

                INFORMATION REQUIRED BY  PROGRAM  COPY
                             TO DEFINE OPTIONS


PLEASE INSPECT (AND MODIFY IF NECESSARY) THE OUTPUT FILE ORGANIZATION:

    FILEORG  = C   OPTIONS = C-CONSECUTIVE, I-INDEXED, R-RELATIVE
    LENGTH   = V   OPTIONS = F-FIXED LENGTH, V-VARIABLE LENGTH
    COMPRESS = Y   OPTIONS = "Y" OR "N"
    REORG    = NO*   (TO REORGANIZE YOUR FILE, TYPE "YES")

FOR INDEXED FILES, PLEASE SPECIFY:

    KEYLEN   = ***
    KEYPOS   = ***** (FROM 1)
    IPACK    = 100 % (PACKING DENSITY FOR INDEX BLOCKS)
    DPACK    = 100 % (PACKING DENSITY FOR DATA BLOCKS)
    RECBLK   = N     DMS/TX RECOVERY BLOCKS (N=NONE, A=ALLOCATED, U=USED)
```

Figure 3-4. Sample COPY Utility Options Screen

The fields on the COPY Utility Options screen are described as follows:

Field	Description
FILEORG	Specify C (Consecutive), I (Indexed), or R (Relative) file organization. By using this option, you can supply consecutive files with an index and supply indexed files with a new primary key. You can also convert a relative file to a consecutive or indexed file, or you can convert a consecutive or indexed file to a relative file. The default value for FILEORG is C.
LENGTH	Specify F (Fixed-length) or V (Variable-length) records. If data compression is used, you must specify V (Variable length) in the Length field. If you change the length from variable to fixed, a pad character is required. The COPY Pad Definition screen appears after the COPY Utility Output Definition screen (Figure 3-5) is displayed, enabling you to select either a blank character or a hexadecimal zero as the pad character. The default value for the Length field is V.
COMPRESS	Specify Y (Yes) or N (No) for data compression. Data is compressed if you accept the default value (Y). Data compression can produce more efficient use of storage. When data is compressed, set the Length field to V.

Field	Description
REORG	Specify YES, if you want to reorganize the output copy. This field is meaningful when an indexed file is to be copied with no change to the file attributes. The REORG option specifies whether the file is copied in Block mode or Record mode. The REORG option can also be used to rebuild an indexed file that has errors in its index structure. You can also use the REORG option to improve file access time. If you specify YES in this field, the system copies the file in Record mode. In Record mode, only the input file's data blocks are read, new index blocks are generated, and the data blocks are consolidated. As a result, the file assumes its most optimum form: high density data blocks and high density, single-level, primary index blocks. The REORG field normally has a default value of NO. A DMS/TX file in an inconsistent state causes the REORG field to default to YES. This results in the file being restored to its original, pre-crash state. Refer to the *VS DMS/TX Reference* for details on DMS/TX protocol.

If the output file is indexed, the following specifications are available:

Field	Description
KEYLEN	Specify the length (in bytes) of the index key. The default value is the primary index key length of the input file.
KEYPOS	Specify the position number of the first byte in the index key field. KEYPOS defaults to the position of the first byte in the input file's primary index key field.
IPACK	Specify the density at which index blocks are packed. The IPACK default value is 100 percent.
DPACK	Specify the density at which data blocks are packed. The DPACK default value is 100 percent.
RECBLK	Specify N (None), A (Allocated), or U (Used) for the DMS/TX recovery option. The RECBLK default value is N for non-DMS/TX files. The A option allocates recovery blocks in the output file, even if the input file is a non-DMS/TX, indexed file. The U option causes COPY to attach the output file to the same data base to which the input file is attached. Refer to the *VS DMS/TX Reference* for more information.

3.3.2 Packing Densities

The system can pack index or data blocks at any density percentage. To illustrate the effects of packing densities, consider the following example. You have a file with 400 records. Each record is 256 bytes, which means 8 records can fit neatly into one 2K block (2048 bytes). Therefore, you need at least 50 blocks at 100% packing density to contain the file. When you copy the file and specify 100% packing density, the operating system allocates 50 blocks and copies the file.

If you want to update your file at a later date, the operating system finds an available block and allows additional information to be placed on that block. Subsequent updates require more blocks and the operating system gets those blocks wherever they are available. This can cause file fragmentation. The operating system keeps track of all of the files' blocks, no matter how fragmented the file may get, but file fragmentation can reduce throughput due to the larger number of seek operations the operating system has to perform.

If you want to keep the blocks in the file contiguous even when you have to update, you can allocate a lower packing density. A packing density of 50% at file creation enables the operating system to fill each block in the file only half way (allowing 4 records per block). This allows for expansion of the file without retrieving a block from memory that may not be contiguous to the file. The disadvantage is that a 50% packing density also means that each block has half of its space empty at file creation and, depending on update activity, significant space may remain unused.

Before you set the packing densities of the index and data blocks, you must determine the use of the file to be copied. If you do not expect to update the copied file, then set the packing density at 100% to optimize disk space. If fragmentation is not an issue, a high density factor can be used. However, if your file requires numerous updates and modifications, it may be more beneficial to allocate space in the beginning to keep future growth of the file within its contiguous blocks than to allow it to become so fragmented as to slow down the processing of the file.

3.3.3 Defining the Output

After you have selected the file, library, or volume to be copied and have selected the reorganization options for the output file, if any, the COPY Utility Output Definition screen (Figure 3-5) appears. Output definition screens for a library or volume require only a destination volume or volume and library.

```
***  MESSAGE 000  BY OPEN

                INFORMATION REQUIRED BY  PROGRAM  COPY
                            TO DEFINE OUTPUT




PLEASE ASSIGN "OUTPUT"          (TO BE CREATED AS OUTPUT BY THE PROGRAM)

TO ASSIGN THIS FILE TO A DISK FILE, PLEASE SPECIFY:
    FILE     = SNOOPY** IN LIBRARY  = DOCMNTR* ON VOLUME   = WP****
                        RETAIN   = 999 DAYS    RELEASE  = YES
    FILECLAS = #
TO SELECT ANOTHER DEVICE, SPECIFY:
    DEVICE   = DISK*******   (ALTERNATES =DISK,PRINTER)


    PRINTER OPTIONS:     PRTCLASS = A    FORM#    = 000    COPIES   = 00001
```

Figure 3-5. Sample COPY Utility Output Definition Screen

NOTE

Depending on the type of file you are copying, the COPY Utility Output Definition screen appears with some output fields on one output screen that do not appear on others. In Figure 3-5, the sample screen shows a COPY output screen for the print file SNOOPY. Some of the fields on the COPY Utility Output Definition screen do not appear when you copy program files, word processing files, or other files.

The fields for a single file Copy Utility Output Definition screen are described as follows:

Field	Description
FILE	Specify the name of the output (destination) file.
LIBRARY	Specify the library name where the file is to reside.
VOLUME	Specify the volume name on which the library resides.
RETAIN	Specify the length of time that the file is protected from deletion. A file can be protected for a maximum of 999 days and a minimum of 0 days. The Retain field defaults to the input file's retention period.
RELEASE	Specify whether or not unused storage extents, previously allocated for data, can be released for use by other files. A value of YES releases the unused extents, a value of NO does not. The Release field defaults to the input file specification. Unused extents begin after the last used block. If an index file is allocated for a large number of records, the root index block is created at a block near the middle of the file, thus "using" all prior blocks even though they may be empty.
FILECLAS	Specify the protection class of the output file. The protection class determines what access privileges (if any) a user has to the file. The following file protection classes are available: #, $, @, blank, and A to Z. (For more information on file class options, consult the *VS System User's Introduction*.)
DEVICE	Specify the device to which the output file is copied. When a file is copied to diskette or hard disk, specify DISK in the Device field. If the output file is a print file, specify PRINTER as the device. (The COPY utility does not copy to tape. The TAPECOPY utility, described in Chapter 25, is provided for this purpose.) The Device field defaults to DISK.

You also have the following options if the input file is a print file:

Field	Description
PRTCLASS	Specify the print class of the file. Print classes are identified with a 1-letter code from A to Z. What you assign to this field determines the printer on which the file is to be printed and the priority of the print file. Each printer on a system is assigned a list of print classes.
FORM#	Specify the form type (0 to 255) to be used for printing a file.
COPIES	Specify the number of copies of the file that you want printed. The default value for the Copies field is 1.

3.3.4 Mounting a New Volume

If you have specified a volume that is not mounted, a message appears at the top of the COPY Utility Output Definition screen that prompts you to perform one of the following actions:

- Specify the disk device number in the Device field and press PF4.
- Respecify the output volume.

The disk device number is actually a concatenation of the word "DISK" and the disk drive number. For example, if the disk drive was Number 5, specify DISK5 in the Device field and press PF4. DISK12X would indicate disk drive Number 12 (the X indicates exclusive use). If you use PF4 to mount a new volume, another screen appears that prompts you to physically mount the disk. When this is done, the COPY utility automatically resumes processing.

3.3.5 Naming Conflicts

If you assign a name to the output file that is identical to an existing file name in an existing library, a naming conflict occurs and a message appears on the screen giving you two options. The COPY Utility Output Definition screen cannot be processed and COPY does not proceed until you resolve the naming conflict. File naming conflicts are handled differently when they occur during library and volume copying (refer to Section 3.4.1.). You can resolve a file name conflict by performing one of two actions:

1. Press PF3. This deletes a file that already resides in the output library. When the existing file is deleted, the copied file of the same name can be copied into the library.
2. Respecify the name of the output (copied) file. This allows you to specify a unique name for the copied file in the output library. You can also respecify the library or volume to place the copied document into a different location.

When COPY processing has completed, the COPY Utility End-of-Job (EOJ) screen appears. This screen displays the message FILE CLIENT IN LIBRARY ACCOUNTS ON VOLUME SYSTEM CREATED WITH 000095 RECORDS. (CLIENT, ACCOUNTS, and SYSTEM are sample names. These names depend on the output values that you specify.) From this screen, you can terminate COPY processing by pressing PF16 or restart the program by pressing PF1. If you press PF1, the COPY Utility Input Definition screen appears again.

3.4 COPYING A LIBRARY

To copy a library, perform the following steps:

1. Specify LIBRARY as the Copy field value on the COPY Utility Input Definition screen (Figure 3-2).
2. Specify the input library that you want to a copy and the volume on which it resides. Press ENTER to process the information on the screen.
3. Specify the output library and the volume on the COPY Utility Output Definition screen (Figure 3-5). If you are copying to a new volume, follow the mount procedure outlined in Section 3.3.4.

When the output screen is processed, the COPY utility copies files from the named input library to the designated output library. The output library can reside either on the same volume as the input library or on another volume. Files stored in the existing output library are not affected by the copy operation, nor is the input library destroyed. File organization cannot be changed by the COPY utility when you copy a library. If the specified output library already exists, then new files are added to the existing library. If the specified output library does not already exist, it is created by the COPY utility.

If the utility encounters a file name from the input library that has the same file name as a file that exists in the output library, a naming conflict occurs and the copy operation halts. You must resolve the conflict before copy processing can continue.

3.4.1 Resolving Duplicate File Names

When the COPY utility encounters a duplicate file name while copying a library or a volume, the program halts and the COPY Utility Naming Conflict Options screen (Figure 3-6) appears informing you of the conflict. This screen identifies the name of the file name found in both the input and output libraries.

```
***  MESSAGE 0018 BY COPY

                INFORMATION REQUIRED BY  PROGRAM  COPY
                            TO DEFINE OPTIONS



FILE 0000     IS LISTED IN BOTH THE INPUT LIBRARY DOCMNTR
                         AND THE OUTPUT LIBRARY DOCMNTTT

PLEASE SPECIFY THE OPTION DESIRED:

    OPTION   = N          OPTIONS: N - NO COPY WILL TAKE PLACE FOR THIS FILE
                                   S - SCRATCH THE OUTPUT FILE BEFORE COPYING
                                   R - RENAME THE FILE THAT IS
                                       ALREADY IN THE OUTPUT LIBRARY
                                   C - COPY THE FILE USING A NEW FILE NAME

    NEWNAME  = ********       (FOR OPTIONS R AND C)
```

Figure 3-6. Sample COPY Utility Naming Conflict Options Screen

3.4.2 Example File Name Conflict Resolution

Table 3-1 gives an example of some naming conflicts between an input library and an output library. Library ALPHA has been selected to be copied into library BETA.

Table 3-1. Sample Libraries with File Name Conflicts

Input Library ALPHA		Output Library BETA	
0001	0008	0004	0017
0002	0013	0005	0018
0003	0019	0006	0019
0006	0020	0007	0025
0007	0025	0016	

You are prompted to specify the method for solving the naming conflict on the COPY Utility Naming Conflict Options screen. In Table 3-1, for example, four naming conflicts occur when library ALPHA attempts to copy the files, 0006, 0007, 0019, and 0025 into library BETA.

In the following illustration, the naming conflicts in libraries ALPHA and BETA (Table 3-1) were resolved as follows:

1. **N** — ALPHA file 0006 is **not copied** because the **N option** is specified to resolve this naming conflict. File 0006 is not deleted from library ALPHA or BETA.
2. **S** — BETA file 0007 is **scratched** (deleted) in order to place ALPHA file 0007 in its place. BETA file 0007 is scratched by specifying the **S option** to resolve this naming conflict.
3. **R** — BETA file 0019 is **renamed** to file 1019. This was done by specifying the **R option** to resolve the naming conflict and entering 1019 in the Newname field.
4. **C** — ALPHA file 0025 is **copied** but with an output file name of 1025 to resolve this naming conflict. This was done by specifying the **C option** to copy the file by changing the Newname field value to include the new output file name.

Table 3-2 shows the resulting library BETA when Steps 1 through 4 are performed on Table 3-1. Library ALPHA is not affected by the COPY operation.

Table 3-2. Results of Copying Library ALPHA into Library BETA

BETA File Name	Library Origin
0001	ALPHA
0002	ALPHA
0003	ALPHA
0004	BETA
0005	BETA
0006	BETA
....	ALPHA 0006 not copied
0007	ALPHA
....	BETA 0007 scratched
0008	ALPHA
0013	ALPHA
0016	BETA
0017	BETA
0018	BETA
0019	ALPHA
0020	BETA
0025	ALPHA
1019	renamed from BETA 0019
1025	copied from ALPHA 0025

Users may assign the same file name for similar types of data stored in different libraries. Naming conflicts are often resolved by slightly altering one of the file names in a COPY operation. For example, if library X and library Y both have a file called Account, when library X is copied into library Y, a naming conflict will occur. If library X's account contained data on people from Department 2, the file might be renamed to Account2. This resolves the naming conflict by assigning a unique but similar name to one of the files in conflict. The user can still draw a logical connection between the file names and the data contained therein.

After the input library has been successfully copied to its assigned output library, the COPY Utility End-of-Job screen is displayed. You can terminate COPY processing by pressing PF16, or you can restart the program by pressing PF1.

CAUTION

Use caution when you copy a file that is part of a database structure (i.e., Wang OFFICE or PACE). Be careful not to rename any database files. Changing the name of such a file could cause many problems for your database because other files may reference the old file (which has just had a name change).

Also, because WP documents contain header references and other structures, the COPY utility is not recommended for copying WP documents. Use the COPYWP utility to copy WP documents (refer to Chapter 6).

3.5 COPYING A VOLUME

To copy a volume, perform the following steps:

1. Specify an input volume on the COPY Utility Input Definition screen. Press ENTER to process the screen information and to display the COPY Utility Output Definition screen.
2. Specify the output volume on the COPY Utility Output Definition screen. When the screen information is processed, files from the specified input volume are copied to the designated output volume.

Files from each library are copied into corresponding libraries of the designated output volume. When no corresponding library appears on the output volume, the library is created and added to the volume.

If a file and a library on the input volume have the same names as a file and a library on the output volume, a naming conflict occurs. The copy operation does not proceed until you resolve the conflict. File name conflicts during a volume copy are resolved in the same manner as they are during a library copy. Refer to Section 3.4.1 for a description of resolving duplicate file names. The input volume is not affected by the copy operation.

File organization cannot be changed by the COPY utility when performing a volume copy.

When the input volume is successfully copied to the designated output volume, the COPY Utility End-of-Job screen is displayed. You can terminate COPY processing by pressing PF16 or restart the program by pressing PF1.

3.6 A SAMPLE COPY PROCEDURE

If you use the COPY utility regularly with the same input and output options and fields, you may find it helpful to write a procedure that will automatically provide the repeated fields. You can control the COPY process through the VS Procedure language. This section includes a sample procedure as a model to help you write a customized procedure. The COPY utility acquires all necessary information through GETPARM requests. A list of COPY GETPARMs is given in Appendix A. For details concerning procedure syntax, refer to the *VS Procedure Language Reference*.

The sample procedure performs the periodic reorganization of a frequently modified indexed file. The procedure specifies that the PATRON file is to be copied from the ACCOUNTS library on the SYSTEM volume and placed on a newly created output file, CLIENT (on the same library and volume). The procedure also selects the reorganization option with new packing densities of 50 percent.

NOTE

You should delete any existing versions of the output file before running the procedure. The COPY utility enables you to correct a naming conflict during the run of a procedure through a "CORRECTION REQUIRED BY..." screen.

```
PROCEDURE
    RUN COPY
        ENTER INPUT COPY=FILE, FILE=PATRON,
            LIBRARY=ACCOUNTS, VOLUME=SYSTEM
        ENTER OPTIONS REORG=YES, IPACK=50, DPACK=50
        ENTER OUTPUT FILE=CLIENT, LIBRARY=ACCOUNTS,
            VOLUME=SYSTEM
        ENTER EOJ 16
RETURN
```

CHAPTER 4
THE COPYOIS UTILITY

4.1 INTRODUCTION

The COPYOIS utility transfers Office Information System (OIS) files to and from the VS. This utility allows you to convert OIS BASIC source programs to VS BASIC source programs and OIS data files to VS data file format. COPYOIS enables you to implement integrated data processing and word processing applications between the VS and OIS environment. You can run COPYOIS interactively, through a procedure, or through Assembly, BASIC, COBOL, or PL/I programming languages. With the COPYOIS utility, you can perform any of the following functions:

- Initialize a diskette with an OIS standard label (Section 4.4).
- Copy single or multiple OIS list processing data files, or BASIC source and data files from the VS to OIS diskette (Section 4.5).
- Copy single or multiple OIS files from an OIS diskette to the VS. You can also convert OIS files to VS files (Section 4.6).
- Create, display, or print the diskette catalog listing of OIS files on an OIS diskette (Section 4.7).
- Rename an OIS file on the OIS diskette (Section 4.8).
- Delete an OIS file from the diskette (Section 4.9).
- Assign a password to the OIS diskette volume or to any file residing on the diskette (Section 4.10).
- Convert an OIS file already residing on the VS (through a previous copy or a telecommunications operation) to a VS file (Section 4.11).

Figure 4-1 shows an overview of COPYOIS processing.

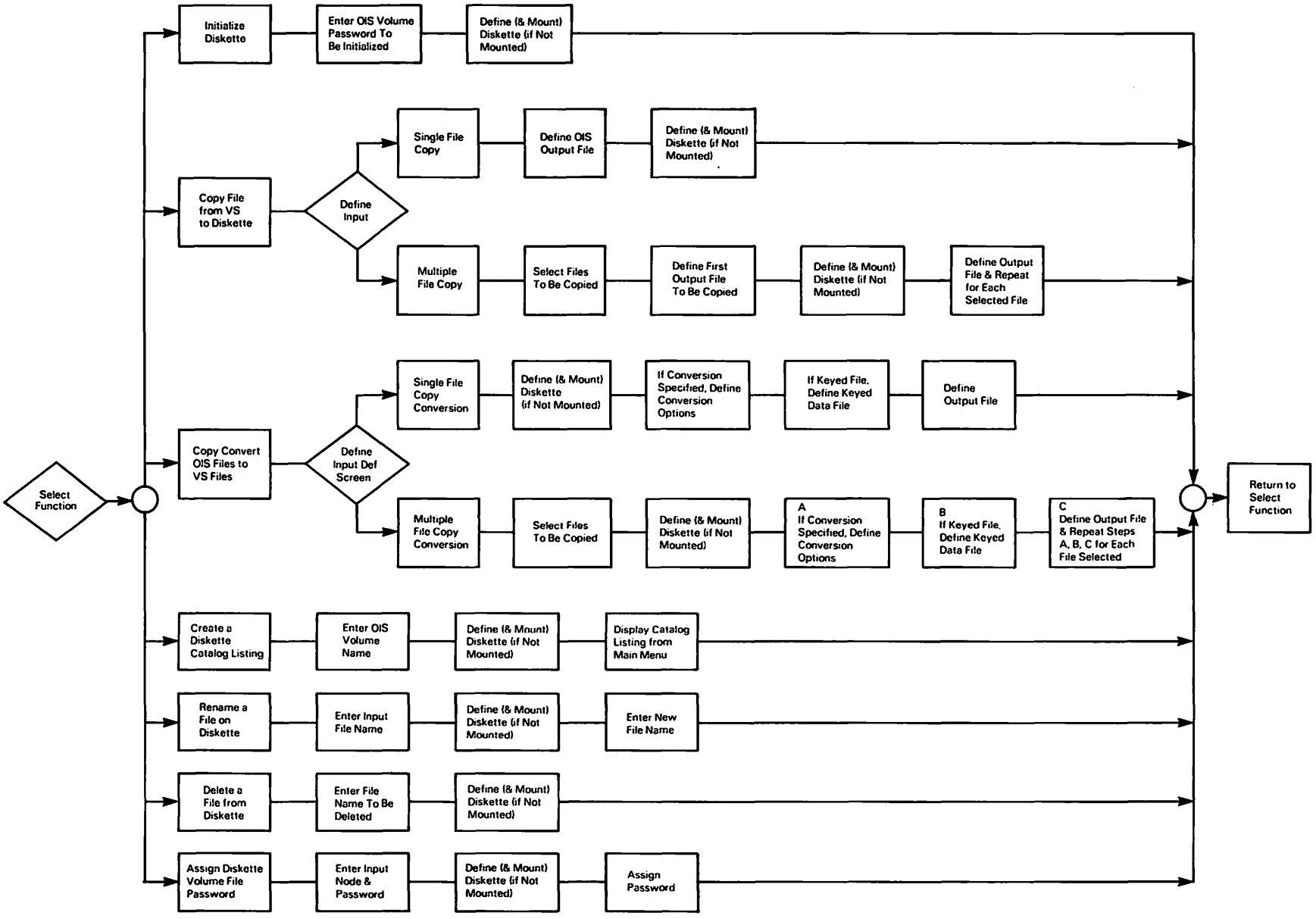


Figure 4-1. COPYOIS Processing

NOTE
The COPYOIS utility supports the use of all hard-sectored and 5 1/4-inch soft-sectored diskettes.

4.2 FUNCTION SELECTION

When COPYOIS processing begins, the COPYOIS Function Selection screen (Figure 4-2) appears. You can select any one of the functions listed in Section 4.1 by pressing the corresponding PF key.

```
***  MESSAGE 0001 BY CPYOIS

                INFORMATION REQUIRED BY  PROGRAM  COPYOIS
                             TO DEFINE MENU



          *** Wang VS COPYOIS Utility Program - Version x.xx.xx ***



            Press the PFkey corresponding to the desired operation:

(1) Initialize Diskette                 (7) Rename File on Diskette
(2) Copy File from VS                   (8) Delete File from Diskette
                                        (9) Assign Diskette Volume/File Password
(4) Copy/Convert OIS File
(5) Create Diskette Catalog Listing

                                       (16) Terminate Processing
```

Figure 4-2. COPYOIS Function Selection Screen

4.3 MOUNTING VOLUMES

You can mount any VS or OIS volume from within a COPYOIS function. Whenever you specify the name of an unmounted volume, COPYOIS displays the COPYOIS Mount Volume screen (Figure 4-3).

```
*** MESSAGE 0001 BY CPYOIS

                INFORMATION REQUIRED BY PROGRAM COPYOIS
                          TO DEFINE MOUNT



                        OIS Volume Name = Diskette

                    Please identify the volume to be mounted:

                VOLUME   = ******      (VS Volume Name)
                DEVICE   = DISKETTE    (DISK or DISKETTE)
                UNIT     = ***         (Device Unit Number)
                PASSWORD = ********    (OIS Diskette Only)

            Press ENTER to continue or PF1 to terminate this operation
```

Figure 4-3. Sample COPYOIS Mount Volume Screen

The COPYOIS Mount Volume screen uses the same fields for VS and OIS volumes. However, the fields are interpreted differently in each case. The fields for the VS and the OIS volumes are as follows:

VS Volume Fields

Field	Description
VOLUME	Specify the name of the VS volume that you want to mount.
DEVICE	Specify this field as DISK or DISKETTE (depending on the volume to be mounted). The Device field value defaults to DISK for VS volumes. Note that a diskette must be a VS standard label (SL) diskette.
UNIT	Specify the device number of the disk or diskette drive.
PASSWORD	This field cannot be modified for VS volumes because a password is not relevant for VS volumes.

OIS Volume Fields

Field	Description
VOLUME	Specify the VS name for the OIS volume that you want to mount.
DEVICE	This field cannot be modified because COPYOIS can only access OIS diskettes. The value specified is DISKETTE.
UNIT	Specify the device number of the diskette drive.
PASSWORD	This field is optional, unless you need to initialize an OIS diskette that already has a password. If you plan to initialize a diskette within the COPYOIS session, enter the correct password. The COPYOIS Mount Volume screen appears if you attempt to initialize the diskette and did not enter the password. Also, you must enter the correct password if you are initializing a diskette that was not previously mounted.

COPYOIS recognizes an OIS volume by the OIS volume name, and not by the VS volume name used for the physically mounted OIS diskette. If you enter an OIS volume name (for input, output, or initialization) that is different from the name on the OIS diskette, COPYOIS assumes that you want to mount a new volume and the COPYOIS Mount Volume screen appears.

4.4 INITIALIZE DISKETTE (PF1)

You can initialize a previously initialized VS nonlabeled (NL) diskette to the standard OIS DOS format through COPYOIS by pressing PF1, the Initialize Diskette function key. (VS NL diskettes are initialized by using the DISKINIT utility. Refer to Chapter 7 for more information.) The diskette is made identical to a diskette that is initialized as a DOS volume on an OIS and can then be used by COPYOIS as an output medium to transfer OIS files to an OIS. COPYOIS performs the same integrity checking as the DISKINIT utility (refer to Chapter 7).

When you press PF1, the OIS Volume Definition screen appears. If you have not already mounted the diskette, the COPYOIS Mount Volume screen appears and prompts you to mount the diskette (refer to Section 4.3). The OIS Volume Definition screen fields are defined as follows:

Field	Description
VOLUME	Specify the name of the OIS volume that you want to initialize.
PASSWORD	Specify a password for the OIS volume (optional).

To return to the COPYOIS Function Selection screen (Figure 4-2) without initializing a diskette, press PF1 again. After the diskette is mounted and the fields have been set, COPYOIS initializes the volume. When the diskette is initialized, COPYOIS displays the COPYOIS Function Selection screen (Figure 4-2), with a completion message that displays the OIS and VS volume names.

4.5 COPY FILE FROM VS (PF2)

COPYOIS enables you to copy unconverted OIS files residing on the VS to an OIS diskette by pressing PF2, the Copy File from VS function key. You can copy a single file or you can copy a group of files that you select from a VS library that contains OIS files. The OIS files can be unconverted list processing data files, unconverted OIS BASIC source files or data files (that have been previously transferred to the VS by COPYOIS, a Wang Systems Networking (WSN) file transfer, or TCCOPY). When PF2 is pressed, the Copy Files from VS Input Definition screen (Figure 4-4) appears.

```
*** MESSAGE 0001 BY CPYOIS

                INFORMATION REQUIRED BY PROGRAM COPYOIS
                          TO DEFINE VSFILE



      Please specify the File, Library and Volume to copying one file
                or the Library and Volume for multiple file copy.

    Please provide the VS file information to perform the desired copy:

  FILE    = ********   in   LIBRARY  = #MMPRT**   on   VOLUME   = WP****


            Press .(ENTER) to copy the specified file
                   (1)   to terminate this file copy
                  .(9)   for multiple file copy
```

Figure 4-4. Sample Copy Files from VS Input Definition Screen

The fields for the Copy Files from VS Input Definition screen (Figure 4-4) are described as follows:

Field	Description
FILE	Specify the name of the OIS file on the VS that you want to copy. A value in this field is required only when copying a single file.
LIBRARY	Specify the VS library in which the OIS file(s) reside.
VOLUME	Specify the VS volume on which the OIS file(s) reside.

4.5.1 Copying a Single File

To copy a single file, specify the file that you want to copy on the Copy File from VS Input Definition screen and press ENTER. COPYOIS then requests you to specify the OIS output (destination) volume and file names on the COPYOIS Output File Definition screen (Figure 4-5).

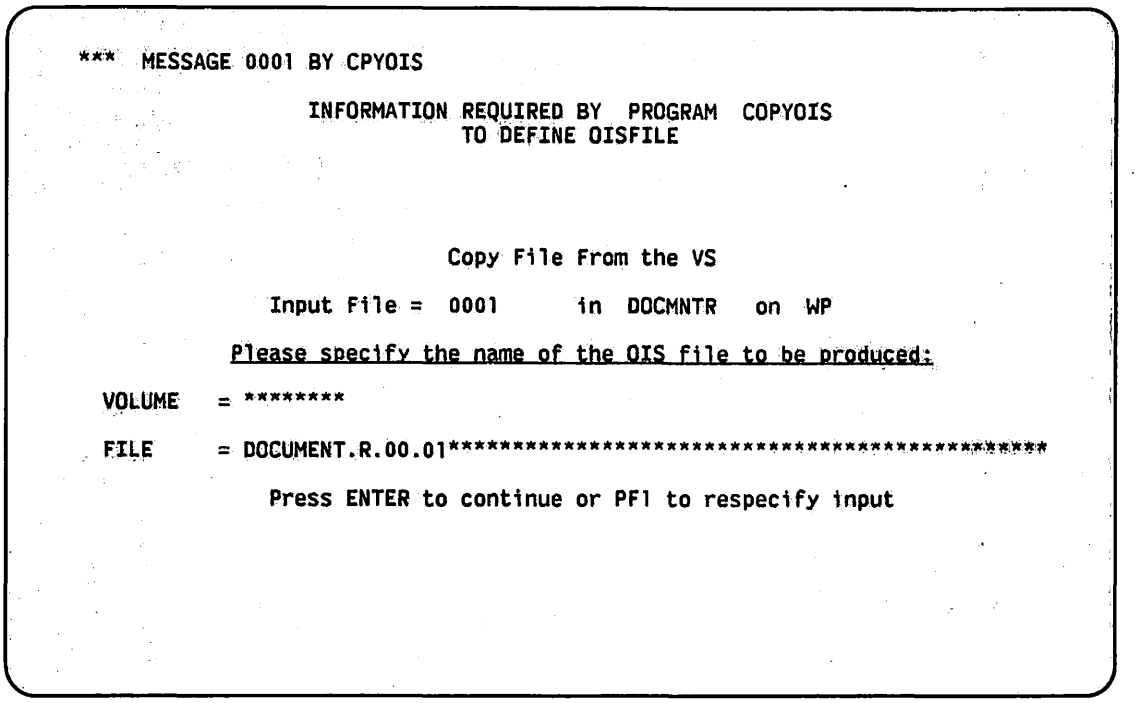


Figure 4-5. Sample COPYOIS Output File Definition Screen

The fields for the COPYOIS Output File Definition screen (Figure 4-5) are described as follows:

Field	Description
VOLUME	Specify the name of the OIS volume on which the copy of the OIS file is to reside.
FILE	Give an OIS file name to the copy of the OIS file. A default value is generated automatically, using the VS library name as the first node and the file name as the second node (e.g., LIBRARY.FILE). However, you can change this value to any valid OIS name. OIS BASIC file names are called file-identifiers. A file-identifier contains a file name, which consists of one to six node names (each containing one to eight letters or digits) concatenated by a period (e.g., BASIC.SRCE.PJK.TEST).

You can respecify the input definitions by pressing PF1 or press ENTER to continue processing. If you have not mounted the diskette, the COPYOIS Mount Volume screen appears and prompts you to mount the diskette (refer to Section 4.3). After the output OIS volume has been mounted, the file is copied to the diskette, and the COPYOIS Function Selection screen (Figure 4-2) appears with a completion message.

4.5.2 Copying Multiple Files

To perform a multiple file copy, specify the library and volume on which the files reside on the Copy Files from VS Input Definition screen (Figure 4-4) and press PF9. The COPYOIS Multiple File Copy screen (Figure 4-6) appears with a menu that lists all the names of the files from the library that you specified.

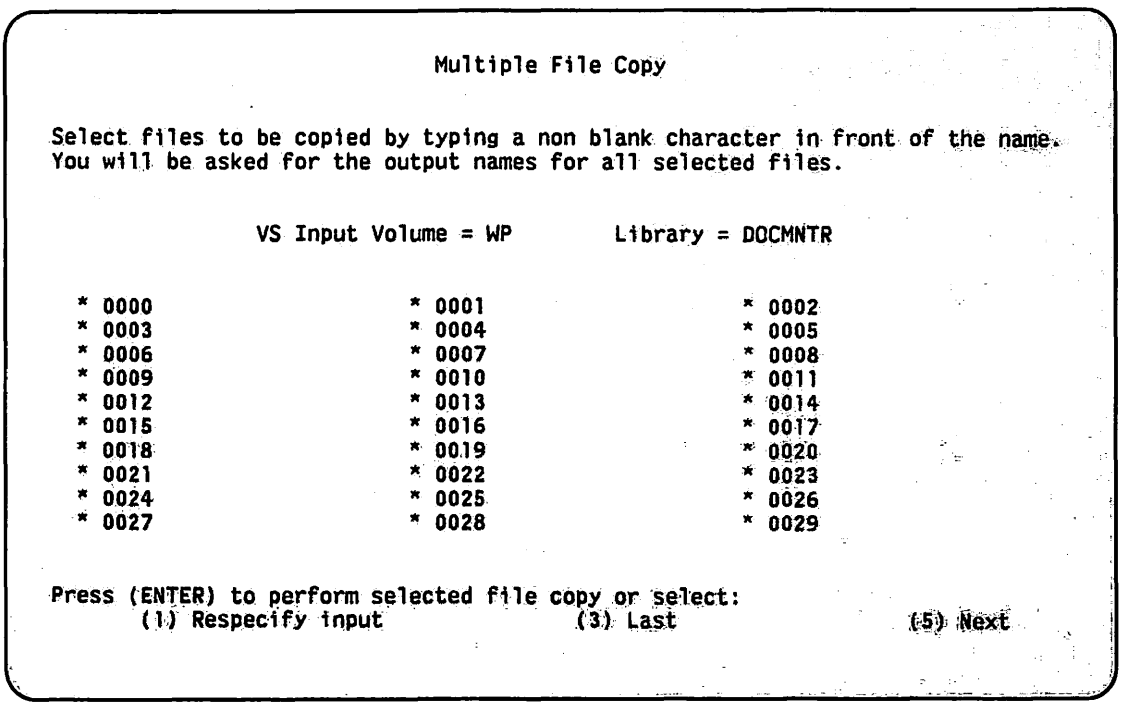


Figure 4-6. Sample COPYOIS Multiple File Copy Screen

Select the files that you want to copy by typing a nonblank character in the block graphic next to each file name. You can respecify the input on the Copy Files from VS Input Definitions screen by pressing PF1. When you have selected all the files to be copied, press ENTER to continue processing.

To enable you to change the default file name, COPYOIS displays one COPYOIS Output File Definition screen (Figure 4-5) for each file that you selected. If you want to change the file name, simply type over the default file name and press ENTER. Otherwise, accept the default file name and press ENTER to continue. When you specify the name of the last file to be copied, the files are then copied to the diskette, after which the COPYOIS Function Selection screen (Figure 4-2) appears with a completion message.

4.6 COPY/CONVERT OIS FILE (PF4)

COPYOIS enables you to copy and/or to convert single or multiple OIS source and data files to the VS by pressing PF4, the Copy/Convert OIS File function key, from the COPYOIS Function Selection screen. The OIS files can reside either on an OIS diskette or on a VS volume. Note that even if the file resides on the VS, you must enter the file name in OIS format (e.g., LIBRARY.FILE).

You have the following capabilities when copying or converting OIS files to VS:

- You can copy and convert the files in a single operation if the files reside on an OIS diskette.
- You can copy OIS files to another VS volume or convert them to the VS file format, if the OIS files already reside on a VS volume.

- You can transfer the OIS files to the VS for conversion at a later date.
- The OIS file may be a list processing data file, or an OIS BASIC source or data file. (OIS files that already reside on the VS have been previously transferred to the VS by COPYOIS, a WSN file transfer, or TCCOPY.)

When PF4 is pressed from the COPYOIS Function Selection screen, the Copy/Convert OIS File Input Definition screen (Figure 4-7) appears. To return to the COPYOIS Function Selection screen, press PF1.

```
***  MESSAGE 0001 BY CPYOIS

                INFORMATION REQUIRED BY  PROGRAM  COPYOIS
                            TO DEFINE INPUT


      Please specify the Volume and File names to copy/convert one file
                  or the Volume name for multiple file copy/convert.

                  Please provide the input file information:

   VOLUME   = WP******
   FILE     = ************************************************************
   DEVICE   = VS*                          (OIS or VS Volume)

     Please specify whether a BASIC file conversion is to be performed:

                     CONVERT  = YES         (YES or NO)

               Press (ENTER) to copy/convert the specified file
                     (1)   to terminate this operation
                     (9)   for multiple file copy/convert
```

Figure 4-7. Sample Copy/Convert OIS File Input Definition Screen

The fields for the Copy/Convert OIS File Input Definition screen (Figure 4-7) are described as follows:

Field	Description
VOLUME	Specify the volume name on which the file(s) (that you want to copy or convert) reside.
FILE	Specify the OIS name of the file that you want to copy or convert. (This field can be omitted for a multiple file copy or conversion from an OIS diskette.) For multiple file operations, you may specify a partial file name. If you specify a partial file name, COPYOIS displays for selection only those files whose names begin with the partial file name.
DEVICE	Specify either OIS or VS as the device on which the files will reside. The Device field value defaults to VS.
CONVERT	Specify whether or not a BASIC file is to be converted. Enter YES to convert the OIS file to a VS file. Enter NO, and the file is copied, but not converted. **NOTE** *Specify CONVERT = NO when copying OIS list processing files to the VS.*

If you want to copy or convert multiple files, press PF9 and refer to Section 4.6.2. Otherwise, specify the values for the fields and press ENTER. If you specify an unmounted volume, COPYOIS prompts you to mount the volume (refer to Section 4.3).

NOTE

Copying multiple OIS files that reside on the VS require a library name and a device. The library is the largest classification of data that can be copied from the VS.

Copying multiple OIS files that reside on diskette require a volume name and a device. The volume is the largest classification of data that can be copied from the OIS.

4.6.1 Copying or Converting a Single File

COPYOIS can transfer a single file with or without conversion when you identify the file on the Copy/Convert OIS File Input Definition screen and press ENTER. You must specify conversion options to convert the input OIS file (BASIC data file or keyed file). An exception is BASIC source files which do not have conversion options; identify only the output VS file. Refer to Section 4.6.3 for a description of the type of conversion performed on an OIS BASIC source file.

If you are converting a BASIC data file, the COPYOIS Conversion Options screen (Figure 4-8) requests information before you define the VS output file. Field descriptions are given, following Figure 4-8. To access instructions about specific fields during processing, press PF13.

```
Press (ENTER) to perform selected file copy or select:
      (1) Respecify input          (3) Last                    (5) Next

***  MESSAGE 0001 BY CPYOIS

                INFORMATION REQUIRED BY  PROGRAM  COPYOIS
                           TO DEFINE OPTIONS


  Input Volume = WP
  Input File   = DOCMNTR.0001

 Please supply the required conversion information for the BASIC data file:

  FORMAT    = FIXED***    (File Type: FIXED, VARIABLE or KEYED)
  LENGTH    = ****        (Data Record Length)
  NUMERICS  = ********    (Output Numeric Format: INTEGER, FLOATING, PACKED)
  DECIMALS  = **          (Decimal alignment for PACKED Numeric Format Only)
  KEYFIELD  = ***         (Key Field Embedded in Record: YES or NO)
  KEYSTART  = ****        (Key Field Start Position)

                Press (ENTER) to continue file conversion
                      (1)   to respecify input
                     (13)   for instructions
```

Figure 4-8. Sample COPYOIS Conversion Options Screen

The fields for the COPYOIS Conversion Options screen (Figure 4-8) are described as follows:

Field	Description
FORMAT	Specify FIXED, VARIABLE, or KEYED as the format of the BASIC data file record. Consecutive files can have either a FIXED or VARIABLE format; indexed files have a KEYED format.
LENGTH	Specify the actual length of the keyed or fixed record, or the maximum length for the variable record length data file. The maximum fixed record length is 2048. The maximum variable record length is 2024. The maximum keyed record length is 2040.
NUMERICS	You do not have to specify numeric conversion if the record does not contain numeric data. Otherwise, specify whether all numerics encountered are to be converted to 4-byte INTEGER, 8-byte FLOATING point, or 8-byte PACKED decimal. All numerics encountered are converted to the same type. If any number is outside the range of the VS numeric representation, that number is converted to zero. Because an OIS numeric occupies nine bytes in the data record (for fixed or keyed files), any trailing spaces are output as zeros in the data record. For variable length records, the exact number of bytes required by the output number is put out in the data record (i.e., 4 or 8 bytes).
DECIMALS	Specify the number of decimal positions for the output packed number.
KEYFIELD	If the data file is keyed, specify whether the key is embedded in the data record (YES or NO).
KEYSTART	If the key is embedded, specify the starting location of the key field in the data record. If the key is not embedded, specify the position in the data record where the key is to be inserted. Also, the record size is increased automatically to account for the addition of the key field in the output record.

If the key file allows duplicates for keyed file conversions, the occurrence (OCC) number (converted to character format of length 5) is appended to the key field. The record size and key field length is increased automatically to account for the addition of the OCC number in the output data record.

When you have completed the conversion option specifications for data and keyed files, COPYOIS requests the output VS file through the Volume, File, and Device fields on a screen that is identical to the Copy/Convert OIS File Input Definition screen (except that the Convert field is not included. Refer to Figure 4-7.). If the file that you are converting is a keyed file, you must provide the keyed data file location. When copy/conversion is completed, the COPYOIS Function Selection screen appears with a completion message. The COPYOIS Function Selection screen adds PF12, Display File Conversion Listing, to the function menu upon completion of a conversion.

A conversion statistics report is generated for all data files converted to VS format. You can display or print the report by pressing PF12 from the COPYOIS Function Selection screen. The statistics report summarizes the options that you requested for data file conversion. The report contains the number of records in the file, the maximum or the actual record size for the records in the data file, the type of output file generated, and the type of numeric data conversion performed.

4.6.2 Copying or Converting Multiple Files

If you want to perform a multiple file operation, press PF9 from the Copy/Convert OIS File Input Definition screen (Figure 4-7). The next screen displayed is the COPYOIS Multiple File Copy/Convert screen (Figure 4-9).

```
                        Multiple File Copy / Convert

Select files to be copied by typing a non blank character in front of the name.
If the input file is a BASIC source or data file, specify whether it is to be
converted.  You will be asked for the output names for all selected files.

                VS Input Volume = WP        Library = DOCMNTR

              Convert                 Convert                 Convert
  * 0000        YES     * 0001        YES     * 0002          YES
  * 0003        YES     * 0004        YES     * 0005          YES
  * 0006        YES     * 0007        YES     * 0008          YES
  * 0009        YES     * 0010        YES     * 0011          YES
  * 0012        YES     * 0013        YES     * 0014          YES
  * 0015        YES     * 0016        YES     * 0017          YES
  * 0018        YES     * 0019        YES     * 0020          YES
  * 0021        YES     * 0022        YES     * 0023          YES
  * 0024        YES     * 0025        YES     * 0026          YES
  * 0027        YES     * 0028        YES     * 0029          YES
```

Figure 4-9. Sample COPYOIS Multiple File Copy/Convert Screen

The COPYOIS Multiple File Copy/Convert screen is similar to the COPYOIS Multiple File Copy screen (refer to Section 4.5.2), except that it requires a YES or NO value in the Convert field for each file that you want to copy or convert.

When copying or converting from the OIS volume (if you do not specify a file node in the File field of the COPYOIS Function Selection screen, Figure 4-2), files on the volume are displayed. If you enter a file node name in the File field on the Copy/Convert OIS File Input Definition screen, all files whose node names begin with the file node entered are displayed for selection.

When the COPYOIS Multiple File Copy/Convert screen appears, select the files that you want to copy by typing a nonblank character in the space next to each file name. Enter YES or NO in the Convert space. The default conversion option appears using the value that you supplied in the Copy/Convert OIS File Input Definition screen. You can respecify the input on the Copy/Convert OIS File Input Definition screen (Figure 4-7) by pressing PF1. To perform the selected file copy, press ENTER.

When you convert a BASIC data file, the COPYOIS Conversion Options screen (Figure 4-8) appears. If the file is keyed, you must also specify the keyed data file location, as described in Section 4.6.1.

To enable you to change the default file name, COPYOIS displays one COPYOIS Output File Definition screen (Figure 4-5) for each file that you selected. To change the file name, simply type over the default file name and press ENTER. Otherwise, accept the default file name and press ENTER to continue. When you enter the name of the last file to be copied and the files are copied to the diskette, the COPYOIS Function Selection screen appears with a completion message. When copy/conversion is completed, the COPYOIS Function Selection screen appears with a completion message. The COPYOIS Function Selection screen adds PF12, Display File Conversion Listing, to the function menu upon the completion of a conversion.

A conversion statistics report is generated for all data files that were converted to VS format. You can display or print the report by pressing PF12 from the COPYOIS Function Selection screen. The statistics report summarizes the options that you requested for data file conversion. The report contains the number of records in the file, the maximum or the actual record size for the records in the data file, the type of output file generated, and the type of numeric data conversion performed.

4.6.3 Source File Conversion

When you enter a source file that you want to transfer and/or copy to the VS, you should specify the conversion option with the copy. COPYOIS checks the input file to verify that the file is actually an unprotected, OIS BASIC source file. If the file cannot be converted to a VS BASIC source program, it is assumed to be an OIS BASIC data file, and you are prompted to enter the fields of the the data files structure.

COPYOIS performs the conversion as follows when the output source program is generated:

- All file SELECT statements are placed before any executable code.
- Lowercase letters are converted to uppercase letters which is required by VS BASIC (except alphanumeric-literal-string constants and comment lines).
- Output statements take any VS BASIC spacing requirements into account.
- The original statement indentation is preserved.
- Statements incompatible with VS BASIC are preceded by an asterisk (*) to convert the statement to a comment.
- Messages generated during conversion are flagged as comments by an asterisk (*) to convert the statements to comments.
- All compound statements in an OIS source line are generated as separate source statement lines.

COPYOIS separates OIS BASIC source lines longer than 66 characters (excluding the line number) in the following way:

- If multiple statements follow the OIS BASIC ERROR statement on the same line, all the statements that follow ERROR on the current input line are converted to comment statements, denoted by the asterisk (*).
- Variable names and BASIC keywords are not separated from one line to the next.
- If the line being separated is a Remark (REM) statement, a second REM statement is generated to contain the remaining text.
- If the characters that you want to separate are actually part of an alphanumeric-literal-string, the string continues to Line 71. An exclamation symbol (!) follows in Column 72, and the remainder of the string starts in Column 7 of the next line.

For example, consider the following OIS BASIC input statement:

```
0200    If str(a$,2,1) = "*" then input"Please specify
        the number of items to be included in the list ?
        ",s:if s=0 then print "Zero items cannot be
        specified":goto200
```

COPYOIS converts this input statement to the following VS BASIC lines:

```
000200    IF STR(A$,2,1) = "*" THEN INPUT "Please specify
          the number of!
000202    items to be included in the list ? ",S
000204    IF S = 0 THEN PRINT "Zero items cannot be
          specified"
000206    GOTO 200
```

NOTE

Since incompatible BASIC statements are converted to comments, you should not renumber the VS BASIC source file because line number references can be contained in a commented line. If this occurs, you may have difficulties trying to track the line reference after renumbering.

All numeric and alphanumeric expressions are examined for incompatible functions (e.g., OCC\# or ERRNO). If they occur in the expression, the statement is converted to a comment (*) statement. In addition, the expressions are examined for functions that must be converted to VS BASIC statements (i.e., CONVERT). The expression is translated so that its execution in VS BASIC is as close as possible to its execution in OIS BASIC. Refer to the *VS BASIC Language Reference* and the *OIS BASIC Language Reference* for more information about the BASIC language.

4.7 CREATE DISKETTE CATALOG LISTING (PF5)

COPYOIS enables you to create a catalog listing of your OIS files on the OIS diskette by pressing PF5 from the COPYOIS Function Selection screen. When you select this option, specify the name of the OIS input volume for which you want a catalog listing.

If you specify the name of an unmounted diskette, you can mount it as described in Section 4.3. When COPYOIS has completed the listing, the COPYOIS Function Selection screen appears with a completion message. You can view the listing by pressing PF14 from the COPYOIS Function Selection screen.

4.8 RENAME FILE ON DISKETTE (PF7)

To rename a file or files on a diskette, press PF7 from the COPYOIS Function Selection screen. The resulting screen requests the following information:

Field	Description
VOLUME	Specify the volume that has the file(s) that are to be renamed.
FILE	Specify the file(s) that you want to rename.

Specify the file that you want to rename and mount the diskette (if it is not already mounted). COPYOIS then requests you to enter a new name for the file that you selected. After you mount the diskette, the file is renamed and the COPYOIS Function Selection screen appears.

4.9 DELETE FILE FROM DISKETTE (PF8)

To delete an OIS file from the OIS diskette, press PF8 from the COPYOIS Function Selection screen. The resulting screen requests the following information:

Field	Description
VOLUME	Specify the volume that has the file(s) to be deleted.
FILE	Specify the file(s) that you want to delete.

Specify the file that you want to delete and mount the diskette (if it is not already mounted). After you mount the diskette, the file is deleted and the COPYOIS Function Selection screen appears.

4.10 ASSIGN DISKETTE VOLUME/FILE PASSWORD (PF9)

To assign a password to a volume or a file on a diskette, press PF9 from the COPYOIS Function Selection screen. The resulting screen requests the following information:

Field	Description
VOLUME	Specify the OIS volume to which the password is to be assigned.
NAME	Specify the name level to which the password is assigned.
PASSWORD	Specify a password to the volume or file node.

OIS names consist of up to six nodes. These nodes within an OIS name are given a priority in order of their hierarchy. A possible OIS name could be VOL2.LIBSET6.ACCOUNTS. RECEIVE.CUSTOMER.CUST0043. VOL2 is the volume, LIBSET6 is a subset of the volume, ACCOUNTS is a subset of LIBSET6, and so on in the hierarchy. The Name field enables you to specify at which level of the hierarchy the password is to be assigned.

For example, if VOL2.LIBSET.ACCOUNTS were specified for the Name field, every level from the ACCOUNTS node down to the actual file node level would be password protected; that is, if access was desired at any level at or below the ACCOUNTS node level, the correct password would have to be supplied first. If the node level already has a password, you must first identify the old password before assigning a new password.

If you specify the name of an unmounted diskette, you can mount it as described in Section 4.3. When you mount the diskette, assign (or reassign) the password. After you assign the password, the COPYOIS Function Selection screen appears.

4.11 A SAMPLE COPYOIS PROCEDURE

You can control COPYOIS processing through the VS Procedure language. A complete list of COPYOIS GETPARMs is provided in Appendix A. Refer to the *VS Procedure Language Reference* for complete details about procedure syntax.

The following procedure mounts an OIS diskette, copies an OIS data file from the diskette, and then converts it to a VS file. Once the file is copied, the procedure displays the statistics report and dismounts the diskette. VS Procedure language automatically converts lowercase to uppercase. For this reason, the colon (:), the equals sign (=), the period (.), any OIS volume names, and file names that have lowercase letters must be enclosed in quotes. Any names that contain a period or a colon in the string must be enclosed in quotes.

```
PROCEDURE
    RUN COPYOIS
        ENTER MENU 4
        ENTER INPUT VOLUME=OISVOL, FILE="Keyed.Reclen.409",
            DEVICE=OIS
        ENTER MOUNT VOLUME=VOL, UNIT=023
        ENTER OPTIONS FORMAT=KEYED, LENGTH=409,
            NUMERICS=INTEGER,
            DECIMALS=   , KEYFIELD=YES, KEYSTART=1
        ENTER DATAFILE VOLUME=OISVOL, FILE="pjadata.datakl1",
            DEVICE=OIS
        ENTER OUTPUT FILE=OISKEYED, LIBRARY=PJADATA,
            VOLUME=ZENITH
        ENTER MENU 12
        ENTER MENU 16
    DISMOUNT DISK VOL
RETURN
```

CHAPTER 5
THE COPY2200 UTILITY

5.1 INTRODUCTION

The COPY2200 utility enables you to copy information between the VS and the Wang 2200 Series systems. COPY2200 transfers data files to and from 2200 diskettes and converts the files to and from the 2200 file formats.

All 2200 diskettes must be mounted on the VS as nonlabeled (NL) diskettes. The VS can access the information on the following types of diskettes:

- 2200 8-inch, single-sided, single-density (SSSD), hard-sectored diskettes (Wang White Label) used by the 2200VP, 2200LVP, and 2200MVP systems
- 2200 8-inch, double-sided, double-density (DSDD), soft-sectored diskettes (Wang Red Label) used by the 2200LVP and 2200SVP systems

NOTE
To access a 2200 soft-sectored diskette, COPY2200 requires a diskette drive that supports soft-sectored diskettes.

COPY2200 can generate and access 2200 and VS nonlabeled (NL) image files. An image file is a byte-by-byte copy of an entire diskette or disk and is described in Section 5.2.

The following list provides a quick reference to the specific subjects that are explained in this chapter. Turn to the corresponding section for the necessary information.

Section	Topic
5.2	Image files
5.3	Function selection
5.4	Create VS files from 2200 files residing on a 2200 diskette or VS image file.
5.5	Convert VS files to the 2200 format and then transfer the files to a 2200 diskette or image file.
5.6	Create a VS disk-image file from the contents of 2200 diskettes.
5.7	Generate a 2200 diskette by copying a VS disk-image file from the VS to a diskette.
5.8	COPY2200 processing through the VS Procedure language.

Figure 5-1 shows an overview of COPY2200 processing.

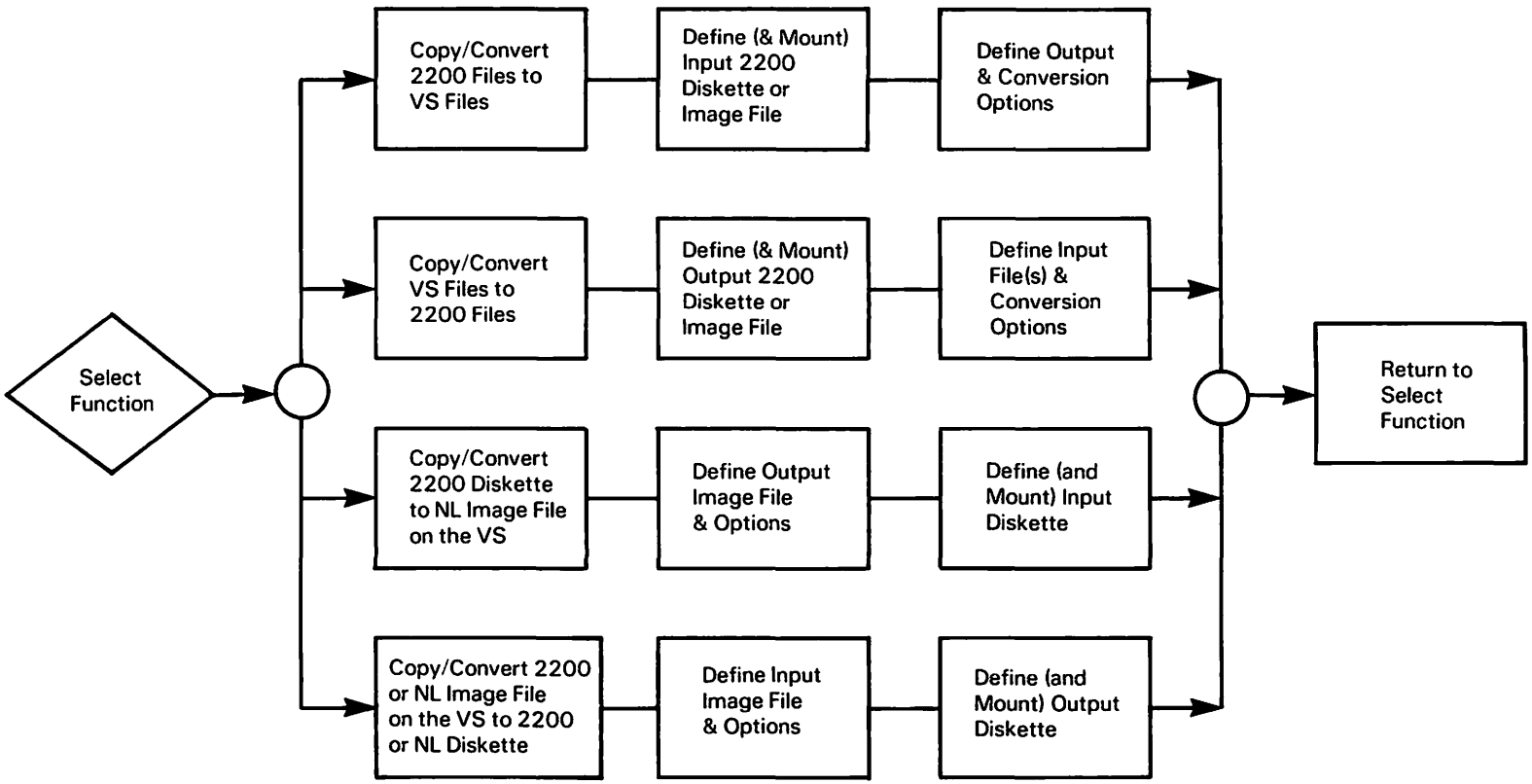


Figure 5-1. COPY2200 Processing

5.2 IMAGE FILES

An image file is an exact duplicate of a file (or group of files) on a disk or diskette(s) except that the image file is segmented into consecutive, 256-byte records. Diskette sectors have a length of 256 bytes, therefore the segmentation of an image file has a practical purpose. Each 256-byte record of an image file represents one diskette sector.

You can use image files to make copies or store the contents of 2200 or NL diskettes. You can also use image files to temporarily save diskette contents until they are converted into VS format (which frees the diskette drive) and to store converted VS files until the diskette drive is available. You can create an image file in two ways:

1. You can copy the entire contents of a diskette (or multiple diskettes, which can include many diskette files) into a single image file, without reformatting the contents.
2. You can convert the contents of a VS file into the diskette format and then copy the contents into an image file.

5.3 SELECTING A FUNCTION

When you run the COPY2200 utility, the COPY2200 Function Selection screen (Figure 5-2) is the first screen that is displayed. Select a function by pressing the corresponding program function (PF) key. The following sections describe the functions in detail.

```
***  MESSAGE CP01 BY COPY22

                    RESPONSE REQUIRED BY  PROGRAM  COPY2200
                            TO SELECT ACTION



              Please select, via PF key, the desired function

PFKEY   FUNCTION                                  PFKEY   FUNCTION

 (1)    Create VS files from a 2200  ------    (2)    Create a 2200 diskette
        diskette (or from a VS      * 2200 *           (or disk-image file)
        disk-image file)             ------            from VS files

 (9)    Create a VS disk-image       -------   (10)    Create diskette(s) from
        file from diskette(s)       * image *          a VS disk-image file
                                     -------

                        Press PF16 to end the program
          <<<<<< WANG VS Intersystem copy program - Version x.xx.xx >>>>>>
```

Figure 5-2. COPY2200 Function Selection Screen

5.4 CREATE VS FILES FROM A 2200 DISKETTE (PF1)

You can create VS files from program files, from certain types of data files, or from VS disk-image files with this function. The files can either reside on 2200 diskettes or in VS image files.

If you press PF1 from the COPY2200 Function Selection screen, the Create VS Files Input Definition screen is displayed (Figure 5-3). This screen requires the input diskette name or the disk-image file name. It also requires other fields which are described following Figure 5-3. You can return to the COPY2200 Function Selection screen from the Create VS Files Input Definition screen by pressing PF1 again.

If you have not already done so, you are prompted to mount the input 2200 diskette (or the volume on which the image file is mounted, i.e., image volume). If you mount the 2200 diskette through the command processor rather than through the COPY2200 utility, you must mount the diskette as an NL diskette.

```
***  MESSAGE 2V01 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                          TO DEFINE INPUT



          Please specify the input diskette or disk-image file
       ------------------------------------------------------------
       *           ENTER the name of the 2200 (NL) diskette         *
       *                    DISKETTE = ******                       *
       ---------------------------- OR ----------------------------
       *           ENTER the location of a VS disk-image file       *
       * FILE     = ********   LIBRARY  = DOCMNTR*   VOLUME   = WP**** *
       *                (consecutive, 256-byte records)             *
       ------------------------------------------------------------

     Please specify correction characters for unusable 2200 file names
   STARTER  = V  (precedes illegal starting character; may be A-Z,0-9,$,@)
   FILLER   = #  (replaces illegal embedded character; may be A-Z,0-9,#,$,@)

        Press PF4 to mount the desired volume on DEVICE#  = ***
                  Press PF1 to return to the copy menu
```

Figure 5-3. Sample Create VS Files Input Definition Screen

You must specify either the Diskette field or the File, Library, and Volume fields. You cannot specify both (or neither). A description of the fields on this screen is as follows:

Field	Description
DISKETTE	Specify the name of the input 2200 diskette. The input source must be a 2200 diskette if it is not a VS image file.
FILE, LIBRARY, VOLUME	Specify the name of the 2200 disk or image file. The input source must be the name of the file, library, and volume of the VS image file if it is not a 2200 diskette.

Field	Description
STARTER	The value in the Starter field is added to the front of any 2200 file name that begins with a VS-invalid character. Starter values have a range of A to Z, 0 to 9, $, and @. The default value for Starter is V.
FILLER	The value in the Filler field replaces all VS-invalid characters except the first (Starter) within the 2200 file name. Filler values have a range of A to Z, 0 to 9, $, @, and #. The default value for Filler is #.
DEVICE#	If you specify a value for Device# and press PF4, COPY2200 allows you to mount the input volume. You can omit Device# if the input volume is already mounted. COPY2200 automatically mounts the diskette as a NL volume.

Starter and Filler values must be entered for both diskette and image file conversion because of the differences in 2200-valid and VS-valid file names. File naming on the VS is more restrictive than on the 2200. The VS allows only alphabetic, numeric, and certain special characters ($, @, and #) for VS file names. File names on 2200 Series systems can consist of any combination of alphanumeric characters utilizing all alphabetic, numeric, and special characters. Therefore, valid 2200 file name characters, that are invalid on the VS, must be converted into valid VS format values. After you define the input, the Copy Mode screen (Figure 5-4) prompts you to select a copy mode option:

```
***  MESSAGE 2V02 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                           TO DEFINE COPYMODE



Please select the desired 2200-to-VS copy mode:
                                MODE     = *
   Options:
             A = Copy all 2200 files to VS (data and program files)
             D = Copy all 2200 data files to VS (no program files)
             P = Copy all 2200 program files to VS (no data files)
             L = Specify a list of files (any type) to be copied


                  Press ENTER after making your selection
                                  - or -
              Press PF1 to return to the input selection screen
```

Figure 5-4. Copy Mode Screen

NOTES

All files specified through any copy mode function are converted into VS format.

BASIC-2 program files can be copied, but BASIC-2 syntax cannot be correctly converted or flagged.

If you specify L from the Copy Mode screen, a listing of all files on the diskette or the image file are displayed. Data files and program files are as indicated on the screen. Select the individual files that you want copied and converted from the list and press ENTER.

There are two output definition screens that request information; one for data files and one for program files.

- If you select option A (All files), both screens request you to supply the necessary information.
- If you select option D (Data files), only the Data Files Output screen requests you to supply the necessary information.
- If you select option P (Program files), only the Program Files Output screen requests you to supply the necessary information.
- If you select option L (List of files), one screen or both screens appear depending on the files that you want to copy.

If you select option A, the Data Files Output screen is displayed before the Program Files Output screen. The following sections discuss the fields that appear on the output screens.

5.4.1 The Data Files Output Screen

The Data Files Output screen (Figure 5-5) requests you to define the VS output data file. Use Table 5-1 as a quick reference to the valid values of the fields that are used in the Data Files Output screen. Each field is discussed in detail following the table.

```
***  MESSAGE 2V04 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                             TO DEFINE DATAOUT



Please specify output parameters for data files to be copied:
            LIBRARY  = ********   VOLUME   = WP****   FILECLAS = *
Please specify output data file organization:
                       TYPE     = *    COMPRESS = YES
   Options:
     F = Consecutive fixed-length        >        For types F,V,T,X:
     V = Consecutive variable-length     > >        RECSIZE  = ****
     T = Standard TC copy                > >      For types T and X:
     X = Extended TC copy                >          TRANSL   = NO*

     E = Editor source copy (80 bytes) - Input is in 2200/VS Editor format
     S = Sector copy (256 bytes) -  Copy all sectors of the file
     N = None - specify separately for each file

                  Press ENTER after making your selection
            Press PF14 for a description of the file type options
```

Figure 5-5. Sample Data Files Output Screen

Table 5-1. Field Ranges for Data Files Output

Field	Range
LIBRARY	Any valid VS library name
VOLUME	Any valid VS volume name
FILECLAS	(blank), $, @, #, or A through Z
TYPE	F, V, T, X, E, S, or N
RECSIZE	If F, V, T, or X is specified for Type, enter valid numeric values only
TRANSL	If T is set for Type, enter a value (YES or NO) in the Transl field.
COMPRESS	YES or NO

The following is a description of the fields for the Data Files Output screen (Figure 5-5):

Field — **Description**

LIBRARY, VOLUME — Specify the destination of the output files. The file name of the output file is the same as the 2200 file, modified as necessary by the previously defined Starter and Filler values.

FILECLAS — Specify the file class of the output files. The default value is your workstation default value (which is assigned through the SET Usage Constants option (PF2) of the Command Processor menu or through the VS Procedure language SET statement).

TYPE — Specify the type of conversion to be performed. Seven conversion types are available:

- F *Fixed-length* — Create an output data file with consecutive, fixed-length records. You must specify the Recsize value for this type.
- V *Variable-length* — Create an output data file with consecutive, variable-length records. You must specify the Recsize value for this type.
- T *Standard Telecommunications (TC) Copy* — Create a VS output data file from an input 2200 standard telecommunications format data file (4 by 62 logical record size). One VS record is generated from each TC record. You must specify the Recsize value and the Transl values for this type.
- X *Extended Telecommunications Copy* — Create an output data file from an input 2200 extended TC format data file (4 by 62 logical record size). One VS record is generated from one or more TC records with or without a header. You must specify the Recsize value and the Transl values for this type.
- E *Source Copy (2200/VS EDITOR)* — Create an output source file from an input 2200/VS EDITOR format, as created by COPY2200 (on the VS through image files) or by the 2200/VS Source Editor (a user aid on the 2200).

Field	Description
TYPE (cont.)	S *Sector Copy* — Copies ali sectors of the files exactly as catalogued, from first to last, including any end-of-file (EOF) and/or control records. You are responsible for ensuring that the files are in VS format. N *None* — No conversion type for all files. You must specify the conversion type separately for each file.
RECSIZE	The Recsize field indicates the size of the records in the files being copied. You must give a value to Recsize if the values F, V, T, or X are specified for Type. Types F and V produce multiple output records if an input record is too long. All four Type options pad files with blanks or truncate characters, where required. The valid values of Recsize for each file organization type are as follows: F *Fixed-length* — Any valid VS fixed-length record size or 0. If the record size is 0, Recsize is assumed to be that of the first record in the 2200 file. V *Variable-length* — The maximum VS record size for the variable-length file. The size must be a legal nonzero value. T *Telecommunications* — A valid VS record size to be created from each TC record or 0. If the record size is 0, the output file is variable length, with a maximum length of 192 bytes. X Either a valid VS record size or 0. If you specify a nonzero record size, each output record is created from as many TC records as needed, assuming that each record determines an 80-byte piece of the logical (output) record. If the record size is 0, the first TC record is assumed to be the 2-byte binary Recsize; the output records are then constructed as described previously.
TRANSL	Translates 2200 information from EBCDIC into ASCII during the copy. You must specify Transl if T is specified for the Type field.
COMPRESS	The Compress field determines whether the output files are to be compressed. The default value, YES, compresses the data. If you enter the value NO next to Compress, the data is not compressed.

If you specify the values F, V, or N for Type, the Numeric Data Format screen (Figure 5-6) follows the Data Files Output screen.

```
***  MESSAGE 2V07 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                           TO DEFINE NUMERICS



Please specify output format for 2200 numeric values:

                                  TYPE     = *

          Options:
            D = 2200 (decimal) floating point
            F = VS (hex) floating point
            B = VS (fullword) binary
            H = VS (halfword) binary
            P = VS packed decimal - specify parameters:
                   Field length in bytes:  LENGTH   = ** (1-16 allowed)
                   Implied decimal places: PLACES   = ** (0-99 allowed)
            N = None - specify separately for each file

                   Press ENTER after making your selection
```

Figure 5-6. Numeric Data Format Screen

The Numeric Data Format screen prompts you for the output format through the following fields for the 2200 numeric values.

Field	Description
TYPE	Specify the output format of the 2200 numeric values for all the specified data files. The output format consists of the following types: D — *2200 decimal floating-point* (no change in format) F — *VS hexadecimal floating-point* *B — VS binary (fullword)* *H — VS binary (halfword)* *P — VS packed decimal* — You must specify the Length and Places fields. N — *None* — Indicate the output format separately for each file. You need only specify the format if the file contains numeric data.
LENGTH	Field length in bytes for packed decimal values. Lengths of 1 through 16 are allowed.
PLACES	Implied decimal places for packed decimal values. Responses of 0 through 99 are allowed.

After you specify all fields, the utility copies the file(s). The COPY2200 Function Selection screen is displayed when the copying is complete. You can then dismount the 2200 diskette by pressing PF14.

5.4.2 The Program Files Output Screen

On the Program Files Output screen, the fields are similar to those required on the Data Files Output screen. You must supply the following information:

Field	Description
LIBRARY, VOLUME	Specify the destination of the output files. You must specify the library and volume names.
COMPRESS	Specify whether the output files are to be compressed. The default value, YES, compresses the data. If you enter NO next to Compress, the data is not compressed.
FILECLAS	Specify the file class of the output files to be created by the copy procedure. The default value is obtained from your default file protection class, which is set through the SET Usage Constants option (PF2) of the Command Processor menu or through the VS Procedure language SET statement).
CONVERT	Specify whether the 2200 BASIC syntax is converted to VS BASIC syntax during the copy procedure. The default value (YES) performs the conversion. A NO value leaves the file in the input 2200 BASIC or BASIC-2 syntax.

After you specify the fields and press ENTER, the utility copies the file(s). The COPY2200 Function Selection screen (Figure 5-2) is displayed when the function is completed.

5.5 CREATE A 2200 DISKETTE (OR DISK-IMAGE FILE) FROM VS FILES (PF2)

You can create Wang 2200-format diskettes or image files from certain types of VS files if you press PF2 from the COPY2200 Function Selection screen. The resulting 2200 Diskette Output Definition screen (Figure 5-7) prompts you for information and then enables you to mount the output 2200 diskette, if necessary. You can return to the COPY2200 Function Selection screen by pressing PF1. If you mount the 2200 diskette through the command processor, rather than through the COPY2200 utility, you must mount the diskette as an NL diskette.

NOTE
The 2200 diskette must be previously formatted by a 2200 system before creating the diskette files from the VS files.

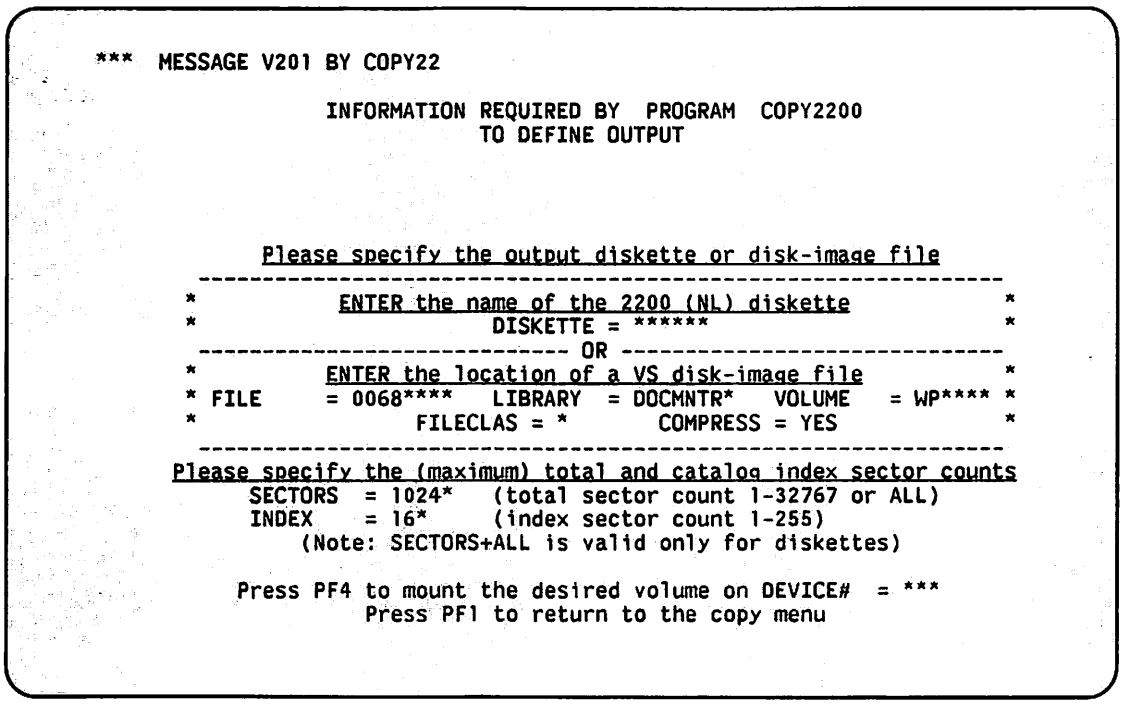


Figure 5-7. Sample 2200 Diskette Output Definition Screen

You must specify either the Diskette field or the File, Library, and Volume fields. You cannot specify both (or none at all).

Field	Description
DISKETTE	Specify the name of a 2200 diskette as the destination of the output information.
FILE, LIBRARY, VOLUME	Specify the file, library, and volume names of the 2200 image file that is being created.
FILECLAS	Specify the file class of the image file to be created by the copy procedure or accept the default value. The default value is obtained from your default file protection class for your workstation, which is set through the SET Usage Constants option (PF2) of the Command Processor menu or through the VS Procedure language SET statement.
COMPRESS	Specify whether the image file is to be compressed. The default value, YES, compresses the file. If you specify NO, the data is not compressed.
SECTORS	Specify the maximum amount of sectors of information to be created on the 2200 diskette or in the image file. Hard-sectored diskettes have a maximum of 1232 sectors. Soft-sectored diskettes have a maximum of 3874 sectors. The Sectors field defaults to 1024. You can specify ALL as the Sectors value instead of using a number for diskette output. This allows the utility to use all of the available sectors on the mounted hard- or soft-sectored disk. If the VS file being copied requires more sectors than 1232 (for hard-sectored diskettes) or 3874 (for soft- sectored diskettes), you must first convert the file into an image file. Image files have a maximum of 32,767 sectors. After conversion, you copy the file to a 2200 disk through multiple diskettes.

Field	Description
INDEX	Specify the number of sectors to be allocated on the output 2200 diskette or diskette image file for the Catalog Index. Except for the first sector (which can contain information for up to 15 files), index sectors contain catalog information for up to 16 files. The default value for the index sector count is 16. The maximum number of index sectors is 255.
DEVICE#	If you have already mounted the output volume, you can omit the device number. Otherwise, specify the device number of the drive on which the output 2200 diskette or image volume is mounted. If you specify a value for Device# and press PF4, you can mount the output diskette or the image volume through COPY2200. COPY2200 automatically mounts a 2200 diskette as a nonlabeled volume.

NOTE

If you use a diskette as an output volume, all data that is currently on the diskette is destroyed by the copy operation. Another screen warns you of this and gives you the option to go ahead with the copy operation or to terminate it. You can continue the copy operation by entering YES in the appropriate field, or terminate the function by entering anything else.

The 2200 Diskette Input Definition screen prompts you to specify the file, library, and volume names of the first file to be copied and press ENTER. It also allows you to mount an input volume by entering the device number in the appropriate field and pressing PF4.

After you define the first input file, the COPY2200 Conversion Type Definition screen (Figure 5-8) requests you to enter the output type for the file being created.

```
***  MESSAGE 0003 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                            TO DEFINE FILETYPE



Please specify output type for consecutive file 0001    :
    (The file contains 000176   fixed-length  records, record size 0256)

              TYPE     = *    FILENAME = 0001****  (2200 file name)
  Options:
    T = Standard TC copy          < RECSIZE  = **** >
    X = Extended TC copy          < TRANSL   = NO*  >

    E = Editor source copy (80 bytes)  - output is in 2200/VS Editor format
    S = Sector copy (256 bytes) - create 2200 file sectors directly


                  Press ENTER after making your selection
            Press PF14 for a description of the file type options
              Press PF1 to return to the file selection menu
```

Figure 5-8. Sample COPY2200 Conversion Type Definition Screen

The output types are described as follows:

Type	Description
T	*Standard Telecommunications Copy* — The VS data file is copied and converted to the 2200 TC format (4 by 62 logical record size). This is the standard format consisting of one VS record for each TC record.
X	*Extended Telecommunications Copy* — The VS data file is copied to a 2200 TC format file (4 by 62 logical record size). This format assumes one or more TC records for each VS record, with or without a header. **NOTE:** *For output type T on the Conversion Type Definition screen, you need to supply a Recsize value. For output type X, the Conversion Type Definition screen requests a Recsize value and a Transl value.*
E	*Source Copy (2200/VS Editor format)* — 2200/VS Editor format files are used with the 2200/VS source Editor (a user aid on the 2200). The input file must have 80-byte records. If a VS BASIC source file is copied, the resulting 2200 file can only be edited by the 2200/VS Editor.
S	*Sector Copy* — The VS data file is copied directly to the physical sectors of the output 2200 file, with no additional trailer or other control information. (Trailer information can include information at the end of the sector.) The input file must have 256-byte records. You must include the 2200 file control information within the 256-byte record.

The fields on the COPY2200 Conversion Type Definition screen (Figure 5-8) are described as follows:

Field	Description
RECSIZE	The Recsize field indicates the size of the records in the files that are being copied and is required for output types T and X. Valid Recsize values are as follows: T — Specify either a valid TC record size (1 to 192) or 0. If the record size is 1 to 192, the output records are padded or truncated to the specified record size. If the record size is 0, the output records are variable-length (1 to 192), with blank-padding and/or truncation of characters beyond Position 192. X — Specify either a valid VS record size (1 to 2048) or 0. If the record size is 1 to 2048 (nonzero), each VS record produces one or more TC records, each of which is an 80-byte piece (with trailing blanks truncated) of the specified record size. If the record size is 0, the output record is constructed as a nonzero record length file, with a record size equal to that of the VS file, and contains an additional 2-byte initial record that contains this record size.
TRANSL	The Transl field translates VS information during the copy operation from ASCII to EBCDIC with a YES value.

After you have entered the necessary information to copy the VS file, press ENTER. The COPY2200 utility enables you to specify additional VS files to be copied or to start copying the file(s) that you have already specified. A COPY2200 Conversion Type Definition screen (Figure 5-8) is supplied for each file that you specify. If you have specified any additional files to copy, press PF16 from the input definition screen to start copying the specified files. Upon completion, the COPY2200 Function Selection screen is displayed.

5.6 CREATE A VS DISK-IMAGE FILE FROM DISKETTE(S) (PF9)

If you press PF9 from the COPY2200 Function Selection screen, you can create VS image files from 2200 diskettes. To return to the COPY2200 Function Selection screen, press PF1.

The Disk-Image File Output Definition screen (Figure 5-9) prompts you for information and then allows you to mount the volume on which the image file is to reside. The output volume is mounted as a VS standard label (SL) volume by COPY2200. If you mount the volume through the Command Processor menu (not the COPY2200 utility) you must mount it as an SL volume.

```
***  MESSAGE DI01 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                           TO DEFINE OUTPUT



                  Please specify output file parameters
          FILE     = 0068****  LIBRARY  = DOCMNTR*  VOLUME   = WP****
                               FILECLAS = *
                               COMPRESS = YES
               Please specify the image type and size parameters
                               TYPE     = T       (T=2200, N=no label)
                               MULTIPLE = NO*     (YES = multiple diskettes)
                               SECTORS  = ALL**
          (This is the total number of diskette sectors to be copied)
           (To copy only those sectors actually used, enter "USED")
                (To copy a single entire diskette, enter "ALL")

                    Press ENTER after specifying parameters
                                    - or -
           Press PF4 to mount the specified volume on DEVICE#  = ***
                    Press PF1 to return to the copy menu
```

Figure 5-9. Sample Disk-Image File Output Definition Screen

The fields for the Disk-Image File Output Definition screen (Figure 5-9) are described as follows:

Field	Description
FILE	Assign a valid, unique file name to the file being created by the COPY2200 operation.
LIBRARY, VOLUME	Specify the library and volume on which the file is to reside.
FILECLAS	Specify the file class of the output file to be created by the copy procedure. The default value is obtained from the default file protection class, which is set through the SET Usage Constants option (PF2) of the Command Processor menu or through the VS Procedure language SET statement.
COMPRESS	The Compress field determines whether the image file is to be compressed. The default value (YES) compresses the data. If you enter NO, the data is not compressed.

Field	Description
TYPE	Specify the type of input diskette. A value of T indicates a 2200-format diskette; N indicates a VS-supported NL diskette.
MULTIPLE	The Multiple field indicates whether multiple diskettes are to be used in the copy. Enter YES or NO.
SECTORS	Specify the total number of sectors to be copied onto the input diskette. You can supply a specific number but the image file cannot exceed the size determined by Sectors. To copy only those sectors actually used by the 2200, you can enter the value USED for the number of sectors. If Multiple = NO, you can specify ALL to copy the entire diskette. The default value is ALL.
DEVICE#	If the output volume is already mounted, you can omit the device number. Otherwise, specify the device number of the drive on which the output volume for the image file is to reside. If you specify a value and press PF4, you can mount the output volume through the COPY2200 utility.

If you mount the 2200 diskette through the Command Processor menu, rather than through the COPY2200 utility, you must mount the diskette as an NL diskette. If you specified Multiple = YES and Sectors = USED on the Disk-Image File Output Definition screen (Figure 5-9), then the COPY2200 Input Diskette Definition screen (Figure 5-10) appears and is repeated until you press PF16 or until you fill up the number of sectors specified in the Sectors value. (Values YES and ALL with the fields on Figure 5-10 are inconsistent.) If Multiple = NO in Figure 5-9, the Sectors field does not appear on the COPY2200 Input Diskette Definition screen (Figure 5-10) and the screen appears only once.

```
***  MESSAGE DI02 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                             TO DEFINE INPUT



  Diskette #1

           Please specify the name of the (next) input diskette
                              DISKETTE = ******

         Please specify the (maximum) number of sectors to be copied
                              SECTORS  = ALL*
                  (Type "ALL" for the maximum sector count)


                    Press ENTER after specifying parameters
                                    - or -
               Press PF4 to mount the diskette on DEVICE#  = ***
              Press PF1 to return to the output selection screen
```

Figure 5-10. COPY2200 Input Diskette Definition Screen

After you have defined the output image file, the COPY2200 Input Diskette Definition screen (Figure 5-10) appears, prompts you for the following information, and then enables you to mount the input 2200 diskette.

Field	Description
DISKETTE	Specify the name of the input diskette volume.
SECTORS	Specify the number of sectors to be copied from the specified diskette if the Multiple field has a YES value. You can specify ALL or a specific number. The default value is ALL.
DEVICE#	If you have already mounted the input volume, you can omit the device number. Otherwise, specify the device number of the drive on which the input 2200 diskette is to be mounted and press PF4. By doing this, you can mount the input diskette through COPY2200. COPY2200 automatically mounts the 2200 diskette as an NL volume.

After all diskettes are copied, the COPY2200 Function Selection screen is displayed. You can dismount the most recently mounted diskette by pressing PF14 from the COPY2200 Function Selection screen.

5.7 CREATE DISKETTE(S) FROM A VS DISK-IMAGE FILE (PF10)

To create 2200 diskettes from VS image files, press PF10 from the COPY2200 Function Selection screen and the Create Diskette(s) from Disk-Image Files Input Definition screen (Figure 5-11) appears. To return to the COPY2200 Function Selection screen, press PF1. If you mount the input volume through the Command Processor menu instead of COPY2200, you must mount the volume as an SL volume.

NOTE
The 2200 diskette must be formatted by a 2200 system before a VS image file is copied to the diskette.

```
***  MESSAGE ID01 BY COPY22

                INFORMATION REQUIRED BY  PROGRAM  COPY2200
                             TO DEFINE INPUT




                      Please specify input file parameters
          FILE     = ********  LIBRARY  = DOCMNTR*  VOLUME   = WP****

              Please specify the image type and size parameters

                                TYPE     = T       (T=2200, N=no label)
                                MULTIPLE = NO*     (YES = multiple diskettes)


                    Press ENTER after specifying parameters
                                    - or -
               Press PF4 to mount the specified volume on DEVICE#  = ***
                         Press PF1 to return to the copy menu
```

Figure 5-11. Sample Create Diskette(s) from Disk-Image Files Input Definition Screen

The Create Diskette(s) from Disk-Image Files Input Definition screen fields are described as follows:

Field	Description
FILE, LIBRARY, VOLUME	Specify the location of the input VS image file.
TYPE	Specify the output diskette type. A value of T indicates a 2200-format diskette; a value of N indicates a VS-supported NL diskette(s).
MULTIPLE	Specify (YES or NO) whether multiple diskettes are to be created.

After you specify the image file, the Create Diskette Output Definition screen prompts you for information and then enables you to mount the diskette. The Create Diskette Output Definition screen is essentially the same as the COPY2200 Input Diskette Definition screen (Figure 5-10), which is discussed in Section 5.6. To return to the Input Definition screen, press PF1. If you mount the diskette through the Command Processor menu instead of the COPY2200 utility, you must mount the diskette as an NL volume. Specify the following field information:

Field	Description
DISKETTE	Specify the name of the output diskette volume.
SECTORS	Specify the number of sectors to be created on the output diskette if Multiple = YES. You can specify a specific number or ALL. The default value for Sectors is ALL.
DEVICE#	If you have already mounted the output diskette, you can omit the device number. Otherwise, specify the device number of the drive on which you mount the output 2200 diskette, and press PF4. By doing this, you can mount the output diskette through COPY2200. COPY2200 automatically mounts the 2200 diskette as an NL volume.

If Multiple has been given a NO value in the Create Diskette(s) from Disk-Image Files Input Definition screen (Figure 5-11), then the Sectors field does not appear on the Diskette Output Definition screen and the output definition screen appears only once. If Multiple has been given a YES value, the Diskette Output Definition screen is repeated until you press PF16 or until you have copied all the records from the diskette image file.

Next, you are prompted to enter the number of sectors to be copied (provided that you gave a YES value to the Multiple field) and the 2200 diskette name. The prompt also informs you that the copy function destroys all information currently stored on the diskette. To continue the copy function, enter YES in the space provided; to terminate it, press any key. After the diskette(s) are created, the COPY2200 Function Selection screen is displayed.

5.8 A SAMPLE COPY2200 PROCEDURE

You can control COPY2200 processing through the VS Procedure language. You can specify all COPY2200 operations through a procedure, including diskette mounting and dismounting. You can embed the mount and dismount operations in the RUN, ENTER (or DISPLAY) sequence, provided that you use the COPY2200 mount and dismount operations instead of the Procedure language MOUNT and DISMOUNT statements. A list of COPY2200 GETPARMs is provided in Appendix A. Consult the *VS Procedure Language Reference* for details about Procedure language syntax.

The following procedure copies all 2200 data files on VL2200 to the variable-length files in the 2200LIB library on the VS SYSTEM volume. Numeric data is converted to VS 4-byte packed decimal data. Invalid characters in 2200 file names are converted to valid VS file names by using a Starter value of M and a Filler value of L. The procedure logically mounts and dismounts the diskette before exiting the utility. You (or the system operator if the procedure is run in Background mode) need only physically mount and dismount the 2200 diskette.

```
PROCEDURE
    RUN COPY2200
        ENTER ACTION 1
        ENTER INPUT 4, DISKETTE=VL2200, STARTER=M, FILLER=L,
            DEVICE#=023
        ENTER COPYMODE MODE=D
        ENTER DATAOUT LIBRARY=2200LIB, VOLUME=SYSTEM, TYPE=V,
            RECSIZE=60
        ENTER NUMERICS TYPE=P, LENGTH=4, PLACES=0
        ENTER ACTION 14
        ENTER ACTION 16
RETURN
```

CHAPTER 6
THE COPYWP UTILITY

6.1 INTRODUCTION

The COPYWP utility converts VS WP documents to VS files and converts VS files to VS word processing documents. This function enables you to transfer information between the VS and Word Processing Systems. In addition, COPYWP performs filing functions for VS word processing documents and libraries.

6.1.1 COPYWP Document Filing Functions

The COPYWP document filing functions enable you to manipulate VS WP documents and libraries in the following ways:

- Copy a document into another document, a library into a document, or a library into another library.
- Delete a single document or all documents in a library.
- Rename a single document or the library in which a set of documents reside.
- Reorganize a document or library to release the unused space in a document or in all documents in a library.
- Merge two input documents into a single output document.

The document filing functions do **not** include an archiving facility. The archiving facility is supported as part of the Document Filing function of VS/IIS. If you use a diskette as an input or an output volume for COPYWP, it must be formatted as a VS SL-diskette using the DISKINIT utility. (Refer to Chapter 7 for infomation about the DISKINIT utility.) COPYWP can only use standard label (SL) VS volumes or diskettes and does **not** support WP-archived diskettes.

6.1.2 COPYWP Document Conversion Functions

The COPYWP file conversion functions enable you to perform conversions in the following ways:

- Convert a VS WP document to a VS data, source, print, or 2780 Telecommunications (TC) file.
- Convert a VS source, print, image, 2780 TC, or consecutive data file to a VS WP document.

Figure 6-1 shows an overview of COPYWP processing.

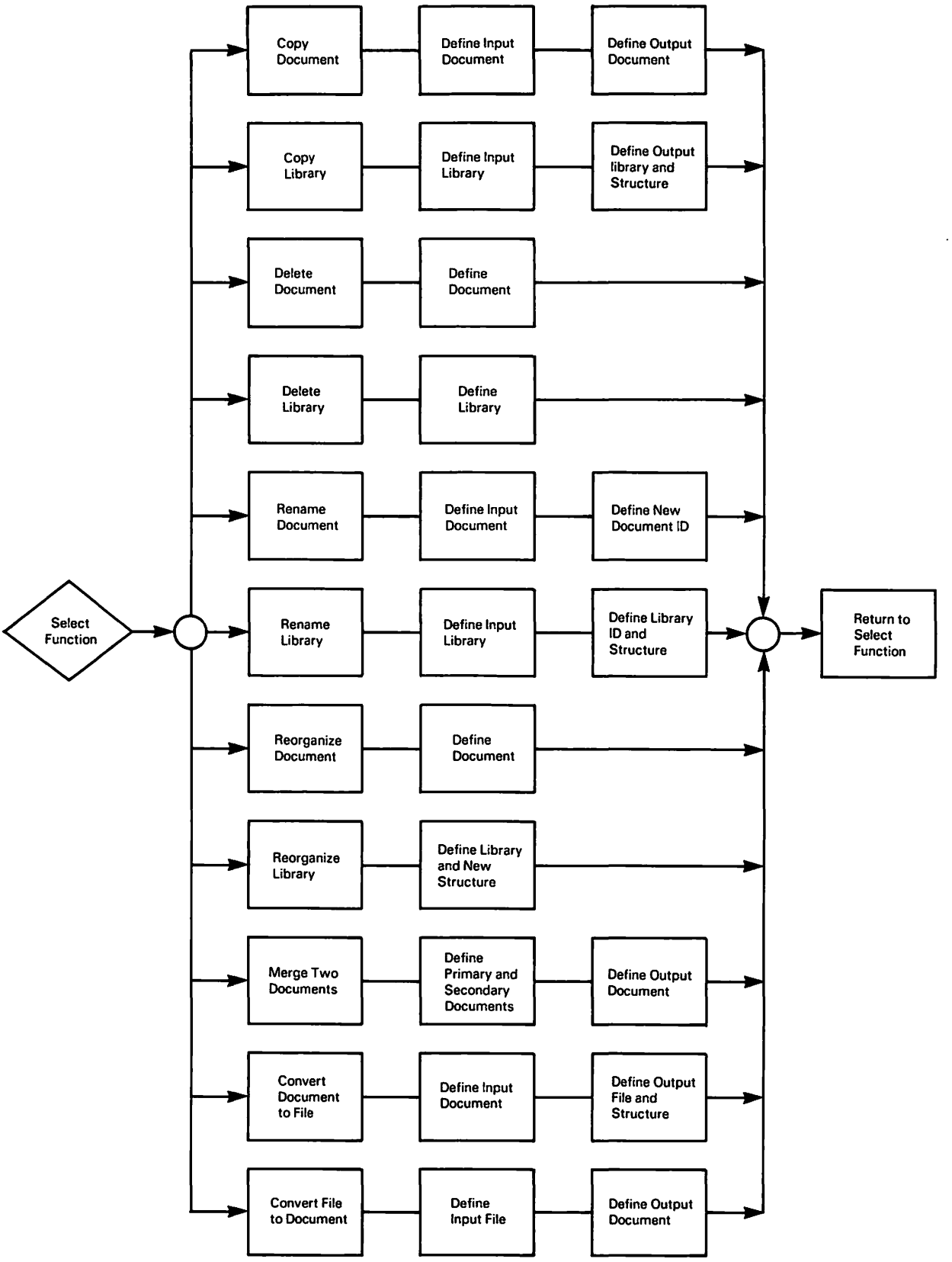


Figure 6-1. COPYWP Processing

6.1.3 Running COPYWP

You can run the COPYWP utility in both data processing (DP) and word processing (WP) environments. In the DP environment, you run COPYWP through the Command Processor RUN Program or Procedure (PF1) function or a Procedure language RUN statement. In the WP environment, you run COPYWP by first selecting the Utilities option from the WP main menu, and then selecting the Document Filing and Conversion function from the VS Word Processing Utilities menu. This chapter is organized as follows:

- Section 6.2 describes how to reference a VS WP document and includes the international options that are available to you.
- Section 6.3 explains how to specify a particular input range or output range for document and library functions.
- Section 6.4 describes the errors that can occur when COPYWP accesses VS WP documents and libraries.
- Section 6.5 discusses the document filing functions that are available from COPYWP.
- Section 6.6 discusses the document conversion functions that are available from COPYWP.
- Section 6.7 explains the file conversion processes for print files, source files, image files, and telecommunications (TC) files.
- Section 6.8 shows how COPYWP supports VS Procedure language control of COPYWP workstation interaction.

6.2 REFERENCING VS WORD PROCESSING DOCUMENTS

The file parameters of a VS WP document are represented differently from those of a data processing (DP) file. A DP file is accessed by a file name, a library name in which the file resides, and a volume name on which the library resides. VS WP documents also have file, library, and volume names, but VS WP documents are accessed by a document ID and a volume name. The volume name indicates the VS volume on which the document and its associated library resides. The document ID and the volume name are the only information that COPYWP and VS WP need to identify a VS WP document.

Each WP document ID has a unique, 4-digit number followed by an uppercase or lowercase letter (0125M, for example). The letter is taken from the suffix of the library name. All WP library names begin with DOCMNT but end with a unique suffix (either a single or a double letter) for each library. These suffixes are attached to DOCMNT to create unique libraries. (Capital letters in a document ID are signified by a single-letter suffix in the library name; lowercase letters in a document ID are signified by the double-letter suffix. For example, document 0125M resides in the library named DOCMNTM. Document 0125m resides in the library named DOCMNTMM.)

For further details about the document ID and other components of the WP document summary, refer to the *Word Processing Reference Manual.*

6.2.1 International Formatting Options

International formats, such as the way a date is written (that is, month/day/year versus day/month/year), differ from country to country. The COPYWP utility incorporates international formatting options that enable you to copy a document and convert it to the format of a specific country. The system administrator sets the international formats for your system through the GENEDIT utility. These formats are transparent to the user for traditional processing.

Copy operations are not affected by a change in international formats. Conversion operations are affected, however. You can change the international formats for a conversion **only** through a COPYWP procedure that has been programmed in the VS Procedure language. Any change in the international format occurs only during COPYWP processing; system formats remain the same.

When you run COPYWP through a procedure, you can specify new format options for conversion operations through a "hidden" GETPARM, identified by the INTERNAT parameter reference name (prname). Hidden GETPARMs are not displayed automatically during interactive COPYWP processing. Because COPYWP supplies default values for the international options, you need only specify the INTERNAT GETPARM if you want to change the default values. The INTERNAT GETPARM enables you to specify the following options for conversion operations:

Option	Description
DATE	Specify the date format: A = "American" (month/day/year); E = "European" (day/month/year). The Date field defaults to the value defined for your VS System through the GENEDIT utility.
DECALIGN	Specify the decimal alignment character (period or comma) that is to be used (for example, the American default value is the period (.) – 23.45). The Decalign field defaults to the value defined for your VS system through the GENEDIT utility.
CURRENCY	Specify the currency symbol to be used (for example, the American default value is the dollar sign ($) – $10.50). The Currency field defaults to the value defined for your VS system through the GENEDIT utility.
REQSPACE	Specify the character to be used to generate a required space character. The Reqspace field defaults to the backslash (\).
DEVCHARS	Specify whether the device-dependent characters (5E beta symbol, 5F paragraph symbol, DE and DF beta and paragraph symbols underscored) should be converted. Conversion of these characters only occurs when copying from WP to DP. The characters are converted to 0E, 0F, 8E, 8F which is the DP representation of these characters. Specify YES to convert the characters. The Devchars field defaults to YES.

6.3 COPYWP PROCESSING

If you do not need to specify new format options, begin interactive processing by pressing PF1 (RUN Program or Procedure) from the Command Processor menu, specify COPYWP in the program field and press ENTER. The COPYWP utility then displays the COPYWP Function Selection screen (Figure 6-2). This section describes the single document and document library processing that is common to all COPYWP functions. Specific filing and conversion functions are described in Sections 6.5 and 6.6, respectively.

```
***  MESSAGE 0001 BY COPYWP

                    RESPONSE REQUIRED BY  PROGRAM  COPYWP
                             TO SELECT FUNCTION



          *** Wang VS COPYWP Utility Program - Version x.xx.xx ***

                    Press PFKey Corresponding To Desired Function

     (1) Copy Single Document                (10) Reorganize Single Document
     (3) Copy Document Library               (12) Reorganize Document Library

     (4) Delete Single Document              (13) Convert Document to VS File
     (6) Delete Document Library             (14) Convert VS File to Document
                                             (15) Document Merge
     (7) Rename Single Document
     (9) Rename Document Library             (16) Terminate Processing
```

Figure 6-2. COPYWP Function Selection Screen

To perform any COPYWP function, you must have appropriate access rights to the document or library file class. The document or library must reside on mounted, VS standard label (SL) volumes. Table 6-1 summarizes the level of access needed for each function. Functions that require Read Only access can operate on any document that is not in use or is only being read by another program. Functions that require Write access can operate only on documents that are not currently in use.

Table 6-1. COPYWP Access Rights

Access Rights	Copy	Delete	Rename	Reorganize	Merge	Convert
Read	X				X	X
Write		X	X	X		

COPYWP is designed to request document and library information in the same way for each function. COPYWP can accept as input (or produce as output) both VS WP documents and VS WP libraries.

6.3.1 Single Document Input

When you select a single document function, the COPYWP Input Document Specification screen (Figure 6-3) appears. This screen appears each time that you select a single document function. The screen prompts you to specify the document ID and volume name of the input document. The Delete Single Document function also includes a Security Erase field which (if you specify YES) writes a pattern over the space on the disk where the document resided.

```
***  MESSAGE 0003 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                          TO DEFINE INPUT


                     Identify Document To Be Copied

DOCUMENT = ****R        [  VOLUME   = ******                    ]
                        [  Optional - Volume Other Than Document ]
                        [             Library's Standard Volume  ]

      Press ENTER To Continue Or PF1 To Terminate Copy Operation
```

Figure 6-3. Sample COPYWP Input Document Specification Screen

The volume name can be omitted (if the volume is mounted and the library that you specified has been created by the Create Library function of the WP Utilities menu). If VS WP is installed on your system, the library for the first input request in a COPYWP session is taken from the library default value set at your workstation. (Default values are set through the Set Workstation Defaults function from the WP Utilities menu.)

After the first input request (or if VS WP is not installed) the input library defaults to the most recently specified input library (if any) in the current COPYWP session. You can override the default library value by supplying another library letter. If the input document is password-protected, another screen prompts you to specify the document's password.

If you want to select another function, press PF1 to return to the COPYWP Function Selection screen (Figure 6-2) from either the COPYWP Input Document Specification screen or the COPYWP Password Specification screen. Otherwise, press ENTER to begin processing of the function that you selected. If you do not specify the correct document ID or volume name, an error screen indicates the error and enables you to respecify the information.

6.3.2 Single Document Output

For single document functions or file-to-document conversion, COPYWP prompts you to enter the document ID and the volume name of the output (destination) document on the COPYWP Output Document Specification screen (Figure 6-4). You do not have to specify the volume name if the document's library has been created using the Create Library function of the WP Utilities menu.

```
***  MESSAGE 0003 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                           TO DEFINE OUTPUT


                  Input Document is 0001R on Volume WP

                      Identify Document To Be Created

DOCUMENT = *****        [  VOLUME   = ******                    ]
                        [  Optional - Volume Other Than Document ]
                        [             Library's Standard Volume  ]


          Press ENTER To Continue Or PF1 To Terminate Copy Operation
```

Figure 6-4. Sample COPYWP Output Document Specification Screen

NOTE
Depending on the COPYWP function that you select, your COPYWP Output Document Specification screen may differ from the one shown in Figure 6-4. Various fields are available for some functions and not for others, but the structure of the screen remains essentially the same.

COPYWP retrieves the most recently supplied (if any) output document library as a default value for an output document ID. You do not need to assign a 4-digit value in the document ID for an output document if you specify the value NEXTR (R being a sample library letter) instead of a numeric value. If you specify NEXTR, COPYWP obtains the next available document number in the library. For example, if 0014R was the last document that was created in library DOCMNTR, then 0015R would be the next document number.

NOTE
Unless you concatenate a library letter to NEXT (e.g., NEXTR), an output document is placed in the next available document in the glossary library.

If you enter the document ID or volume name incorrectly, an error screen indicates the error and allows you to respecify the data. The output document name that you select must be unique to the destination library, otherwise a naming conflict occurs. When naming conflicts occur, an error message displays the actions that you can take to resolve the conflict.

6.3.3 Document Library Input

When you select a library function, COPYWP prompts you to enter the library letter and volume name of the input library on the COPYWP Input Library Specification screen shown in Figure 6-5. You do not have to specify the volume name if the document's library has been created using the Create Library function of the WP Utilities menu.

If VS WP is installed on your system, the library for the first input request in a COPYWP session is taken from your workstation's library default value. After the first input request (or if VS WP is not installed) the input library defaults to the most recently specified input library (if any) in the current COPYWP session. You can override the default library value by supplying another library letter.

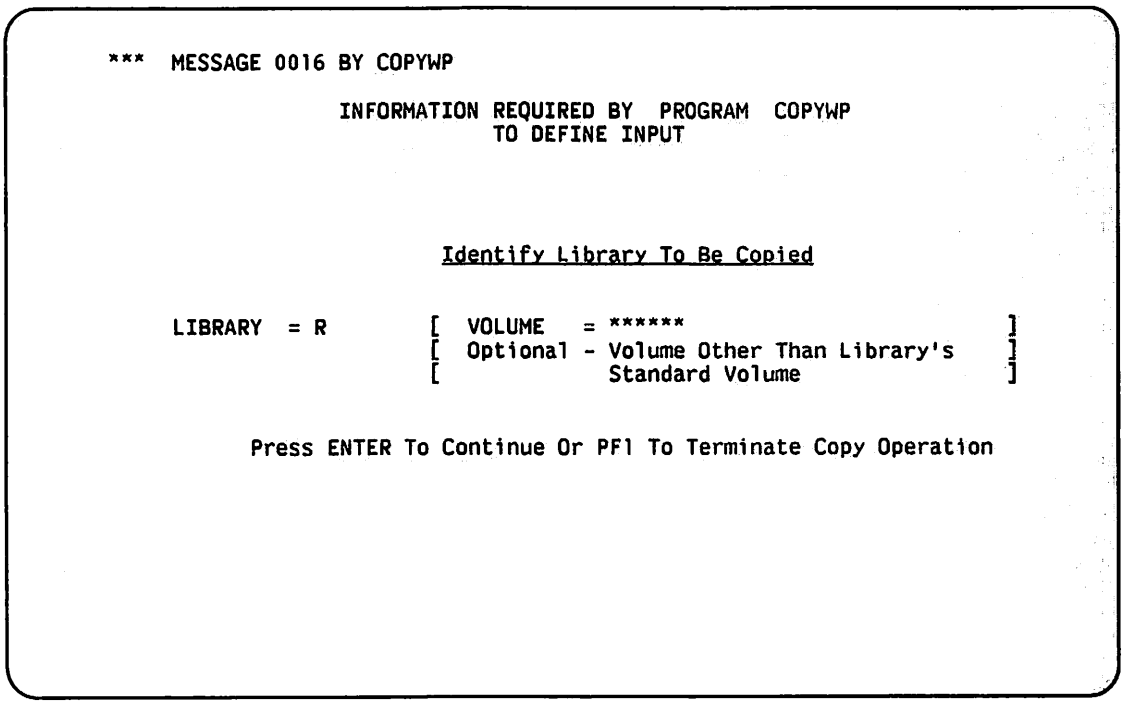


Figure 6-5. Sample COPYWP Input Library Specification Screen

The COPYWP utility displays the COPYWP Password Specification screen for all library functions (except reorganization) when it encounters the first password-protected document. Passwords are not requested for library reorganizations to allow the function to perform in background mode.

COPYWP also displays the COPYWP Password Specification screen when it encounters a document with a **different** password than the password it previously encountered; that is, if all documents in the library have the same password, COPYWP requests the password only once. If you do not know the password, you can skip the password-protected document by pressing PF2, or you can cancel the copy operation for the entire library and return to the COPYWP Function Selection screen (Figure 6-2) by pressing PF1. If you do not know the password for a document, you cannot copy that document. If you specify an incorrect library or volume name, a screen indicates the error and enables you to respecify the information.

6.3.4 Document Library Output

For library functions, the COPYWP Output Library Specification screen (Figure 6-6), prompts you to enter the library letter and the volume name of the output library. You do not have to supply the volume name if the library resides on the volume.

```
***  MESSAGE 0003 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                            TO DEFINE OUTPUT


                     Input Library is R on Volume WP

                Identify Library To Be Created Or Merged Into

     LIBRARY  = *          [  VOLUME   = ******                          ]
                           [  Optional - For Output to Volume Other      ]
                           [             Than Library's Standard Volume  ]

     RENUMBER = NO*          YES To Resequence All Document IDs Within
                             the Library Starting with DOCUMENT = 0001

                           For Document Name Conflicts

     DUPFILES = PROMPT       PROMPT To Handle Documents On An Individual Basis
                             NOCOPY To Skip Copying Of All Such Documents
                             DELETE To Delete All Such Documents

              Press ENTER To Continue Or PF1 To Terminate Copy
```

Figure 6-6. Sample COPYWP Output Library Specification Screen

COPYWP retrieves the most recently supplied (if any) output document library as a default value for the library. Two other fields (Renumber and Dupfiles) are provided to allow you more control over the documents in the output library.

The Renumber Field

The Renumber field renumbers the documents in the output library starting with the document number that you specify (or 0001, which is the default document number). The default value (NO) for the Renumber field uses the input document number for the output document. Specifying YES renumbers the output document library, beginning with the number that you specified (or 0001).

NOTE

Renumbering documents in the library through COPYWP does not affect the number that VS WP uses to create the next document. You can only change the starting number for document creation through the VS Word Processing Create Library Supervisory function of the WP Utilities menu.

If the output library does not exist, COPYWP places a prototype document into Document 0000 in the output library. (A prototype document is a document that creates a default format line for other documents in the library. Without the prototype document, documents cannot be created.) The other input documents are placed in output documents beginning with the document number that you specified. The renumbering operation always transfers an input prototype document to Document 0000. If the output library already contains a prototype document, the conflict is resolved by the Dupfiles option.

If the output library exists when you select the renumbering function, the documents that are already in the output library are renumbered from the specified starting document number before the input documents are transferred. An existing prototype document, or any document to which you do not have Write access, is not renumbered.

The Dupfiles Field

The Dupfiles field determines how to resolve naming conflicts. Naming conflicts occur when you specify a document ID as output to a library that already contains a document with the same ID. If you select the Renumber function, a conflict can only occur with prototype documents. The three options of resolving conflicts are NOCOPY, DELETE, and PROMPT. The default value is PROMPT. Each option is described as follows:

Option	Description
NOCOPY	If you select the NOCOPY option and a naming conflict occurs, the input document is bypassed (that is, the function is not carried out on the input document and the existing document of the same name is unaffected).
DELETE	The DELETE option deletes the document that already resides in the output library and replaces it with the input document. (You need Write access to the existing document, and the expiration date must have passed.)
PROMPT	The PROMPT option prompts you each time a document ID naming conflict occurs. In each case, a screen provides you with the following options to resolve the conflict. Specify how you want to resolve the naming conflict: N *No Copy*. Do not process this document. D *Delete* the document that already resides in the output library and replace it with the input document. (You need Write access to the existing document, and the expiration date must have passed.) R *Rename* the document that already resides in the output library as specified in the NEWDOCID field and replace it with the input document. (You need Write access to the existing document, and the expiration date must have passed.) C *Change* the input document's ID as specified in the NEWDOCID field, which is supplied to you when you specify the Change option. If you specify the Next option, another conflict could possibly occur.

6.4 COPYWP ERROR CONDITIONS

Two error conditions can be detected by COPYWP during a single document operation:

1. **Document Damaged** — If COPYWP determines that the input document is damaged, it terminates the copy and displays an error message. (You can usually recover a damaged document when the document is edited through VS WP.)
2. **I/O Error** — An I/O error message is displayed if COPYWP detects an error with the disk drive or storage medium.

Two other types of errors can also be detected during a document library copy operation. The following errors terminate the operation on the current document and cause an error message to appear:

1. **Current Document Error** — If only the current document is damaged, COPYWP allows two responses. Press PF1 to terminate the entire library copy operation, or press PF2 to bypass the current document and continue the operation with the next document.
2. **Fatal Error** — If more serious errors occur, COPYWP only allows you to terminate the entire library copy operation by pressing PF1.

If COPYWP runs as a noninteractive procedure, a current document error automatically skips over the damaged document and continues the operation with the next document. A fatal error cancels the COPYWP program.

6.5 DOCUMENT FILING FUNCTIONS

The document filing functions (refer to the COPYWP Function Selection screen, Figure 6-2) perform maintenance operations on VS WP documents and libraries. The filing functions are

- Copy Single Document (PF1)
- Copy Document Library (PF3)
- Delete Single Document (PF4)
- Delete Document Library (PF6)
- Rename Single Document (PF7)
- Rename Document Library (PF9)
- Reorganize Single Document (PF10)
- Reorganize Document Library (PF12)
- Document Merge (PF15)

6.5.1 Copy Single Document (PF1)

The Copy Single Document function (PF1) from the COPYWP Function Selection screen (Figure 6-2) enables you to make a copy of a specified single document. All unused space in the input document (caused by frequent edits) is eliminated from the output document. The copy operation preserves any glossary verification information contained in the document. To perform the copy, specify the input and output documents (refer to Sections 6.3.1 and 6.3.2). Note that you cannot copy a document to itself.

6.5.2 Copy Document Library (PF3)

The Copy Document Library function (PF3) copies all documents in a specified input library to an output library (and optionally renumbers the documents in the output library). The output library can be a new or existing library. All unused space in the input documents is removed from the output documents. The copy operation preserves any glossary verification information contained in library documents. After selecting the Copy Document Library function, specify the input library (refer to Section 6.3.3), and the output library and organization (refer to Section 6.3.4).

6.5.3 Delete Single Document (PF4)

The Delete Single Document function (PF4) deletes a single document from a volume, freeing the disk space it once occupied. In order to delete a document, the document expiration date must have passed, you must have Write access to the document, and you must provide any assigned password.

When you select the Delete Single Document function, COPYWP requests the input document fields (refer to Section 6.3.1). The Delete Single Document Input screen also allows you to select the Security Erase function. Because this feature involves more time and system resources than normal document deletion, the default value of the Erase field is NO. You can select the Security Erase function by changing this value to YES. COPYWP deletes the document **without** prompting you for confirmation.

Security Erase

COPYWP provides a Security Erase function that writes over the data on the disk as well as the document's VTOC entry. Files and documents are usually deleted by eliminating any reference to the document in the VTOC, but this does not destroy the actual data of the document. Unless you set the Security Erase to YES, the data remains on the disk until it is written over by another file or document. Security Erase is intended for the deletion of confidential or sensitive documents. If you select Security Erase, all blocks spanned by the document that you delete are written over with binary ones before the VTOC is updated.

6.5.4 Delete Document Library (PF6)

The Delete Document Library function (PF6) deletes an entire document library from a disk volume, freeing the disk space it once occupied. In order to delete a library, the expiration dates of all documents in the library must have passed, you need Write access to each document, and you must provide any assigned password(s).

When you select the Delete Document Library function, COPYWP requests you to enter the input library fields (refer to Section 6.3.3). The Delete Document Library Input screen also allows you to select the Security Erase function. Because this function consumes more time and resources than normal library deletion, the default value of the ERASE field is NO. If you change the default value to YES, all documents in the library are deleted with the Security Erase function. If you select Security Erase, all blocks spanned by all documents in the library are written over with binary ones before the VTOC is updated.

NOTE
While the library is being deleted a message appears on the screen acknowledging the deletion. However, when the COPYWP Function Selection screen (Figure 6-2) reappears, confirmation of deletion is not displayed.

6.5.5 Rename Single Document (PF7)

The Rename Single Document function (PF7) changes the ID of a single document. You must have Write access to the document and must provide any assigned password. You can use the Rename function to change both the document ID and the library ID. If you change the library ID, the document is assigned to the new library (which does not need to be on the same volume). If the library of the new document ID is on a different volume from the library of the original document ID, the document is moved to the new volume. The document is not copied if the volume is not mounted. When the document is moved, it is copied and the original document is deleted. When you select this function, COPYWP requests you to enter the input document fields (refer to Section 6.3.1), and then requests the output document fields (refer to Section 6.3.2).

6.5.6 Rename Document Library (PF9)

The Rename Document Library function (PF9) changes the library IDs of all the documents in a library by changing their document ID library letters (assigning them to a new library). The new library may or may not already exist, and can exist on a different volume. Optionally, you can renumber the documents in the output library.

To rename a library, you need Write access to all the documents and must provide any assigned passwords. If the new library is on a different volume from the original library, all documents in the original library are moved to the new volume. When the documents are moved, they are copied and the original documents are deleted. To rename a document library, first define the input library (refer to Section 6.3.3). COPYWP then requests the output library fields and structure options (refer to Section 6.3.4).

6.5.7 Reorganize Single Document (PF10)

The Reorganize Single Document function (PF10) eliminates all unused disk space within a document and consolidates the document to a single disk extent. You need Write access to the document and must know any assigned password. To reorganize a document, you need only specify the input document (refer to Section 6.3.1).

6.5.8 Reorganize Document Library (PF12)

The Reorganize Document Library function (PF12) eliminates unused disk space within all documents in a library and consolidates each document to a single disk extent. Optionally, this function can also renumber the documents within a library. To reorganize a library, you need Write access to all the documents in the library.

To reorganize a document library, supply the information requested on the COPYWP Reorganize Document Library Input screen (Figure 6-7). Library specification is described in Section 6.3.3, and the Renumber option is identical to that described in Section 6.3.4.

```
***  MESSAGE 0016 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                          TO DEFINE INPUT


                     Identify Library To Be Reorganized

    LIBRARY  = R        [  VOLUME   = ******                    ]
                        [  Optional - Volume Other Than Library's ]
                        [             Standard Volume           ]

    RENUMBER = NO*         YES To Resequence All Document IDs Within
                           the Library Starting with DOCUMENT = 0001

          Press ENTER To Continue Or PF1 To Terminate Operation
```

Figure 6-7. Sample COPYWP Reorganize Document Library Input Screen

6.5.9 Document Merge (PF15)

The Document Merge function (PF15) merges two input documents in the same way that the Merge Print function through WP merges two print documents except that the output is a WP document. The resulting output document can then be edited and printed.

NOTE

Documents must be properly set up for merging. For information concerning document merge set up, refer to the Word Processing Reference Manual.

If you select the document merge function, you are prompted to specify the two documents that you want merged. Each document is specified separately as a primary or secondary document. (The primary document normally contains standard text. The secondary document normally contains variable text.)

A standard-text document contains information that will not change; for example, an employee form that does not have information placed in it's variable fields. A variable field on an employee form can be an area that appears next to the text of a standard text document like "Name" (variable field), "Address" (variable field), and "Zip Code" (variable field). A variable-text document contains information that supplements the standard text document's variable fields. Refer to the *Word Processing Reference Manual* for more information about standard- and variable-text documents.

After you have specified the two documents, you are prompted to specify an output document. If you specify a document ID that already exists, an error message indicates this and allows you to respecify the output document ID or press PF3 to delete the existing document. When processing begins, the primary document is copied into the output document. The merge process copies the Header, Footer, Work pages, and text from the primary document.

If COPYWP encounters a merge graphic (↕) in the primary text, it pauses and opens the secondary document. COPYWP then takes the first variable entry from the secondary document and copies it into the output document. The end of a variable entry is marked by another merge graphic. When COPYWP encounters this merge graphic in the secondary document, it resumes copying the primary document from where it left off. It repeats the process until both documents have been merged.

If either the primary or secondary documents contain a glossary, the glossary is not copied into the output document. If the output document exceeds 120 pages or becomes too large to archive on a single diskette, you are prompted for an additional document ID. If you specified NEXT as the original output document ID, the merge process continues uninterrupted with the next available document ID in the specified library.

6.6 DOCUMENT CONVERSION FUNCTIONS

The document conversion functions convert VS WP documents to text data records within certain forms of conventional VS data files. They also convert text within certain forms of conventional VS data files into VS WP documents. The filing functions are as follows:

- Convert Document to VS File (PF13)
- Convert VS File to Document (PF14)

NOTE
With the exception of conversions to and from 2780 TC data file format, conversion functions are not symmetric. Use of conversion functions in a circular fashion does not yield files or documents equivalent to the original documents or files.

The document conversion functions convert an entire file or document. COPYWP does **not** provide field or record selection options. If you need to update documents or perform extensive text string searches from within programs, you should use the VS Document Access Subroutines described in the *VS Programmer's Guide to VS/IIS*. The subroutines are more efficient and may require less time and resources than the conversions described here. They also preserve all document formatting characters.

The conversion process is performed for the current COPYWP processing through the INTERNAT GETPARM (subject to the International options that were selected or defaulted). All conversions performed during processing must use the same International options (refer to Section 6.2.1).

The COPYWP conversion process is only **approximate**. Documents have formatting codes and conventions that are not analogous to the conventional VS data file, and vice versa. These incompatibilities are discussed in detail in Sections 6.6.1 and 6.6.2.

6.6.1 Convert Document to VS File (PF13)

The Convert Document to VS File function (PF13) transforms a VS WP document into a specified type of VS DP file. You need Read access to the input document and, if the document is password-protected, you must provide the password. If you select this function, COPYWP prompts you to specify the input document ID. When you press ENTER, the COPYWP Conversion Type screen (Figure 6-8) appears.

```
***  MESSAGE 0003 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                          TO DEFINE OUTPUT

                  Input Document is 0065R on Volume WP

                       Identify The File To Be Created

   FILE     = ********   in   LIBRARY  = ********   on   VOLUME   = ******

 Conversion TYPE     = ******   DATA   To produce a VS Program Accessible File
                                PRINT  To produce a VS Print File
                                SOURCE To produce a VS Program Source File
                                TC     To produce a 2780 Transmittable File

       Press ENTER To Continue Or PF1 To Terminate Conversion Operation
```

Figure 6-8. COPYWP Conversion Type Screen

The COPYWP Conversion Type screen prompts you for the file, library, and volume names of the output file, and the file structure in which the output file is to be constructed. The library and volume names default to your OUTLIB and OUTVOL values, respectively.

You can convert the input VS WP document to a consecutive data file, a print file, a program source file, or a TC file. After you specify one of the Conversion Type field values, conversion processing occurs. If subsequent document-to-file conversions are processed, Conversion Type defaults to the previously specified value. The following paragraphs describe the Conversion Type options.

The Data Format Conversion Type Option

Converting a VS WP document to the Data format produces a file that can be suitable for access by a user program. The resulting file is consecutive and compressed, with a 628-byte maximum record length. The Data format is organized for easy access by a user program.

The Data format contains two record types: the first record type describes the format of the text; the second contains the actual text of the document (as well as information describing the current record). The format specification of the two record types produced through the TYPE = DATA option of the Convert Document to VS File function is described as follows.

A Format Line record describes the arrangement of the text records that follow it. A Format Line record signals either the occurrence of a new page in the original document or another format line. The record length is entirely dependent upon the number of TAB (▸) characters in the corresponding format line of the original document. A byte-by-byte description of the Format Line Record is shown in Table 6-2.

Table 6-2. Format Line Record Description Table

Field Number	Data Type	Length [a]	Field Description
1	Character	1	Byte 1 contains the letter F
2	Character	1	Byte 2 signifies whether the format line is a new format line (0) or a new page (1).
3	Binary	2	Byte 3 contains the current record length
4	Character	1	Byte 4 determines the line spacing: 1 — Single spacing 2 — Double spacing 3 — Triple spacing H — 1/2 Spacing Q — 1/4 Spacing W — 1 1/2 spacing
5	Reserved	1	
6	Binary	2	Byte 6 contains the maximum length of data records (until the next format record or end of file)
7	Binary	2	Byte 7 contains the first tab position or 0 (no tabs)
8	Binary	2	Byte 8 contains the length from the first tab position to the second tab position, or if no additional tabs are in the format line, the length from the first tab position to the RETURN (◂) (the end of the format line). If no tabs are in the format line, Byte 8 contains the length of the entire format line. Repeat for each additional tab position.
5+2n	Binary	2	If n = 1 then see Byte 7. If n > 1 then each byte represented by 5 + 2n contains the nth tab position.
6+2n	Binary	2	Byte 6 + 2n contains the length from the nth tab position to the (n + 1)th tab position, or if no additional tabs are in the format line, the length from the nth tab position to the RETURN (◂) (the end of the format line).

[a] Length is specified in bytes

The Text record provides the actual text of the document, in screen format, without any formatting codes other than RETURN (◂) and NOTE (!!). A byte-by-byte description of the Text Record is shown in Table 6-3.

Table 6-3. Text Record Description Table

Field Number	Data Type	Length [a]	Field Description
1	Character	1	Byte 1 contains the letter D.
2	Reserved	1	Byte 2 is reserved.
3	Binary	2	Byte 3 contains the current record length (n + 4 bytes).
4	Character	n	Byte 4 contains the document text formatted as it would approximately appear printed, unjustified, with a left margin of 0, delimited by a RETURN (◂) character (X'03'), if one is on that line.

[a] Length is specified in bytes

Table 6-4 shows the restrictions in conversions from VS WP documents to VS data files and lists of those document elements that cannot be transferred to the DP file.

Table 6-4. Document-to-File Conversion Restrictions

Document Feature	Data Conversion	Print Conversion	Source Conversion
Document summary	Ignored	Optional	Ignored
Headers/footers	Ignored	Implemented	Ignored
Single underscore	Ignored	Implemented	Ignored
Double underscore	Ignored	Single underscore	Ignored
1/4 Spacing	Described by format record	Single spacing	Single spacing
1/2 Spacing	Described by format record	Single spacing	Single spacing
1 1/2 Spacing	Described by format record	Double spacing	Double spacing
Superscripts	Ignored	Ignored	Ignored
Subscripts	Ignored	Ignored	Ignored
Emphasis printing	Ignored	Implemented	Ignored
Notes	Implemented	Optional	Implemented
Stop codes	Ignored	Ignored	Ignored
Merge codes	Ignored	Ignored	Ignored
Don't Merge codes	Ignored	Ignored	Ignored

NOTE
You cannot convert files in Data format back to VS WP documents.

The Print Format Conversion Type Option

Converting a VS WP document to the Print format produces a VS print file that provides a line-formatted representation of the original document. The resulting print file can be automatically queued to print through PRNTMODE field default (set through the SET Usage Constants function (PF2) of the Command Processor menu).

You can specify certain formatting options on the COPYWP Print File Options screen shown in Figure 6-9. Default information is obtained from the document. Refer to Table 6-4 for a listing of those document elements that cannot be transferred to the DP file.

```
***  MESSAGE 0004 BY COPYWP

                INFORMATION REQUIRED BY  PROGRAM  COPYWP
                            TO DEFINE PRINT



          Specify Print Options for Document 0065R on Volume WP

Print from Page    START    = 001     Print thru Page       FINISH   = 001
Starting as Page   NUMBER   = 0002    First Header Page     HEADER   = 002
First Footer Page  FOOTER   = 002     Footer Starting Line  LINE     = 50
Left Margin        MARGIN   = 010

FORMAT   = UNJUSTIFIED    JUSTIFIED   for Justified output
                          UNJUSTIFIED for Unjustified output
                          NOTES       for output With Notes

STYLE    = FINAL      FINAL Style      SUMMARY  = PRINT  PRINT Summary
                      DRAFT Style                        OMIT  Summary

            Press ENTER To Continue Or PF1 To Terminate Operation
```

Figure 6-9. Sample COPYWP Print File Options Screen

The Source Format Conversion Type Option

Converting a VS WP document to the Source format produces a file that can be accessed by the EDITOR utility and VS language compilers. Documents to be converted must **not** have any format lines that exceed 81 characters. If any line in the document is 81 characters long, character 81 must be a RETURN graphic (◂).

You must provide the proper line sequence numbers within the document or leave space for such sequence numbering by the EDITOR. Refer to Table 6-4 for a listing of those document elements that cannot be transferred to the DP file.

The Telecommunications (TC) Format Conversion Type Option

Converting a VS WP document to the TC (telecommunications) format produces a file that can be used as input to the VS TCCOPY utility for 2780-type transmission to another Wang Word Processing System, Office Information System (OIS), or VS System. This conversion preserves all of the original document's formatting information.

VS WP documents have some formatting codes and conventions that are not analogous to conventional VS data files. Refer to Table 6-4 for a listing of those document elements that cannot be transferred to the DP file.

6.6.2 Convert VS File to Document (PF14)

The Convert VS File to Document function (PF14) transforms VS Print, Source, Image (text), and TC DP files into VS WP documents. No other type of file, including the output of a DATA type document-to-file conversion, can be converted to a document. Thus, COPYWP cannot convert indexed, program, or WP files to VS WP documents.

If you select this function, COPYWP prompts you to specify the file, library, and volume names of an input DP file. If you incorrectly specify the file, library, or volume names, COPYWP prompts you to respecify the input fields. You must also respecify the fields if you do not have Read access to the file or if COPYWP encounters an ownership conflict.

After you identify the input DP file, you must provide the output document fields on the COPYWP Output Document Specification screen (refer to Figure 6-4). Specify the VS WP document ID, the volume name of the output document, and the document summary information (title, operator, author, and comments). COPYWP determines the input file's format based on the file's label, so you do not need to specify the file type for print and TC file conversions.

Because 80-byte consecutive files can be converted as Source or Image files, you must select the conversion type for 80-byte record files. Consecutive data files with record lengths greater or less than 80 bytes are converted to documents as Image files. Depending on the type of input file, COPYWP can also provide two additional conversion options: Lines per Page and Automatic Insertion of Tabs. These options are described in the context of the particular file conversion process.

6.7 FILE CONVERSION PROCESSES

This section describes the file conversion processes of the Print, Source, Image, and TC input files.

6.7.1 Print Files

A Print file is a consecutive, compressed file with the print attribute. To be acceptable as input, the maximum record length (line length) of a print line cannot exceed 160 characters (including the print control characters).

Each page of the print file translates into a page of the document, based upon skips to Channel 1 carriage control characters that are set for printers. Channel 1 indicates to a printer how many lines it needs to advance to get to the top of the next page. If the print file exceeds 120 pages or is too large to be archived on a single diskette, the utility prompts you for an additional document ID. However, if you specified Next as the original output document ID, the conversion process continues uninterrupted with the next available document ID in the specified library.

All occurrences of carriage control skips to channels other than Channel 1 (top of form) are translated into a RETURN (◂) and an Operator NOTE (!!) on a line indicating the location of the original channel skip. COPYWP does not offer the Lines per Page option on the COPYWP Output Document Specification screen (Figure 6-4) because the print file automatically determines page breaks. Unless the International options indicate otherwise, the following hexadecimal codes in the input print file are converted as indicated:

Input Code	Output Code
00-0D	5C (\)
80-8D	5C (\)
5E	5C (\)
0E(↑)	5E
0F(↓)	5F

Specify YES or NO for Automatic Insertion of Tabs for Print file conversion. If you specify YES (the default value), COPYWP determines optimal tab settings on a page-by-page basis and uses TAB (▸) and DEC TAB (⌴) characters where appropriate. Depending on the contents and format of the text within the print file, the Tab feature can substantially reduce the document's disk storage requirements and ease subsequent manual editing of the document. If you specify NO, no TAB or DEC TAB characters appear in the resulting document. If you intend to typeset the document, specify NO.

6.7.2 Source Files

A Source file is a consecutive file with 80-byte fixed-length records, or a consecutive file with compressed records that have a maximum length of 80 bytes. To be acceptable as input for COPYWP, the input file must be acceptable to theVS EDITOR utility or the product of the VS EDITOR, but it does not need to have line sequence numbers.

Files that are eligible for Source file conversion can also be converted to documents as Image files. The COPYWP Output Document Specification screen (Figure 6-4) also allows you to switch to Image conversion.

Consecutive input files (with 80-byte record lengths) default to Source file conversion. COPYWP invokes the Image file conversion process if you set the Type field value to Image. Each record of the input file translates into a line of the document. If the resulting document exceeds 120 pages or requires too much space to be archived, the utility prompts you for an additional document ID. However, if you specified Next as the original output document ID, the conversion process continues uninterrupted with the next available document ID in the specified library. Unless the International options indicate otherwise, the following hexadecimal codes in the input Source file are converted as indicated:

Input Code	Output Code
00-0D	5C (\)
80-8D	5C (\)
5E	5C (\)
0E(↑)	5E
0F(↓)	5F

COPYWP enables you to set the number of lines per page through the Lines per Page option on the COPYWP Output Document Specification screen (Figure 6-4) because no pagination is contained in the input file. The COPYWP utility retrieves a default value for Lines per Page from your Default Lines per Page field. Lines Per Page can be set by PF2 of the Command Processor menu, a procedure, or a program.

Specify YES or NO for Automatic Insertion of Tabs for Source file conversion. If you specify YES (the default value), COPYWP determines optimal tab settings on a page-by-page basis and uses TAB (▸) and DEC TAB (⌴) characters where appropriate. The Tab feature can substantially reduce the document's disk storage requirements and ease subsequent manual editing of the document (depending on the contents and format of the text within the Source file). If you specify NO, no TAB or DEC TAB characters appear in the resulting document. If you intend to typeset the document, specify NO.

6.7.3 Image Files

An Image file is any consecutive data file that does not qualify for Print or TC conversion. Image file conversion is automatically invoked for eligible files with record lengths other than 80 bytes. (For 80-byte record files, a different COPYWP Output Document Specification screen allows you to specify the conversion type, as well as the document summary fields.) You should ignore the Automatic Insertion of Tabs and Lines per Page fields that appear on the COPYWP Output Document Specification screen (Figure 6-4) for 80-byte record files, because these options do not operate for the Image file conversion process.

Source file conversion provides automatic conversion, but Image file conversion provides more direct control over the format of the output document. Pagination in the document is indicated by a slash (/) in the first position of a record in the data file (COPYWP provides no automatic pagination). Automatic document overflow handling is **not** available for Image file conversions. You must convert the input file to a single document.

With the exception of format (|) and page (|) characters, which are represented by hexadecimal 06 and 86 respectively, all legal document control characters in the input file pass directly into the output document. This preservation of document control characters is especially useful in the case of merge and don't merge control characters. Table 6-5 relates each document control character to its hexadecimal code and function.

Table 6-5. Document Control Character Codes

Document Control Character	Hexadecimal Code	Function
♦	01	Center
▸	02	Tab
◂	03	Return
→	04	Indent
⌐	05	Dec Tab
|	06	Format
■	0B	Stop
!!	0C	Note
↕	0D	Merge
↑	0E	Superscript
↓	0F	Subscript
⊥	86	Page
↕	8D	Don't Merge

The characters that are illegal in a VS WP document or inappropriate in the Image file are converted as shown in Table 6-6:

Table 6-6. Illegal Document Control Character Conversion Codes

Input Code	Output Code
00	20 (' ')
06 (|)	2A ('*')
80	A0 ('_')
81	01
82	02
83	03
84	04
85	05
86 (⊥)	AA ('*')

After you specify the output document fields (and select Image file conversion, if necessary), COPYWP displays the COPYWP Image Conversion Options screen shown in Figure 6-10.

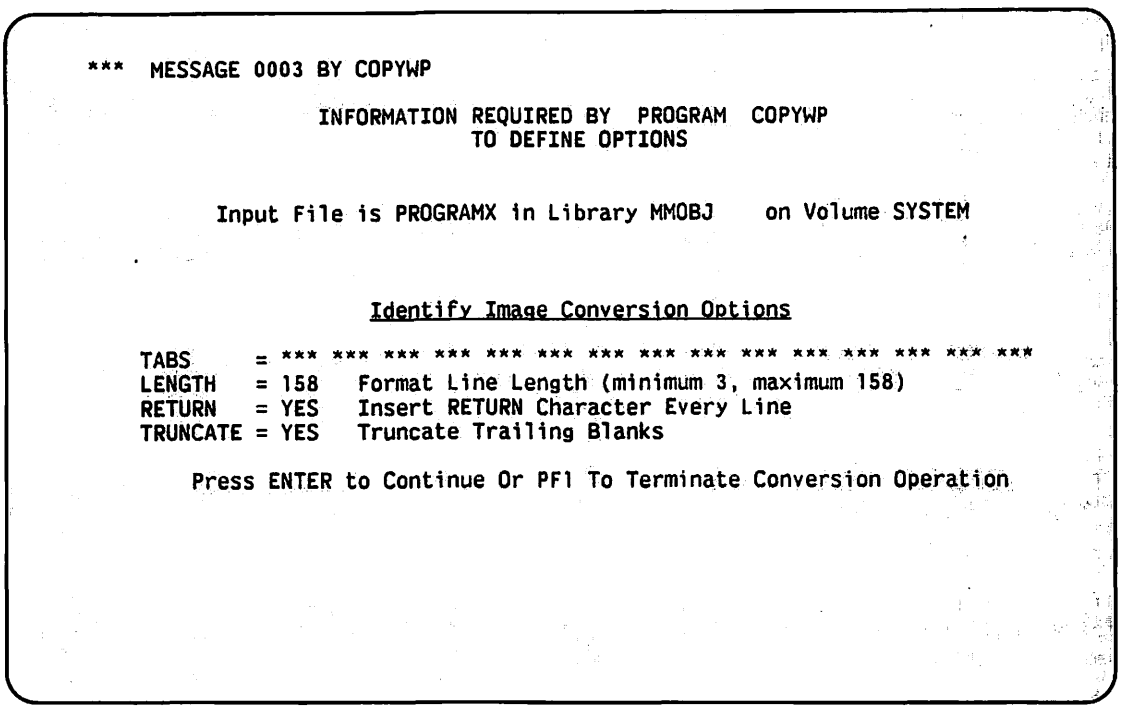


Figure 6-10. Sample COPYWP Image Conversion Options Screen

The Image Conversion Options screen fields are described as follows:

Field	Description
TABS	Specify the column positions where you want tabs set. If blanks coincide with the selected column settings COPYWP converts the blanks in the input file to tabs. You can specify up to 15 tab positions. A tab can be located in any column between column 3 and the number value of the Length field minus one. You should not set tab positions if the output document will be typeset.
LENGTH	Specify the length of the format line in the resulting document. Length defaults to one greater than the record length of the input file or to 158, whichever is smaller.
RETURN	Accept the default value (YES) if you want to place a return character at the end of each record in the input file. The default value of Return is YES because an input file record generally corresponds to a document line. However, NO is appropriate for custom-designed VS files that contain their own carriage returns and/or other control characters.
TRUNCATE	Accept the default value (YES) if you want to truncate trailing spaces from the end of each line, saving space in the output document.

6.7.4 TC Files

A TC file is the product of the VS TCCOPY utility. It results from a 2780-type transmission from another Wang Word Processing System, Office Information System (OIS), or VS System. TC conversion preserves all of the original document's formatting information.

You are not prompted for the Automatic Insertion of Tabs option or for a value for Lines per Page because pagination, format, and tab information are contained in the TC file.

6.8 A SAMPLE COPYWP PROCEDURE

You can control COPYWP workstation interaction through the VS Procedure language. For example, document filing or conversion operations that you regularly perform can be automated by writing a procedure that specifies all functions, input, and output. Appendix A provides a complete list of all COPYWP GETPARMs. Consult the *VS Procedure Language Reference* for information about procedure syntax.

COPYWP also enables you to select International options for document conversion operations through a hidden GETPARM. The hidden GETPARM is identified by the INTERNAT parameter reference name (prname), and allows you to specify the date format, date separator, time separator, decimal point, thousands separator, currency symbol, and national code. Because COPYWP supplies default values for these fields based on the format specified for your system through GENEDIT, you need only specify the INTERNAT GETPARM if you want to override the system defaults or change the default values for other International options.

The values specified through the INTERNAT GETPARM remain in effect during an entire COPYWP session. All conversion operations performed during the session must use the same International options because you cannot respecify the GETPARM for individual operations. The keywords, meanings, and default values for the INTERNAT GETPARM are listed in Appendix A.

The following procedure performs periodic maintenance on two VS WP libraries. The procedure reorganizes the M document library and renumbers the resulting library, beginning with document 1001. The procedure then copies Document 2011a to another volume before deleting the "a" library and exiting from COPYWP. You must enclose lowercase library letters, passwords containing lowercase letters, or document IDs with lowercase library letters in matching single or double quotes to prevent the Procedure Interpreter from converting the value to uppercase.

```
PROCEDURE
    RUN COPYWP
    ENTER FUNCTION 12
    ENTER INPUT LIBRARY=M, VOLUME=SYSTEM, RENUMBER=YES,
        DOCUMENT=1001
    ENTER FUNCTION 1
    ENTER INPUT DOCUMENT='2011a', VOLUME=SYSTEM
    ENTER PASSWORD PASSWORD=BAGGINS
    ENTER OUTPUT DOCUMENT='2011a', VOLUME=ALTSYS
    ENTER FUNCTION 6
    ENTER INPUT LIBRARY='a', VOLUME=SYSTEM, ERASE=NO
    ENTER PASSWORD PASSWORD=BAGGINS
    ENTER FUNCTION 16
RETURN
```

CHAPTER 7
THE DISKINIT UTILITY

7.1 INTRODUCTION

The primary function of the DISKINIT utility is to initialize disks, also referred to as volumes. Initialization prepares a new volume for use. A new volume (including a new member of a volume set) must be initialized before you can store programs or data on that volume or volume set. Volume sets are described in Chapter 1. This chapter describes the various aspects of DISKINIT as follows:

Section	Topic
7.2	An overview of DISKINIT processing.
7.3	Mounting volumes for DISKINIT processing.
7.4	Using DISKINIT on Nonlabeled volumes.
7.5	The Initialize function — initializes a new disk volume and creates a Volume Table of Contents (VTOC). You can also allocate a dump file, and allocate a page pool. Any data existing on the disk volume isdestroyed.
7.6	The Reformat function — reformats a previously initialized disk volume by creating a new volume label and VTOC. You can use this function to allocate a dump file and a page pool. The access to the data is destroyed.
7.7	The Relabel function — relabels a disk volume by changing the volume name. You have the option to allocate a page pool. For a VS15, VS25, VS45, or VS65, Relabel also calculates a pointer for the bootstrap file, if the file exists. Data on the disk volume is not affected.
7.8	The Verify function — reads a disk volume to ensure that all blocks on the disk surface can be addressed and read. Data on the volume is not affected.
7.9	The Remove function — removes a bad block from use on the disk volume and is only valid for the disk drives that have cylinders that can be (and are) reserved for replacement of bad blocks.
7.10	The Erase function — erases all data from a disk volume in accordance with the *Department of Defense Industrial Security Manual* (DoD #5220.22-M, Section 13, Paragraph 116). The volume does not need to be reinitialized because it contains a standard empty VTOC after the Erase is performed.
7.11	Fault Tolerance for a volume.
7.12	Dump file allocation.
7.13	Page pool allocation.
7.14	A sample DISKINIT Procedure language procedure.

7.2 DISKINIT PROCESSING AND FUNCTION SPECIFICATION

To begin DISKINIT processing, press PF1 (RUN Program or Procedure) from the Command Processor menu (refer to Figure 1-1). Specify DISKINIT in the Program field on the Run Program or Procedure screen (refer to Figure 1-2) and press ENTER. The DISKINIT Function Specification screen (Figure 7-1) appears when DISKINIT processing begins. Specify the DISKINIT function that you want performed, the volume on which the function is to run and include the VSID (volume set identification) number if the volume is a member of a volume set. VSID 0 (the default) indicates that the volume is not a member of a volume set. (A blank in the VSID field is interpreted as VSID 0.) If you are initializing a new volume, specify the name to be assigned to the new volume. DISKINIT operates on all types of Wang-supported disks and diskettes. Figure 7-2 shows an overview of DISKINIT processing.

```
*** MESSAGE 001  BY WT1M00
                RESPONSE REQUIRED BY PROGRAM DISKINIT
                       TO SELECT FUNCTION
                   ACTIVE SUBPROGRAM IS DISKINIT

Please enter a function and specify volume to be processed.



          WANG VS DISK INITIALIZE UTILITY PROGRAM - VERSION xx.xx.xx
                    (Copyright, Wang Laboratories, Inc. 1985)

                       Functions: Initialize
                                  Reformat
                                  Relabel
                                  Verify
                                  Remove
                                  Erase

     FUNCTION = VERIFY****   on   VOLUME   = ******        VSID    =    0**

  (13) Display Information                          (16) Exit
```

Figure 7-1. DISKINIT Function Specification Screen

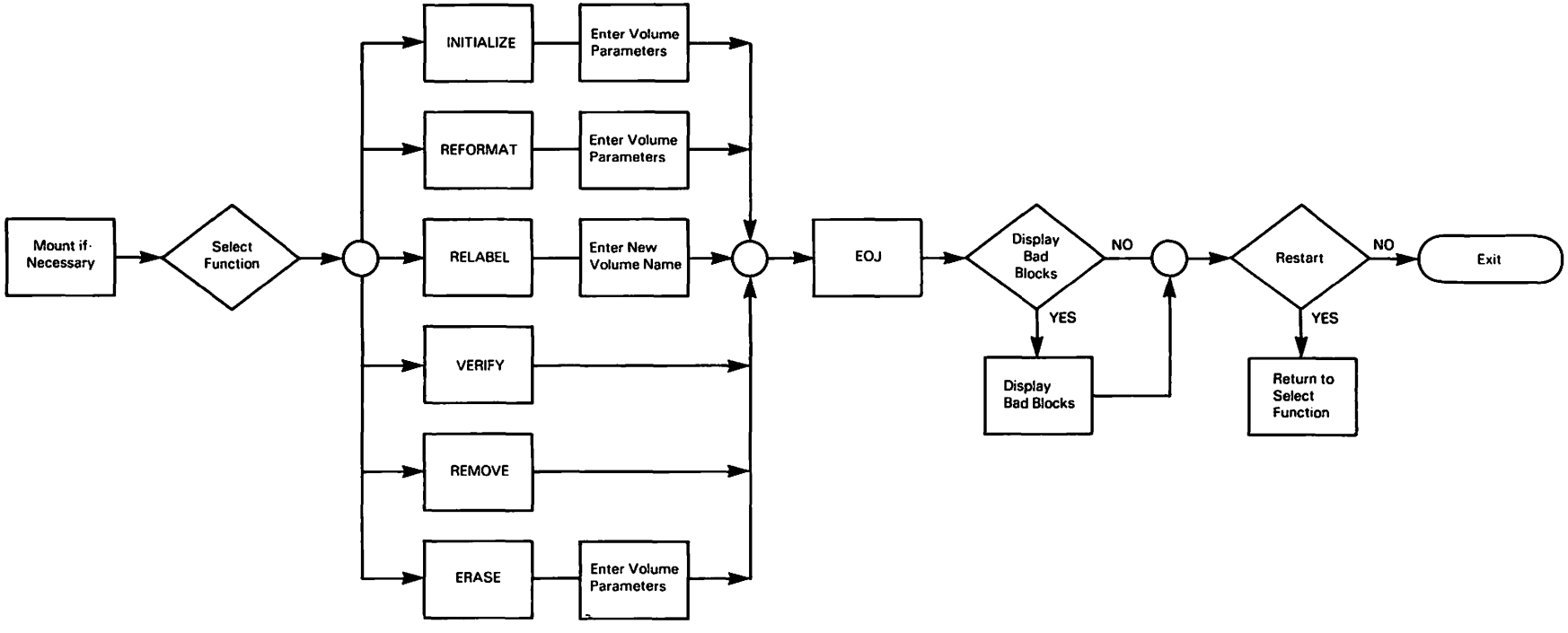


Figure 7-2. DISKINIT Processing

To obtain a brief, on-line description of each DISKINIT function, press PF13 (Display Information) from the DISKINIT Function Specification screen. Figure 7-3 shows the DISKINIT Function Definition screen.

```
*** MESSAGE 035  BY WT1M00
                    RESPONSE REQUIRED BY PROGRAM DISKINIT
                          TO ACKNOWLEDGE INFO
                     ACTIVE SUBPROGRAM IS DISKINIT

                 Press ENTER to return to the Function Screen.


                          FUNCTIONS PROVIDED BY DISKINIT
INITIALIZE - To initialize a new disk volume by analyzing the disk surfaces
             for reliable data storage and optionally creating a standard
             Volume Table of Contents and volume name for it.
REFORMAT   - To build a new, empty Volume Table of Contents on the volume.
RELABEL    - To change the name of a disk volume.
VERIFY     - To scan the surfaces of a disk volume and report any unreadable
             (bad) blocks that exist.
REMOVE     - To remove an unreliable block from a volume and replace it
             with a reserved block, if available.
ERASE      - To erase all data and build a new Volume Table of Contents.
                                    ***
Volume Set ID (VSID):          0 = Single Volume
                               1 = Root of a Volume Set
                     2 to 255 = Other members of a Volume Set
                     (Copyright, Wang Laboratories, Inc. 1985)
```

Figure 7-3. DISKINIT Function Definition Screen

7.3 MOUNTING A VOLUME

Mounting a volume on a system prepares a disk for use by system resources. In order to perform any disk maintenance functions on a disk, the disk volume must be mounted. You can use either of the following mount procedures to mount a volume for use:

- One way to mount a volume is to initiate a mount procedure. There are two steps in this type of mounting process; logical mounting and physical mounting. The first step, logical mounting, tells the system to ready a specified disk drive for physical mounting. You can logically mount a volume for exclusive or shared use. The second step, physical mounting, is physically placing the disk volume in the drive. When a disk or diskette is dismounted, the same steps are followed; the logical dismount precedes the physical dismount.
- Another way to mount a volume is to physically mount a volume on a disk drive. When you physically mount a volume, the operating system attempts to logically mount the volume for shared use. If the attempt fails (possibly due to a volume with the same label is already mounted), you must initiate the mount through the process that was previously described.

If the volume that you specified on the DISKINIT Function Specification screen is not yet mounted (e.g., if it is a new volume to be initialized), a message is displayed at the top of the screen that prompts you to perform a mount operation. Either respecify a currently mounted volume, along with it's volume set identification (VSID) number, and press ENTER to continue, or press PF5 to invoke the Mount Disk Volume function.

The VSID that you specify indicates whether you are mounting a single volume, a root volume, or secondary volume of a volume set. If you specify a 0 or blank, you are mounting a single volume. If you specify a 1, you are mounting the root volume of a volume set. If you specify any number from 2 to 255, you are mounting a secondary volume of a volume set. Refer to Section 7.3.1 if you are mounting a member of a volume set.

The Mount Disk Volume screen prompts you to enter the unit number of the disk drive on which the disk volume is to be mounted. This device number should be clearly marked on the device. If it is not, check the menu (containing the available devices and their device numbers), which is obtained through the Manage DEVICEs option (PF6) from the Command Processor menu. If no volume is currently mounted on the device specified, mount the new volume.

If another volume is mounted at the specified device address, you must dismount it before you can mount the new volume. A message is displayed that instructs you to dismount the volume (unless the volume is currently in use or mounted for exclusive access by another workstation).

A volume or volume set member is in use if one or more of its files is currently open. The Mount Disk Volume routine does not allow you to dismount a volume with open files. A volume that is mounted for exclusive use can be dismounted only from the workstation at which it was mounted. When the volume is successfully mounted, the system is automatically informed to resume processing of the DISKINIT function that you selected.

NOTE
You can mount new disks or diskettes for initialization through DISKINIT only. You cannot mount new disks or diskettes from the Operator's Console menu, from the Command Processor menu, or by a procedure.

7.3.1 Mounting a Volume of a Volume Set

If you mount a volume that is part of a volume set, you must specify the volume set name and the volume set identification (VSID). The VSID must be a unique number (from 1 to 255) for all volumes of the same volume set. The root volume always has a VSID of 1.

For example, the volume ALPHA can be a volume set that spans 15 disk volumes. Each volume would be named ALPHA but also have a unique numeric identifier (the VSID). The root volume has VSID 1, or ALPHA 1. You can assign any number from 2 to 255 to secondary volumes of a set.

If the ALPHA volume set has sequential VSIDs from 1 to 15, ALPHA 4 identifies a specific secondary volume in the ALPHA volume set. However, ALPHA 4 does not have precedence over any other secondary volume in the set. Logical volume sequence in a volume set is entirely up to you.

CAUTION
*You should assign VSID numbers sequentially within a volume set (although it is not necessary), **especially** when adding volumes to the volume set. If you try to assign an existing VSID number to a new member when that existing member is mounted, an error message appears. However, you can create a new member with a VSID that already exists in the volume set as long as the member with the duplicate VSID is not mounted at the same time that you are creating the new member. This situation, though, can cause unpredictable results. Therefore, you must be extra careful to assign to each new member a unique VSID.*

7.4 NONLABELED VOLUMES

Nonlabeled (NL) diskette volumes are used to transfer data to and from Wang VS Systems and other manufacturers' computers. NL diskettes are used by VS Word Processing, as well as by the FLOPYDUP and COPY2200 utilities. NL diskettes can also contain system dumps. A mounted volume is nonlabeled if

- It has not been previously initialized by DISKINIT.
- It was specified as nonlabeled (NL) when previously initialized.
- It was created on a Wang Word Processing System, a Wang 2200 System, or another manufacturer's computer system.
- It is a member of a volume set and is mounted on an operating system that predates Release 7.10.

If the mounted volume is nonlabeled, the program alerts you and asks whether it can proceed with the selected function. If you do not want to proceed, press PF1 to terminate processing. Press ENTER if you want to proceed with the selected function.

7.5 THE INITIALIZE FUNCTION

The Initialize function formats a new volume with sector control information and verifies that you can read and write to the disk. Initialize can also create a label and VTOC, allocate a dump file, and allocate a page pool.

The Initialize function first formats the volume then initializes it by writing a pattern of data over the existing data on the disk. If you specify the BRIEF option in the Passes field, DISKINIT writes a pattern over the existing data once (and verifies the blocks only once). The NORMAL option writes 8 patterns to the disk over the existing data. The purpose of the write operations is to ensure that the blocks on the disk are usable. If they are not usable, they are put on a bad block list.

Bad block addresses are listed in the VTOC as not available for use. Since an unavailable or bad block cannot be accessed, they are effectively eliminated from use. Bad blocks are replaced by good blocks if there is a reserve cylinder on the volume and if there are available blocks on that cylinder. If the volume does not have a reserve cylinder or all available blocks on the reserve cylinder have been used, you are informed that the volume is unusable.

NOTE
The reserve cylinder contains reserve blocks and is also referred to as the replacement pool. As a safeguard to the amount of data storage space on the volume, a block is allocated from the replacement pool when a block on the disk is unusable. However, once the pool is depleted, the volume is considered unusable, since subsequent bad block occurrences would shrink the logical size of the volume. Also, not all volumes contain a reserve cylinder because the probability of a bad block occurrence is small. This is especially the case with diskettes; i.e., the smaller the volume, the fewer blocks it has and the less it needs reserve blocks as a safeguard. For more information about the replacement pool, refer to Section 7.8.

DISKINIT initializes both single- and double-sided, soft-sectored diskettes and hard-sectored diskettes to Wang standard formats. Because all Wang standard soft-sectored diskette formats are double-density, you do not need to select a format. As a result, soft-sectored diskette initialization does not differ externally from other disk initializations. If you mount a double-sided, soft-sectored diskette, DISKINIT initializes it as a double-sided, double-density (DSDD) diskette. Single-sided, soft-sectored diskettes are initialized to single-sided, double-density (SSDD).

Figure 7-4 shows the DISKINIT Initialize Volume Definition screen. To initialize a volume, accept the default value for each field or specify new values for the fields and press ENTER.

```
***  MESSAGE 013  BY WT1M00

                INFORMATION REQUIRED BY  PROGRAM  DISKINIT
                          TO DEFINE INPUT
                     ACTIVE SUBPROGRAM IS DISKINIT

Please enter the following input parameters for Initialization.

Specified volume is SYSTEM on device 014  removable platter, VSID 0.

WARNING:  INITIALIZE function destroys the VTOC and all other data on volume!
          To terminate this function, press PF1.  Press ENTER to continue.

   NEWVOL   = ******          (Volume Name for initialized volume)
   VSID     = **0             (0 -Single Volume; 1-255 =Volume Set, 1 =Root)
   LABEL    = SL              (SL - Standard Label, NL - No Label)
   TOLERATE = NONE*           (NONE  - Smallest VTOC - No fault tolerance)
                              (CRASH - Medium VTOC - Tolerate system halt)
                              (MEDIA - Largest VTOC - Also tolerate bad media)
   VTOCSIZE = 0941            (Minimum of 4 page-size blocks)
   OWNER    = **************  (Owner identification for VOL1 label)
   DUMPFILE = NO*             (YES - Allocate system dump area)
   PAGEPOOL = NO*             (YES/NO - Allocate a pool for paging)
   PASSES   = BRIEF*          (BRIEF  - Only 1 pattern for quick initialize)
                              (NORMAL - All patterns for thorough initialize)
```

Figure 7-4. Sample DISKINIT Initialize Volume Definition Screen

The fields found on the DISKINIT Volume Definition screen are described as follows:

Field	Description
NEWVOL	Specify a name for the new volume. The default value is the value that you specified in the Volume field on the DISKINIT Function Specification screen.
VSID	Specify the volume set indentification number (0 – 255) for the new volume. The default value is retained from the VSID value that you specified on the DISKINIT Function Specification screen (Figure 7-2).
LABEL	Specify SL (Standard Label) or NL (Nonlabeled) as the type of label that you want placed on the disk. A volume can contain a standard label and a VTOC, or no label and no VTOC. The default value is SL. Volumes used in standard processing must have standard labels. You can use an NL diskette volume to exchange data between the VS and other computer systems or VS Word Processing Systems, or to receive FLOPYDUP output.

Field	Description
LABEL (cont.)	Volume Sets have a different VTOC structure than single volumes. Volume set members must be mounted as SL (standard label). If the volume to be initialized is a member of a volume set (VSID > 0), then specify SL in the Label field. **NOTE** *If you mount a volume set member on an operating system that predates Release 7.10, the volume is mounted as Nonlabeled. These precautions prevent corruption of the volume set member's VTOC by preventing a user from copying or deleting files from the volume set member.*
TOLERATE	Specify the level of fault tolerance (NONE, CRASH, or MEDIA) that you want the volume's VTOC to have. The default value is NONE. Fault Tolerance is discussed in Section 7.11. **NOTE** *Volume set volumes require a tolerance of MEDIA or CRASH.*
VTOCSIZE	Specify the size of the Volume Table of Contents. The default value of the VTOC size is computed by the system and assumes an average combination of average-sized program and text files, but the default value may not be appropriate for all applications. The default value can be overridden if you expect to make an unusually large number of entries to it. You are encouraged to modify the VTOC size, based on your experience with your applications. For example, you should enlarge the VTOC size if many relatively small files are to be created on the volume. The minimum size that you can assign to any disk is four blocks.
OWNER	Optionally specify the volume owner's name (up to 14 characters). This field defaults to lowercase characters. To use uppercase characters, use the SHIFT key.
DUMPFILE	Specify YES if you want to allocate a permanent dump file. The default value is NO. If you specify YES in the Dumpfile field, the DISKINIT Dump File Size Definition screen appears. The DISKINIT Dump File Size Definition screen and dump files are discussed in Section 7.12.
PAGEPOOL	Specify YES if you want to allocate a page pool. The default value is NO. If you specify YES in the Pagepool field, the DISKINIT Page Pool Definition screen appears. The DISKINIT Page Pool Definition screen and page pools are discussed in Section 7.13.
PASSES	Specify the depth of analysis to be performed. The Initialize function automatically verifies the accessibility of each block on the disk surface. This verification process can be done briefly (BRIEF option) if the volume is not expected to contain especially critical information, or it can be done more thoroughly (NORMAL option) if the integrity of the information to be stored on the volume is a matter of critical concern. Because new disks are the usual media for the Initialize function, NORMAL is the recommended option.

Before initialization occurs, the DISKINIT Bad Blocks screen (Figure 7-5) appears if the volume has been previously initialized and it contains bad blocks. The DISKINIT Bad Blocks screen enables you to specify how to manage any new bad blocks that this initialization encounters.

```
*** MESSAGE    028  BY WT1M00
                 INFORMATION REQUIRED BY PROGRAM DISKINIT
                         TO DEFINE BADBLKKL
                     ACTIVE SUBPROGRAM IS DISKINIT


The specified volume already contains a list of bad blocks.  You may choose
to simply retain the list, to add to it, or to build a new one.

            (4) - Retain the current bad block list.  Do no further
                  analysis for bad blocks.

            (6) - Retain the current bad block list and add to it
                  any new bad blocks found during this initialization.

            (8) - Discard the current bad block list.  Build a new
                  one from any bad blocks found during this initialization.


Press the appropriate PF key to perform the indicated action, or
press PF1 to return.
```

Figure 7-5. DISKINIT Bad Blocks Screen

The following actions occur to the disk or diskette when the initialization of the disk or diskette is performed, depending on the option that you select:

PF Key	Action
4	If you press PF4, DISKINIT constructs a new volume label and VTOC. No formatting or initialization is performed.
6	If you press PF6, DISKINIT retains the current bad block list, formats and initializes the disk or diskette, and adds any bad blocks to the current list. To ensure the most reliable disk or diskette, select this option.
8	If you press PF8, DISKINIT discards the current bad block list, formats and initializes the disk or diskette, and creates a new bad block list.

After you specify the fields and press ENTER, formatting and initialization begins. After initialization is complete, restart the program by pressing PF1, or terminate DISKINIT by pressing PF16. You can also press PF15 to display the bad blocks (if any).

7.6 THE REFORMAT FUNCTION

The Reformat function creates a new label and VTOC on a previously initialized disk volume. In creating a new VTOC, all pointers that access existing data are reset; the information in the files is no longer accessible. You cannot run the Reformat function on an uninitialized volume.

Volume set members (VSID > 0) must have a standard label (LABEL = SL) placed on the volume. If you mount a volume set member on an operating system that predates Release 7.10, the volume is mounted as Nonlabeled.

CAUTION

Do not use the Reformat function for removing sensitive or confidential data from a disk. The Reformat function changes the VTOC and therefore the accessing of data on the disk, but does not affect existing data on a disk. Use the Initialize or Erase function to write over a disk, thereby destroying existing data. If there is a question of security when reformatting a disk, erase it with the Erase function.

Since reformatting does not destroy the data in files when the VTOC is reformatted, existing information is destroyed when new information is written over the old. Writing over the data after reformatting is as transparent to an end user as writing onto a clean disk. After you select the Reformat function and enter the volume name to the DISKINIT Function Specification screen, the DISKINIT REFORMAT Option screen (Figure 7-6) appears.

NOTE

There is a difference between reformatting (through the Reformat function), which means to construct a new label and VTOC, and formatting (through the Initialize and Erase functions), which means to write out sector control information to the volume.

```
***  MESSAGE 013  BY WT1M00

                INFORMATION REQUIRED BY  PROGRAM  DISKINIT
                          TO DEFINE INPUT
                    ACTIVE SUBPROGRAM IS DISKINIT

Please enter the following input parameters for this functions:

Specified volume is SYSTEM on device 014  removable platter, VSID 000.
WARNING:  REFORMAT function destroys all current files on the volume!
          To terminate this function, press PF1.  Press ENTER to continue.
Extent limit values are ignored for Non-Root members of Volume Sets (VSID>1).
    NEWVOL   = ******          (New Name for the volume)
    VSID     = **0             (0 -Single Volume; 1-255 =Volume Set, 1 =Root)
    TOLERATE = NONE*           (NONE, CRASH, MEDIA  --  Fault tolerance level)
    VTOCSIZE = 0000            (Minimum of 4 page-size blocks)
    OWNER    = **************  (Owner identification for VOL1 label)
    DUMPFILE = NO*             (YES/NO - Allocate System dump area)
    PAGEPOOL = NO*             (YES/NO - Allocate a pool for paging)
    XTNTOPEN = 003             (Extent limit at file creation: 3-255)
    XTNTTOTL = 000000013       (Total Extents Limit: 13-255 [MVF: 999999999])
                               (For volume sets ONLY, 0 = Unlimited extents)
                    Copyright, Wang Laboratories, Inc. 1985)
```

Figure 7-6. Sample DISKINIT REFORMAT Option Screen

The fields found on the DISKINIT REFORMAT Option screen are described as follows:

Field	Description
NEWVOL	Specify a name for the new volume. The default is the value of the Volume field on the DISKINIT Function Specification screen (Figure 7-1).
VSID	Specify the volume set indentification number (0 – 255) for the new volume. The default value is retained from the VSID value that you specified on the DISKINIT Function Specification screen (Figure 7-1).

Field	Description
TOLERATE	Specify the level of fault tolerance (NONE, CRASH, or MEDIA) that you want the volume's VTOC to have. The default value is NONE. CRASH or MEDIA tolerance is required for volume set members. Fault Tolerance is discussed in Section 7.11.
VTOCSIZE	Specify the size of the Volume Table of Contents. The default value of the VTOC size is computed by the system and assumes an average combination of average-sized program and text files. The averages used are based on empirical system tests, and may not be appropriate for all applications. The default value can be overridden if you expect to make an unusually large number of entries to it. You are encouraged to modify the VTOC size, based on your experience with your applications. For example, you should enlarge the VTOC size if many relatively small files are to be created on the volume. The minimum size that you can assign to any disk is four blocks.
OWNER	Optionally specify the volume owner's name (up to 14 characters). This field defaults to lowercase characters. To use uppercase characters, use the SHIFT key.
DUMPFILE	Specify YES if you want to allocate a permanent dump file on a single volume. (Dump files are not supported for volume sets.) The default value is NO. If you specify YES in the Dumpfile field, the DISKINIT Dump File Size Definition screen appears. The DISKINIT Dump File Size Definition screen and dump files are discussed in Section 7.12.
PAGEPOOL	Specify YES if you want to allocate a page pool on a single disk or diskette volume. (Page pools are not supported for volume sets.) The default value is NO. If you specify YES in the PAGEPOOL field, the DISKINIT Page Pool Definition screen appears. The DISKINIT Page Pool Definition screen and page pools are discussed in Section 7.13.
XTNTOPEN	Specify the maximum number of extents that are allocated to a file when it is opened for creation. Extent limits for volume sets can be set only on the root volume. However, the number of extents that are allocated to a file applies to the whole volume set, regardless of the disk(s) on which the file resides. (Values set for secondary volumes are ignored.)
XTNTTOTL	Specify the maximum number of extents that are allocated to a file when additional allocation of extents is required. Values range from 13 to 255 extents for single volumes, 13 to 999,999,999 extents for volume sets. A value of 0 means an unlimited amount of extents for volume sets. Extent limits for volume sets can be set only on the root volume. (Values set for secondary volumes are ignored.)

After the fields have been specified, the program begins the reformatting procedure. When reformatting has completed, you can restart the program by pressing PF1 or terminate DISKINIT by pressing PF16.

7.7 THE RELABEL FUNCTION

The Relabel function enables you to rename a previously initialized disk volume by changing the label and gives you the option to allocate a page pool. Relabel does not place a new VTOC on the volume and does not destroy the information on the volume. If you select the Relabel function from the DISKINIT Function Specification screen, the DISKINIT RELABEL Option screen (Figure 7-7) appears.

Volume set members (VSID > 0) must have a standard label (LABEL = SL) placed on the volume. If you mount a volume set member on an operating system that predates Release 7.10, the volume is mounted as Nonlabeled.

A must

NOTE

For VS15, VS25, VS45, and VS65 systems, Relabel calculates an address for the bootstrap file, @MCBOOT@, in library @SYSTEM@ if the file currently resides on the volume. You need this file to perform an Initial Program Load (IPL). The address is necessary because the above systems' IPL software cannot read the volume's VTOC. The software uses the address to gain access to the bootstrap file, which can read the VTOC.

Therefore, it is recommended that you perform a Relabel operation on a volume from which you want to IPL if you have moved the bootstrap file (e.g., the bootstrap file can be moved during a Backup and Restore operation on a volume).

```
***  MESSAGE 013  BY WT1M00

                INFORMATION REQUIRED BY  PROGRAM  DISKINIT
                          TO DEFINE INPUT
                     ACTIVE SUBPROGRAM IS DISKINIT

Please enter the following input parameters for this function:

Function RELABEL ready to process.

Specified volume is SYSTEM on device 014  removable platter, VSID 0.
To terminate this function, press PF1.  Press ENTER to continue.

     NEWVOL   = ******      (New Name for the volume)

     PAGEPOOL = NO*         (YES/NO - Allocate a pool for paging)

     XTNTOPEN = 003         (Extent limit at file creation: 3-255)

     XTNTTOTL = 000000013   (Total extents limit: 13-255 [MVF: 999999999])
                            (For volume sets only, 0 = Unlimited extents)
```

Figure 7-7. Sample DISKINIT RELABEL Option Screen

The fields found on the DISKINIT RELABEL Option screen are described as follows:

Field	Description
NEWVOL	Specify a name for the new volume. The default is the value of the Volume field on the DISKINIT Function Specification screen.
	CAUTION *Changing the name of a volume set member causes data to be lost unless the names of* ***all*** *volume set members are changed to the same name at the same time. Changing VSID numbers is not allowed in the Relabel function because data would be lost on the volume. Use Reformat if you want to change the name or the VSID (or both), and if you do not care about the data on the volume.*
PAGEPOOL	To allocate a page pool on a single volume, specify YES in the Pagepool field. (Page pools are not supported for volume sets.) If the current volume contains a page pool, the default value of the Pagepool field is YES (otherwise, the default value is NO). If you specify YES in the Pagepool field, the DISKINIT Page Pool Definition screen appears. The DISKINIT Page Pool Definition screen and page pools are discussed in Section 7.13.
XTNTOPEN	Specify the maximum number of extents that are allocated to a file when it is opened for creation. Extent limits for volume sets can be set only on the root volume. However, the number of extents that are allocated to a file applies to the whole volume set, regardless of the disk(s) on which the file resides. (Values set for secondary volumes are ignored.)
XTNTTOTL	Specify the maximum number of extents that are allocated to a file when additional allocation of extents is required. Values range from 13 to 255 extents for single volumes, 13 to 999,999,999 extents for volume sets. A value of 0 means an unlimited amount of extents for volume sets. Extent limits for volume sets can be set only on the root volume. (Values set for secondary volumes are ignored.)

You can terminate the Relabel function without renaming the volume by pressing PF1. When relabeling has completed, you can restart the program by pressing PF1 or terminate DISKINIT processing by pressing PF16.

7.8 THE VERIFY FUNCTION

The Verify function reads a disk volume to ensure that all blocks on the disk surface can be addressed and read. Verify does not alter or destroy any information on the volume.

Unlike the Initialize function, which reads and rewrites each block, the Verify function only reads. It is therefore possible for a block to be bad for a write operation and not be reported by this function. Any block that cannot be read is a bad block. The Remove function, detailed in Section 7.9, enables you to replace a bad block with a reserve block from the replacement pool.

The replacement pool is a reserve of blocks that you can allocate as the need arises. Once the pool is depleted, and there are still bad blocks that have not been replaced, the volume is unreliable.

If you specify the Verify function from the DISKINIT Function Specification screen and press ENTER, the volume name and device number is displayed on another screen. Press ENTER to begin the verification process. DISKINIT then informs you of how many bad blocks were found and in which files they were located. If the bad block(s) are in the VTOC, you are informed that the volume is not suitable for use.

The addresses of each bad block, and (if a bad block occurs in a file) each file name, are displayed. Verify does not replace a bad block, it only informs you that one exists. Use the Remove function to replace bad blocks. After verification is complete, you can restart DISKINIT by pressing PF1 or terminate DISKINIT by pressing PF16.

7.9 THE REMOVE FUNCTION

The Remove function enables you to remove a bad block from the volume and replace it with a reserve block if, and only if, one is available (otherwise, the bad block remains and you are informed that the replacement pool is depleted). The system removes a bad block from use by listing its address in the bad block list and removing it from the available block list. It is then unavailable and the system no longer recognizes it when searching for disk space on which to read or write information. This effectively eliminates the block from use. You can use the Remove function only with disk drives that have cylinders that are (and can be) reserved for replacement of bad blocks.

The Initialize function also flags bad blocks, while validating all blocks on a volume, but sometimes an initially validated block shows input/output (I/O) errors after a period of use. If the I/O error log shows repeated errors in a particular block, or if the Verify function discloses the bad block, the block is flagged as unreliable and can be removed from the volume by using the Remove function. A number of extra blocks are reserved on some disk types to serve as replacements for bad blocks. If a reserved block is available, a removed block is replaced by a reserved block. The location of a bad block determines how you should replace it.

- If a bad block is located in the VTOC, you may lose some data. You should back up the data, reinitialize the volume, and restore what data you can.
- If a bad block is located within a file, you should back up the file, delete the original, remove the block, and restore the file. Since the block may have contained file data, only part of the file may be recoverable.
- If a bad block is listed as an available block, you can remove it and replace it with one from the reserve pool.

When you specify the Remove function and the volume name on the DISKINIT Function Specification screen and press ENTER, DISKINIT displays another screen showing the function, volume name, and device number. The screen also displays a caution that this function permanently removes bad blocks from the volume. If you remove a block that is not bad, it can be recovered through an Initialize operation. At this point, DISKINIT enables you to terminate the function before any blocks are removed by pressing PF1.

To continue the Remove function, enter the block number into the Block field and press ENTER. After the Remove function has completed, you can restart the program by pressing PF1 or terminate DISKINIT by pressing PF16. You can also display any bad blocks by pressing PF15.

7.10 THE ERASE FUNCTION

The Erase function essentially performs the same function as the Initialize function except that Erase conforms to the standards outlined in the *Department of Defense Industrial Security Manual*. The main purpose of the Erase function is to render all data on a disk volume unreadable. The DISKINIT Erase Volume Definition screen is the same as the DISKINIT Initialize Volume Definition screen (Figure 7-4) except that you cannot specify the number of passes for Erase. Refer to Figure 7-4 for a description of the fields.

Both Initialize and Erase use the same initial format (sector control information) pass and also use the same initial pattern on the first initialization pass to write over all sectors on a disk. Erase then uses the following patterns:

1. A pattern of binary zeros (0) are written and verified over the entire disk.
2. A pattern of binary ones (1) are written and verified over the entire disk.
3. A random, nonrepeating pattern is written and verified over the entire disk.

Write commands are issued to all sectors, including those already in the bad block list. All write commands are verified by the Erase function. Any sectors which cannot be successfully overwritten will be reported. When Erase has completed the write over passes, a standard VTOC is built (if SL was specified for the volume label). The bad block list is updated and restored and any sectors with failed verify/write commands are added to the bad block list. If the disk has a diagnostic cylinder, it is also erased and reinitialized. During Erase processing, the message "Erasing cylinder 050 of 077 on pass 1 of 4" appears on the workstation screen (sample numbers are used here) to verify erasure of cylinders.

The Erase function destroys all data on a disk. Furthermore, disks that have been cleared by the Erase function need not be reinitialized because an erased disk contains a standard VTOC (if SL was specified for the volume label).

Volume set members (VSID > 0) must have a standard label (LABEL = SL) placed on the volume. If you mount a volume set member on an operating system that predates Release 7.10, the volume is mounted as Nonlabeled.

CAUTION

Bad blocks listed in the bad block list after the Erase operation ***may*** *not have been erased. If some or all of the erase patterns fail to write to the bad blocks on the disk, the data in those bad blocks may not be destroyed. (Failed write operations to the bad blocks are reported to you upon completion of the Erase function.) As a consequence, if you discover bad blocks in the sensitive areas of your volume after the erasure, you may need to physically destroy the disk to ensure data security. If you are sure that none of the bad blocks are in the data sensitive areas, then simply reuse the volume.*

7.11 FAULT TOLERANCE

Fault tolerance is a feature that protects a volume's VTOC from damage by copying the physical representation of the VTOC blocks. DISKINIT enables you to format a volume's VTOC with one of three levels of fault tolerance. The first level (None) is actually not tolerant (that is, no duplicates of the VTOC blocks are made). The second level (Crash) makes two copies of each VTOC block, and the third level (Media) makes four copies of each VTOC block on separate cylinders. You can specify fault tolerance through the Initialize or Reformat functions.

7.11.1 Crash Tolerance

Crash tolerance provides protection against VTOC damage caused by system failure in the middle of a VTOC operation. Within the VTOC, blocks are physically allocated into pairs on the volume. Each block is copied onto a second adjacent block. The twin blocks are updated alternately which results in one of the two blocks serving as a backup copy for the other. A bit map block at the head of the VTOC indicates which block is currently the update version. A copy of the bit map block is maintained in an adjacent block.

7.11.2 Media Tolerence

Media tolerance provides protection against mechanical disk failures, such as power failures or defective disk media. Mechanical failures can affect entire disk cylinders.

A disk drive rotates a mounted disk (made of several disk platters stacked on top of each other called a disk pack) at a high speed, while a read/write head "floats" on a cushion of air, about 30 micro-inches above each platter surface. Information is stored on each disk surface (except the very top and the very bottom surfaces of a disk pack, to prevent damage) in hundreds of concentric circles called tracks. Each track is divided into segments called sectors. A set of tracks in a vertical line on the platters of a disk pack is called a cylinder. (For example, each surface of a platter has a Track 0 and all of the Track 0s combined make up Cylinder 0.)

Any direct physical contact between head and disk is known as a head crash, and can have disasterous results for stored data and the disk itself. Media tolerance ensures that each block within the VTOC is allocated four physical blocks on the volume, giving you two complete crash-tolerant VTOCs. To protect against an event such as a head crash, each of these crash-tolerant VTOCs is located on separate cylinders. The volume label, bad block list, and bit map block are duplicated, as well.

The first version of the VTOC begins at Block 4, Cylinder 0, occupying the rest of Cylinder 0. If necessary, it also occupies Cylinder 2, and as many more even-numbered cylinders as needed. The second version of the VTOC begins in Block 4, Cylinder 1, and continues on as many odd-numbered cylinders as needed. The first blocks in Cylinders 0 and 1 contain the volume label, the bad block list, and the bit maps (version 1 and 2) and are not technically part of the VTOC.

For example, if a VTOC requires three cylinders, the first version of the VTOC (beginning with Block 4 of Cylinder 0) occupies Cylinders 0, 2, and 4; the second version of the VTOC (beginning with Block 4 of Cylinder 1) occupies Cylinders 1, 3, and 5. A media-tolerant VTOC occupies at least two complete cylinders.

Media tolerance virtually eliminates the possibility of losing files due to VTOC errors. However, media tolerance requires more than four times the disk space and four times the number of VTOC write operations as a VTOC without fault tolerance. In addition, a bit map read operation is required when the VTOC is first accessed; the bit map block, once read, is retained in memory. Media tolerance is recommended whenever maximizing disk storage space is not an overriding consideration and when data is of paramount importance.

7.12 DUMP FILE ALLOCATION

A dump file is a storage area into which the contents of main memory can be placed for analysis. DISKINIT enables you to allocate a dump file through the Initialize, Reformat, or Erase options. A dump file is used in conjunction with the system dump facility for system-initiated and snapshot (user-initiated) dumps.

NOTE
Dump files are not supported on volume sets.

If a dump file exists on a disk volume, the dump file receives the contents of main memory after you command the system to execute a dump. The system automatically switches to Control mode when the system encounters a problem that prevents it from proceeding with normal operations. At this point, a system dump can be taken. For more information about system and snapshot dumps, refer to Chapter 19. For a complete explanation of system and snapshot dumps, refer to the *VS System Operator's Guide* and the individual processor handbook for your system.

DISKINIT allocates the dump file in a single contiguous extent and places a pointer in the volume label. It is created as a consecutive file with 2048-byte (2K) records. If you enter YES in the Dumpfile field on the Initialize or Reformat screen, DISKINIT displays the Dump File screen. The Dump File screen requests you to specify, in kilobytes (KB), the size of the dump file. The dump file must be large enough to contain the main memory of the central processor for which you intend to use it (otherwise, no dump can be made to the file). The maximum size of the dump file is 65,536 KB.

7.12.1 Allocating the Dump File

If you specify YES in the Dumpfile field of any DISKINIT function, the DISKINIT Dump File Size Definition screen (Figure 7-8) appears.

```
*** MESSAGE    028  BY WT1M00
                 INFORMATION REQUIRED BY PROGRAM DISKINIT
                         TO DEFINE DUMPFILE
                      ACTIVE SUBPROGRAM IS DISKINIT


   Please specify the size of the preallocated system dump file.
   The size of the file should correspond to the size of main memory for any
   CPU on which you intend to use this disk.  Enter a size of zero (0) if
   you do not want to allocate a dump file at this time.

              Size of pre-allocated dump file:   SIZE     = ***** Kilobytes




                       Press ENTER to continue or PF1 to return.
                       (Copyright, Wang Laboratories, Inc. 1985)
```

Figure 7-8. DISKINIT Dump File Size Definition Screen

If you specify a dump file size of zero, no dump file is allocated. The field value defaults to the previously established file size during a Reformat operation on a volume that already contains a dump file.

You set the size of the dump file when you initialize or reformat the disk. If the size that you specify is an odd value, DISKINIT rounds it up to the next 2K value. The dump file is given the name @CMDUMP@ in the library @SYSDUMP by DISKINIT. You cannot rename or delete the file @CMDUMP@ or the library @SYSDUMP.

If you subsequently use the Reformat function, the address of the dump file (if one exists) is preserved. If the dump file does not exist, DISKINIT stores ASCII blanks (hexadecimal 20) in the dump file address field. (The address for adump file is maintained in the volume label.)

7.13 PAGE POOL ALLOCATION

DISKINIT enables you to allocate a page pool through the Initialize, Reformat, Relabel, or Erase functions. There are three factors to consider when allocating a page pool: the physical size of the page pool (and it's location), the maximum modifiable data area sizes of the users that will be using the page pool, and the commitment ratio of the page pool. The following sections describe the concept of paging and the process of allocating a page pool.

NOTE
Page pools are not supported on volume sets.

7.13.1 System Processing and Paging

When the operating system runs a program, it must retrieve the program from the disk and load it into main memory. (System tasks are also programs and are consequently loaded in the same way.) The operating system loads program code in 2-KB portions called pages. Main memory is partitioned into 2-KB areas (called page frames) to receive these pages. The process of loading pages into and out of main memory is called "paging" (paging in and paging out). Certain page areas are dedicated to operating system routines. Other pages may contain parts (or all) of any other programs or data.

When tasks compete for main memory processing space, data that is currently in main memory must be written over in order to allow other tasks to access the main memory processing space. If the data to be written over is modifiable and it has been modified since being "paged" into main memory, it must first be copied into another location before its space in main memory is taken.

When you allocate a page pool, you specify an area on the disk where modified pieces of code are placed if main memory is needed for other tasks. The page pool is a temporary location for pages that have been modified during the processing of a task or program.

Main memory can hold a number of pages which do not have to be from the same program nor does the program have to be limited to the size of main memory. (The number of pages your system can hold depends on the size of main memory in your system.) Stated simply, this is how simultaneous processing and virtual storage are implemented. When a program is submitted to the operating system to be run, the operating system finds the program on the disk and attempts to load as many pages of the program as it can (depending on system processing demands) into main memory.

7.13.2 Page Faults

In main memory, the operating system processes a program page by page. As code is executed, the code may reference other pages needed to continue processing. A page fault occurs when a page not already in main memory is required to continue processing.

When a page fault occurs, the region and page number of the next page are supplied to the operating system so that it can bring that page into main memory to be processed. Tasks compete for main memory processing space; when a new page is "paged in", it is laid over a page that the operating system has determined to be the best candidate to be removed from main memory. This means that the data in that page frame is removed but it can be retrieved from data storage again if it is needed. If the page that is to be removed is a modified page of code, that page is placed into the page pool (if page pooling is enabled; otherwise, the page is placed into a paging file). If the removed page is not a modifiable page (or if it is a modifiable page but has not been modified since it has been in main memory), it is written over. If the data is subsequently needed, it can be retrieved from the original address on the disk.

7.13.3 The Modifiable Data Area Space

The modifiable data area contains all areas that can be modified in all user-run programs. The system administrator assigns the modifiable data area size in virtual memory (through the SECURITY utility) for each user. The modifiable data area size is computed by using the data segment size requirements of the applications that each user intends to use. Many users need only a fraction of the modifiable data area space that has been allocated. Consequently, page pools can be smaller than the total modifiable data areas of all users because pages from the page pool are only assigned as necessary to carry out a user's task.

7.13.4 The Commitment Ratio

The **commitment** of a page pool is the running total of the maximum allowed modifiable data area sizes of the users who are logged on and assigned to the pool (that is, it is the sum of the modifiable data areas of all current users of the pool). The **commitment ratio** is expressed as a percentage and is the total modifiable data area space of all users of the pool divided by the size of a page pool.

Since tasks normally use less than their maximum modifiable data area space, a commitment ratio is used to allow additional tasks to be assigned to a pool so that disk space is not wasted. The commitment ratio applies to all active page pools on the system. Since commitment ratios are only used to increase the commitment, setting ratios to less than 100% is not allowed. The valid range for a commitment ratio is 100 to 999. The default commitment ratio is 400%.

A commitment ratio of 100% on a system indicates a page pool space that can accommodate all allocated modifiable data area spaces. However, a risk is assumed when commitment ratios of over 100% are given to page pools. Refer to the *VS System Operator's Guide* for information about setting the commitment ratio.

NOTE
*You should **initially** set the commitment ratio low (close to 100%) and check the peak usage factor (through POOLSTAT) after a truly representative workload has run. Peak usage is reset only at Initial Program Load (IPL), so you should let the system run long enough (days) to ensure that the system has encountered its "worst case" load.*

If peak usage is significantly lower than 100% (50%, for example), then you can safely raise the commitment ratio. The higher commitment ratio reduces the number of individual paging files created by the system, and thereby saves disk space; or if **all** tasks were previously assigned to the page pool, the higher commitment ratio enables you to reduce the physical size of the page pool (or assign more tasks to the same page pool), which also saves disk space.

For many system environments, commitment ratios of 200%, 250%, or even higher are safe. Ratios that are greater then 500% are generally safe for only the most unusual system environments.

The commitment ratio is determined by the system administrator when the page pool is assigned. The greater risk of potential overflow also yields the greater disk space savings by reducing the size of your page pool. The balance between the risk and the savings depends on the actual use of your system.

7.13.5 Page Pool Commitment

To illustrate the concept of a page pool commitment, assume that your system has a single, 10-MB page pool allocated through DISKINIT and enabled through the Operator Console. Your system has 10 users, each of whom has been allocated 1MB of modifiable data area space to their user IDs. (Users are allocated modifiable data area space by the system administrator through the SECURITY utility; refer to the *VS System Administrator's Reference* for more information). The combined modifiable data area space for the 10 users is 10 MB. All 10 users can run their applications safely to the maximum limit of their modifiable data area size.

If you added three new users to your system (assuming each is also allocated 1 MB of modifiable data area space), the sum of the 13 users' modifiable data area space is now 13 MB. If the page pool's commitment ratio were 100%, then the three new users' tasks would not be assigned to this page pool but instead the tasks would be assigned to paging files of 1 MB each. If the page pool's commitment ratio were 200%, however, the tasks would be assigned to the pool since the 200% ratio enables assignments of tasks **until** the sum of the combined users' modifiable data area maximums exceeds 20 MB (200% of the allocated page pool space).

Assigning a 200% commitment ratio to a 10-MB page pool presents a problem **only** if the users use more combined modifiable data area space than 10 MB (the size of the page pool). Since each users' tasks normally uses less than their maximum modifiable data area space during system operation, there is a low probability that the page pool will overflow in this case. However, a risk of overflowing the page pool does exist. Through an analysis of system use, you can determine a relatively safe risk for committing an amount of memory to a page pool.

CAUTION

If tasks have paged out so many pages that the physical use of the pool to which they are assigned reaches 100%, the next time a task attempts to page out pages to the pool, the pages become frozen in memory. This happens because there is no physical room in the pool for the pages. If program completions do not reverse this situation, more pages are frozen in memory and performance is significantly degraded.

Warning messages appear on the operator console when the page pool reaches a capacity of 75%, 87.5%, and 100%. If these messages appear, you should take action to reduce the risk of page pool overflow. For information about the page pool warning message, refer to the VS System Operator's Guide.

The safest commitment ratio is 100%, which prevents any possible page pool overflow. However, a 100% commitment ratio can prevent more users from using the page pool and valuable disk space may be wasted if users use only a fraction of their modifiable data area space during processing.

When a 200% commitment ratio is assigned to a page pool, the risk is assumed that half (or less) of each users' modifiable data area space (averaged over the user population) is actually needed to accommodate paging for user tasks; if less than half of the combined modifiable data area is used, there is no page pool overflow and the risk was successful.

7.13.6 Fully Committed Page Pools

If a page pool is available, tasks can be assigned to the page pool until it is fully committed. A fully committed page pool occurs when the users' combined modifiable data area space exceeds the size of the physical page pool times the page pool commitment ratio (for example, when the combined modifiable data area size exceeds twice the physical size of a page pool that has a commitment ratio of 200%).

If there is more than one volume containing a page pool, the system assigns a new task to the pool with the least tasks. After a new task is assigned to a page pool, the system then commits all modifiable data area pages that the new task requires for operation from that pool. If all of the page pools are fully committed, or if there are no volumes enabled with page pools, then a paging file is established in the @SYSPAGE library for the new task. The system may place the file anywhere on the disk and spreads paging files among the volumes that have been enabled for paging and that have available space.

7.13.7 Allocating a Page Pool

A page pool is given the file name @POOL@ and is located in the library @SYSPOOL. You specify the relative location of the @POOL@ file on the disk. If you specify YES in the PAGEPOOL field of any DISKINIT function, the DISKINIT Page Pool Definition screen (Figure 7-9) appears.

```
*** MESSAGE    028  BY WT1M00
                 INFORMATION REQUIRED BY PROGRAM DISKINIT
                             TO DEFINE PAGEPOOL
                       ACTIVE SUBPROGRAM IS DISKINIT

      Please specify the size and location of the user paging pool.
The size of the pool should be based on the Total Modifiable Data Area
(Segment 2) sizes of all tasks which may use this disk for paging.
      The location of the paging pool in relation to the VTOC may affect
system performance.
      Enter a zero (0) if you do not want to allocate a paging pool at
this time, otherwise the minimum page pool size is 2048 Kilobytes.

 Size of paging pool:                          SIZE     = ***** Kilobytes

 Location of paging pool relative to VTOC:   LOCATION = 0      (0 - 9)
                      ( 0 = Nearest VTOC       )
                      ( 9 = Farthest from VTOC )

                   Press ENTER to continue or PF1 to return.
                   (Copyright, Wang Laboratories, Inc. 1985)
```

Figure 7-9. Sample DISKINIT Page Pool Definition Screen

Page Pool Size

Through an analysis of the use of your system and the sum of the combined modifiable data area sizes assigned to users for their applications, you determine how much disk space to allocate for a page pool (not to exceed 32 MB).

To properly adjust the amount of disk space used for paging on your system, you should monitor the net effect of the factors (total modifiable data area, page pool size, and peak usage) that affect page pool use through the POOLSTAT utility. Allocate a large amount of space for the page pool and then reduce it as the demands on the system resources become clear. For example, if you determine that ten users, each with 1 MB modifiable data area, consistently use only 33% of their combined modifiable data area space (3.3 MB), then you can feel reasonably safe in reducing the page pool size to 4 MB. If you kept the 10-MB page pool with only 33% peak use, 6.7 MB of disk space would not be used.

Page Pools Versus Paging Files

The advantage of page pools over paging files is that they keep the modifiable data areas of all tasks contiguous, which can reduce seek time and significantly improve system throughput. The alternative to a page pool is discrete paging (that is, paging files). Paging with discrete pages may impair throughput because the pages in the paging file may not be contiguous. If pages are located in different locations on the disk, references and seek operations from one location to another can cause the system to use valuable time searching for and accessing these locations. By keeping the paging areas in a pool of contiguous space, seek time can be reduced, which can be of special benefit for smaller systems. Page pooling also reduces the amount of disk space that is used for paging.

If page pooling is not enabled, or if there is not enough space to accommodate the page pool sizes and maximum commitment ratios, then paging files are created.

7.13.8 Specifying Page Pool Size and Location

The size of a page pool can range from 2 KB to 32 MB (32,767 KB). If the volume already contains a page pool, the default size is the size of the current page pool. If you specify a size of zero, DISKINIT does not create a page pool.

NOTE

It is better to overestimate the size of the page pool than to underestimate it. You cannot easily increase the size of the page pool at a later time because files may have since been allocated around the page pool and the page pool must be allocated in a single extent. Use the Relabel function to modify the page pool size without destroying data.

Page Pool Location

If you allocate a page pool, you must also specify an integer between 0 and 9 for the location of the page pool relative to the VTOC. A value of 0 indicates a location nearest to the VTOC; 9 indicates a location farthest from the VTOC. If a volume already contains a page pool, the default location is the location of the current page pool. You should locate the page pool near the most active areas of the volume. For example, if the VTOC is the most active part of the volume, specify 0 as the relative location. This reduces seek time and increases throughput.

DISKINIT allocates the page pool in a single extent. If there is not sufficient space to do this, DISKINIT displays a message and allocates as large a page pool as possible. Because DISKINIT looks for an extent in the specified location, you may want to specify another location relative to the VTOC in order to allocate a page pool of the specified size.

If you cannot expand your current page pool in its location because files have since been allocated around it, and you must increase the size of your page pool to accommodate system tasks, you have three options:

1. You can reallocate the page pool to an area with enough space (if you have any contiguous space large enough). This could slow throughput if you have to locate your page pool away from the most active areas of memory.
2. You can run DISKINIT on a separate disk volume and assign the larger page pool, then copy the data from the current disk to the new disk.
3. Back up the disk to tape, initialize the disk volume, assign a larger page pool to the disk volume, then restore the data.

You should **initially** allocate a page pool large enough to handle the combined modifiable data area sizes and set the commitment ratio close to 100%. Check the peak usage factor after a representative workload has run. Let the system run long enough (days) to ensure that the system has encountered its "worst case" load. At that time, if peak usage is significantly lower than the actual combined modifiable data area sizes, then you can safely raise the commitment ratio.

The higher commitment ratio reduces the number of individual paging files created, freeing up disk space and probably increasing the rate of disk paging I/O. Or, if all tasks were previously assigned to the page pool, the higher commitment ratio allows you to reduce the size of the page pool (or allow more users to logon) with the Relabel function, which also saves disk space.

7.13.9 Estimating the Page Pool Size

The following formula can help you estimate the initial size of a page pool (in Megabytes):

PAGE POOL SIZE = [5 + (1/2 * AVERAGEMDA * MAXUSERS)] / VOLUMES

The formula variables are described as follows:

Variable	Value
AVERAGEMDA	This number is the average modifiable data area (user-modifiable memory) size, in MB, that is currently used in your system. Use the default modifiable data area size specified in GENEDIT or, if adjustments have been made for specific tasks or user IDs, calculate the actual average. When in doubt, overestimate the value.
MAXUSERS	This is the number of tasks (interactive and noninteractive) that are supported during a peak period.
VOLUMES	This is the number of volumes to be enabled with page pools.

The variable 1/2 (in the formula) reflects a 200% commitment ratio. If you were to commit 400% to a page pool, the variable would be 1/4; 700% would equal 1/7, and so on. This should be reflected in your formula when estimating your page pool size.

This formula includes a 5-MB padding factor to accommodate system tasks and to provide a margin of safety against page pool overflow. The 5-MB padding factor may be larger than your system tasks warrant. If you are not running communications, a more appropriate padding factor may be 2 MB and should also be reflected in your formula. The padding factor can be adjusted to fine tune your system's performance. Estimating the size of a page pool is of particular importance for small systems where system resources are more restricted than those of larger systems.

In the following example, a 200% commitment ratio has been assigned, most tasks (on average) use 512 KB (0.5 MB) of the modifiable data area, there are 60 tasks at peak usage, and one volume is enabled for paging:

PAGE POOL SIZE = [5 + (1/2 * 0.5 MB * 60)] / 1 = 20-MB page pool

After you establish a page pool, and monitor its activity, you can reduce its size through the Relabel function of DISKINIT to save additional space if actual use is light, or you can increase the size of the page pool to provide a satisfactory margin of safety if actual use nears the pool's capacity. You can use the POOLSTAT utility to monitor the use of a system's page pools. (Refer to Chapter 20 of this manual for more information on POOLSTAT.)

Tables 7-1 and 7-2 display a list of various system and communication tasks and their current data segment size requirements. You can use these tables to help you to estimate your page pool size requirements.

NOTE
For a complete list of minimum modifiable data area size requirements, refer to the VS Operating System Release 7.10 Customer Software Release Notice.

Table 7-1. System Tasks Data Segment Size Requirements

Task	Size Requirement
@TSKMGR@	1 MB
@PRTTSK@	280 KB
@SYSTSK@	1 MB
@SHARER@	1 1/4 MB (or 2 MB, if more than 100 buffers are needed)
WP printer task	192 KB (only if WP printers are in your configuration)
Total System Tasks:	
Without printers:	3 1/4 MB or 4 MB
With printers:	3 5/8 MB or 4 3/8 MB

Table 7-2. Communications Tasks Data Segment Size Requirements

Task	Size Requirement
@SESMGR@	1 MB
@FTMTSK@	1 MB
Total Communications Tasks:	2 MB

NOTE
Communications tasks data segment size requirements only apply if your system has the communications components in the system configuration.

7.14 A SAMPLE DISKINIT PROCEDURE

You can control DISKINIT processing through the VS Procedure language. The DISKINIT utility obtains information through GETPARM requests. The VS Procedure language MOUNT and DISMOUNT statements can provide the logical requirements of any necessary mounting or dismounting of initialized volumes.

Because you cannot embed the Procedure language MOUNT and DISMOUNT statements in a RUN, ENTER, or DISPLAY sequence, all Procedure language MOUNT and DISMOUNT operations must be performed prior to or following automated DISKINIT processing. A complete list of DISKINIT GETPARMs is provided in Appendix A. Refer to the *VS Procedure Language Reference* for details about Procedure language syntax.

The following procedure relabels a single disk volume. When you run the procedure, DISKINIT relabels the volume and allocates a page pool. The page pool has a size of 16,384 KB, and its location is 0 (i.e., nearest to the VTOC).

```
PROCEDURE
    RUN DISKINIT
        ENTER FUNCTION FUNCTION = RELABEL, VOLUME = ZENITH,
            VSID = 0
        ENTER INPUT PAGEPOOL = YES
        ENTER PAGEPOOL SIZE = 16384, LOCATION = 0
        ENTER EOJ 16
RETURN
```

CHAPTER 8
THE DISPLAY UTILITY

8.1 INTRODUCTION

The DISPLAY utility enables you to examine the contents of any disk file at your workstation. The utility provides information about the type of file organization and the number and size of records in the file. For indexed files, DISPLAY also indicates the location of the index key and its size and the number of alternate indexes. You can perform the following functions with the DISPLAY utility:

- Display a file in the American Standard Code for Information Interchange (ASCII) or hexadecimal code
- Find and display an indexed record by key value
- List the index descriptions for an alternate indexed file and optionally change the access path
- Find and display a record in a consecutive file by record, line, or block number
- Find and display a record in a relative file by record, line, or block number
- Find and display a record by text string
- Produce a printed copy of all or part of a file in Record or Block access mode

8.1.1 Record Access Versus Block Access

A file can be accessed by record or block. Record access displays the file in records (as opposed to blocks) in two different formats: report-oriented format and record-oriented format. Report-oriented format displays consecutive or relative files on a line by line basis; that is, each record is displayed on one line. The DISPLAY utility cannot present indexed files in report-oriented format. Record-oriented format displays consecutive, relative, or indexed files on a record by record basis.

Block access displays an exact image of the file layout as it exists. The file is displayed in 2-KB (2048-byte) blocks in physical block sequence. Data blocks and index blocks are displayed for an indexed file.

To run the DISPLAY utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify DISPLAY in the Program field and press ENTER. Figure 8-1 shows an overview of DISPLAY processing.

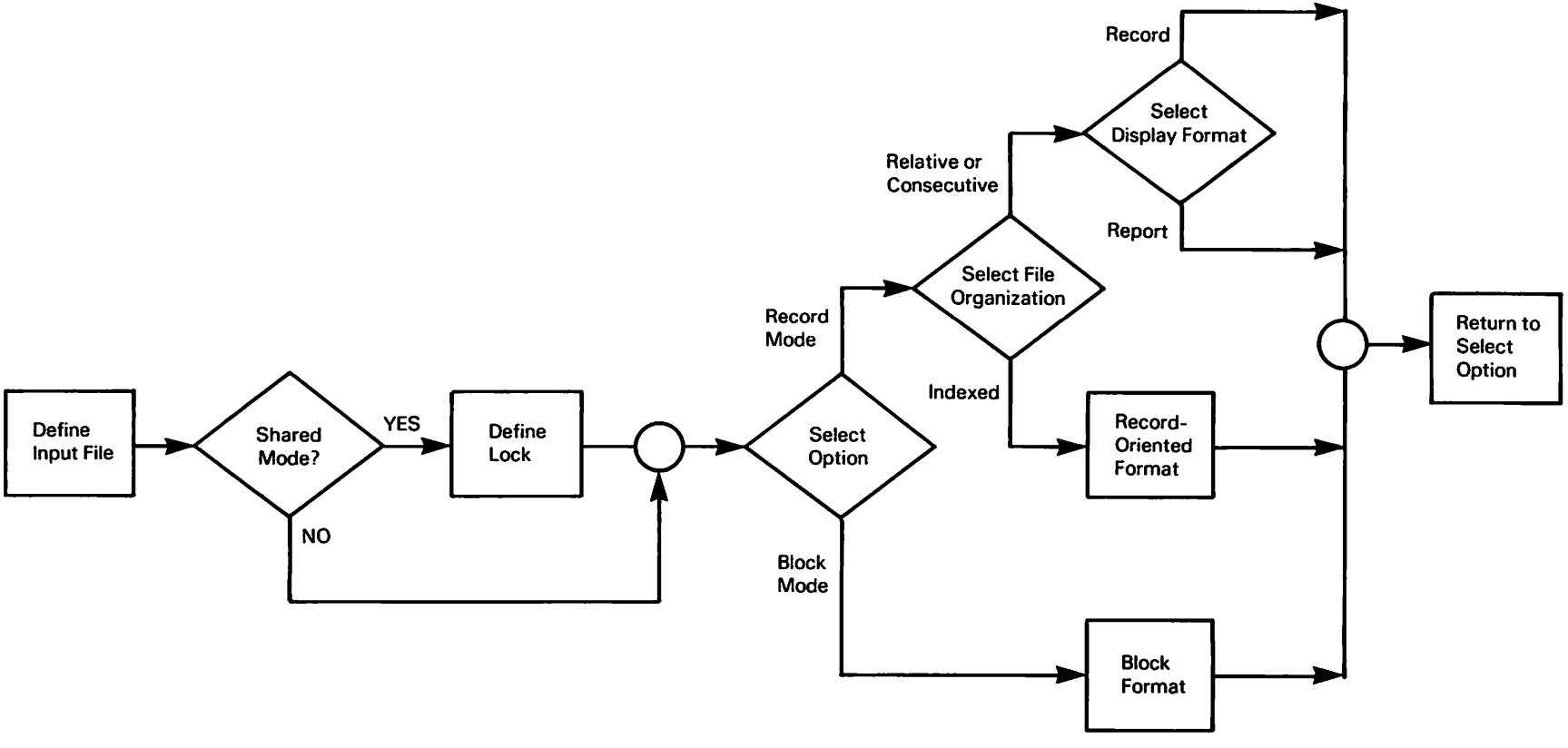


Figure 8-1. DISPLAY Processing

8.2 DEFINING DISPLAY INPUT

When DISPLAY processing begins, the DISPLAY Input Definition screen (Figure 8-2) appears. The DISPLAY Input Definition screen enables you to define the input file and select a Record access or Block access display.

```
***  MESSAGE 0000 BY DISPLY

                INFORMATION REQUIRED BY  PROGRAM  DISPLAY
                            TO DEFINE INPUT

            *** Wang VS File Display Utility - Version  x.xx.xx ***

To display a file, enter the name and location of the file to be displayed.
        FILE     = ******** in LIBRARY  = ******** on VOLUME   = ******
Select the access mode to be used when displaying the file.
        ACCESS   = RECORD       (RECORD / BLOCK)
Specify whether the file should be opened in input or shared mode.
        MODE     = INPUT*       (INPUT / SHARED)

                (Press PF16 to exit the DISPLAY program.)
```

Figure 8-2. DISPLAY Input Definition Screen

The fields on the DISPLAY Input Definition Screen are described as follows:

Field	Description
FILE	Specify the name of the file that you want to display or print.
LIBRARY	Specify the name of the library in which the input file resides. The default value for Library is your INLIB value, which is set through the SET Usage Constants (PF2) function from the Command Processor menu or through the Procedure language SET statement.
VOLUME	Specify the name of the volume on which the input file resides. The default value for the Volume field is your INVOL value, which is set in the same way that INLIB is set.
ACCESS	Specify the access mode (RECORD, BLOCK, or PRINT) in which the input data is displayed. The default value is RECORD. For more information on display formats, refer to Section 8.3. If you specify RECORD access, the input file is displayed in logical records. If the input file is consecutive or relative, you can format the records record by record or line by line. You must format indexed files record by record.

Field	Description
ACCESS (cont.)	If you specify BLOCK access, the input file is displayed in 2-KB physical records. Specifying BLOCK access enables you to view an exact image of the file layout as it exists on disk. You cannot display or print shared files in Block access mode. If the input file is consecutive or relative, you can also specify PRINT access. PRINT access displays the file in report-oriented format without first displaying the Display Options menu. PRINT access does not display a file in record-oriented format.
MODE	Specify the indexed or consecutive file to be opened as INPUT or SHARED. The default value is INPUT. To display a relative file, you must specify INPUT.

8.2.1 Displaying Files in Shared Mode

If you specify Shared mode, the DISPLAY Utility Lock screen (Figure 8-3) prompts you to indicate whether you want the file to be locked.

```
***  MESSAGE 0031 BY DISPLY

                INFORMATION REQUIRED BY  PROGRAM  DISPLAY
                            TO DEFINE LOCK



Please ENTER the following options to process a file in shared mode.

Should a lock be placed on the file to be displayed?   LOCK      = NO* (YES/NO)
  If YES, no changes to the file can be made while it
  is displayed. If NO, changes to the file can be made.

If LOCK=YES, specify TIMEOUT and BYPASS:

How many seconds should the TIMEOUT be?                TIMEOUT   = 10* SECONDS

If the timeout expires, should the file be bypassed?   BYPASS    = NO* (YES/NO)
```

Figure 8-3. DISPLAY Utility Lock Screen

The fields on the DISPLAY Utility Lock screen are described as follows:

Field	Description
LOCK	Specify (YES or NO) whether you want to suspend updates to a file that you are displaying. If you lock a file (YES), no changes to the file can occur while you are displaying it. If you specify NO, no lock is placed on the file, and there is no need to specify the Timeout and Bypass options. The default value is NO.

Field	Description
TIMEOUT	Specify a Timeout value (0 – 255 seconds) for a file if Lock = YES. If another user currently holds the file for update, the Timeout field specifies the length of time that the DISPLAY utility will wait to open the file in Shared mode with a lock. The default value is 10 seconds.
BYPASS	Specify (YES or NO) whether you want to skip a file if the Timeout value expires. If BYPASS = YES and the timeout expires, the DISPLAY utility skips the file. If BYPASS = NO and the timeout expires, the Lock screen reappears with the message "FILE filename IN libname ON volname IS HELD BY USER xxx." The default value is NO.

When timeout expiration occurs, you can continue with the DISPLAY operation; redefine the Lock, Timeout, and Bypass options, and press ENTER. The DISPLAY utility also displays a message that informs you of the following options:

- You can skip the particular file on which the timeout occurred by pressing PF1.
- You can terminate the DISPLAY operation if you are copying a library or volume by pressing PF16.

If a timeout expires with a Bypass value of NO during a background task, the DISPLAY operation is automatically cancelled.

NOTE
The Record Access Method (RAM) is always used to display shared files. For more information about RAM, refer to the VS Data Management System (DMS) Reference.

8.3 DISPLAY OPTIONS

After you specify the input file and the access type (RECORD, BLOCK, or PRINT), one of the six DISPLAY Options screens appears. Six different DISPLAY Options screens are available because each has a menu specific to the type of access and the organization of the input file. All of the DISPLAY Options screens are similar in appearance. The six types of access and organization of the input file are as follows:

- Record access, consecutive file, and report-oriented format
- Record access, consecutive file, and record-oriented format
- Record access, relative file, and report-oriented format
- Record access, relative file, and record-oriented format
- Record access, indexed file, and record-oriented format
- Block access

Consecutive and relative files can be displayed in either record-oriented format or report-oriented format. Indexed files can be displayed only in record-oriented format. You can change the file display on the record- and report-oriented format screens by pressing a PF key which is associated with an option.

Most PF keys used in DISPLAY processing perform the same function from screen to screen. However, some PF key functions change to accommodate the particular access and organization of the input file that you are displaying. Block access has the same options as the Record-Oriented Format listing with the exception of PF11 (Report Mode). The following list illustrates the different uses of PF keys for each format.

PF Key	Record-Oriented Format	Report-Oriented Format
1	Display	Menu
2	First	First
3	—	Position
4	Previous	Previous
5	Next	Next
6	Down	Down
7	Up	Up
8	Find Record	Find
9	Find Text	L Margin
10	Mode (ASCII/HEX)	R Margin
11	Report Mode	Left 15
12	—	Right 15
13	—	Left 1
14	—	Right 1
15	Print	Print
16	End Processing	Exit

A brief description of the various DISPLAY options is as follows:

Option	Description
Display	Press PF1 to display the input file in the specified access mode.
Menu	Press PF1 to return to the appropriate menu.
First	Press PF2 to display the first record (record-oriented), the first group of lines (report-oriented), or the first block of data in the file.
Position	Press PF3 (report-oriented format only) to display the line or record number and column position of the cursor for consecutive or relative files.
Indices	Press PF3 (indexed files only) to list the index descriptions for an alternate indexed file. This option enables you to change the access path.
Previous	Press PF4 to display the previous record (record-oriented), the previous group of lines (report-oriented), or the previous block of data in the file.
Next	Press PF5 to display the next record (record-oriented), the next 20 lines (report-oriented), or the next block of data in the file.
Down	Press PF6 to move the display down one record (record-oriented) or one line (report-oriented). If Block access is used to display the file, the display moves back one line within the block.
Up	Press PF7 to move the display forward (up) one record (record-oriented) or one line (report-oriented). If Block access is used to display the file, the display moves forward one line within the block.

Option	Description
Find	Press PF8 to find a record by the key value, record number, line number, text string, or block number that you enter. DISPLAY locates indexed file records by key value. In a consecutive or relative file, if you are displaying the file in Record access and the format is record-oriented, a record is found by record number; if Record access is used and the format is report-oriented, you can search for a line or text string. If Block access is used, a block is found by block number. (When finding hexadecimal values, you must use single quotes.)
Text	Press PF9 (record-oriented format or Block access) to find the next occurrence of a text string.
L Margin	Press PF9 (report-oriented format only) to display the first 80 columns of each line.
Mode	Press PF10 (record-oriented format or Block access) to change the display to hexadecimal mode if currently in ASCII, or to ASCII if currently in hexadecimal. This option is available only if Block access is used, or if Record access is used and the format is record-oriented.
R Margin	Press PF10 (report-oriented format only) to display the last 80 columns of each line.
Report	Press PF11 (available only for consecutive or relative files if Record access is used with record-oriented format) to display the file in a report-oriented format, that is, by line rather than by record.
Left 15	Press PF11 (report-oriented format only) to move the display 15 columns to the left, (that is, the display window is moved to the left, not the text).
Right 15	Press PF12 (report-oriented format only) to move the display 15 columns to the right (that is, the display window is moved to the right, not the text).
Left 1	Press PF13 (report-oriented format only) to move the display 1 column to the left (that is, the display window is moved to the left, not the text).
Right 1	Press PF14 (report-oriented format only) to move the display 1 column to the right (that is, the display window is moved to the right, not the text).
Print	Press PF15 to print all or part of the file. For additional information on printing a file, refer to Section 8.4.
Exit	Press PF16 (report-oriented format only) to return the file to record-oriented format; that is, the file is to be displayed by record rather than by line.
End Processing	Press PF16 (record-oriented format or Block access) to end the current file's display. This function generates a screen that gives the option of displaying another file or terminating the DISPLAY utility.

8.4 PRINTING A DISPLAY FILE

To print all or part of a file, press PF15 (the Print option). When you press PF15, the DISPLAY (Block Access) Print Options screen (Figure 8-4) appears. The DISPLAY Print Options screen varies slightly for each input display option available. Specify the starting and ending record, key, line, or block numbers, and the number of lines per page to be printed.

A file is printed according to the PRNTMODE default, which is set through the SET Usage Constants function (PF2) of the Command Processor menu. If PRNTMODE is set to S, the file will automatically be spooled for printing. If it is not set to S, the file is held until released for printing.

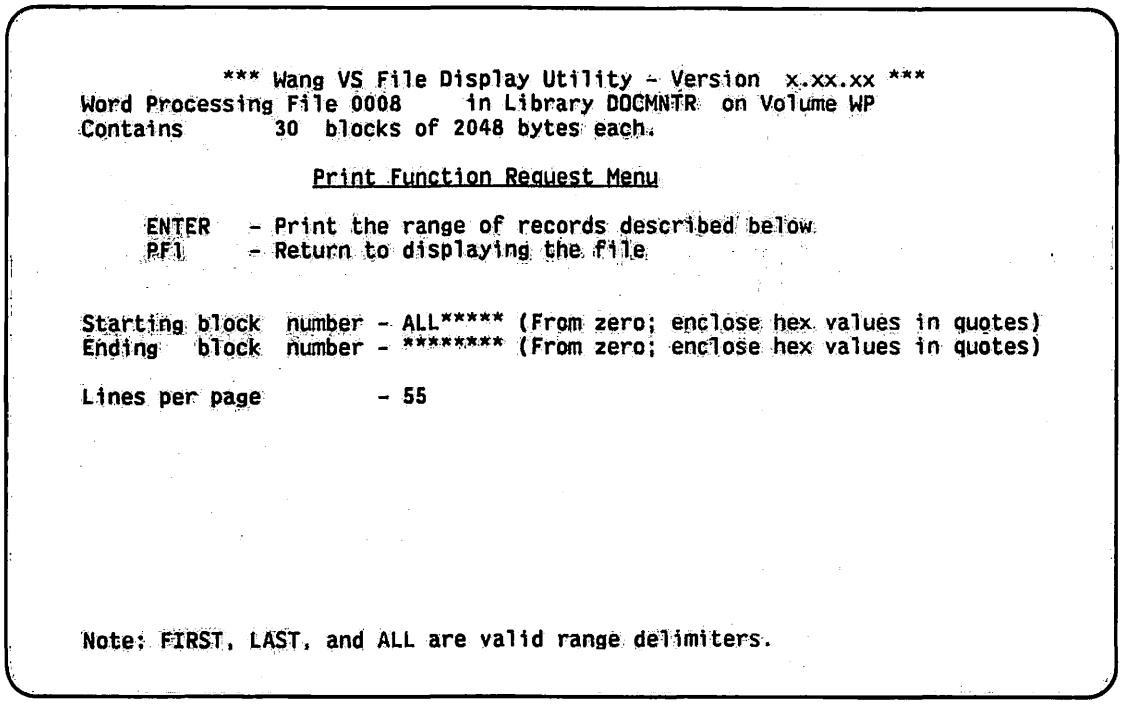


Figure 8-4. DISPLAY (Block Access) Print Options Screen

The DISPLAY Print Options screen fields are described as follows:

Field	Description
Starting Number	Specify the location at which the print file is to begin.

- If the format is record-oriented (for a consecutive or relative file) the Starting Number is a record number.
- If the format is report-oriented (for a consecutive or relative file) the Starting Number is a line number.
- If Record access is used for an indexed file, the Starting Number is a key number.
- If Block access is used, the Starting Number is a block number.

NOTE
If you specify the block number in hexadecimal, you must enclose the number in single quotes. In addition to record, line, key, or block number, FIRST is also a valid delimiter. Starting Number defaults to ALL.

Field	Description
Ending Number	Specify the location at which the print file ends.

- If the format is record-oriented (for a consecutive or relative file) the Ending Number is a record number.
- If the format is report-oriented (for a consecutive or relative file) the Ending Number is a line number.
- If Record access is used for an indexed file, the Ending Number is a key number.
- If Block access is used, the Ending Number is a block number.

NOTE
If you specify the block number in hexadecimal, enclose the number in single quotes. In addition to record, line, key, or block number, you can use LAST as a valid delimiter. No default for Ending Number exists.

Lines per Page	Specify the number of lines that you want printed on each page. Lines per Page defaults to the Lines value that was previously set through the SET Usage Constants function (PF2) of the Command Processor menu.

8.5 DISPLAY FORMATS

Included at the top of each DISPLAY Options screen is the input file name, the file organization, the maximum number of bytes in each record or block, and the number of records or blocks in the file. Key position, key size, and alternate index information is included for indexed files. The six DISPLAY Options screens are described in the sections that follow.

8.5.1 Record Access, Consecutive File, and Record-Oriented Format

The file is displayed record by record. Records are displayed in the order in which they appear in the file. You can use PF10 to display the record in the ASCII or hexadecimal representation. Pressing PF10 converts the record display to the opposite of the current representation. You can print a hard copy in this format for all or part of the file. The following example is a consecutive file in record-oriented format (ASCII representation):

```
RECORD 1
   0   13217BOS, BLANCHE      100 JERICHO RD     BUTLER
  64               NJ07405
RECORD 2
   0   14281CLEMENS, CORA     304 VASSAR RD      POUGHK
  64   EEPSIE     NY12648
```

```
RECORD 3
   0    14359FARNABY, ANNIE    243 23RD ST        CUYAHO
  64    GA FLS     OH44223
RECORD 4
   0    15692IONA, HELEN       23 E ELM AVE       QUINCY
  64              MA02170
```

8.5.2 Record Access, Consecutive File, and Report-Oriented Format

The file is displayed line by line; that is, each record is displayed on one line. The options for manipulating the display operate on a line basis. You can perform horizontal and vertical scrolling. You can print a hard copy in this format for all or part of the file. The following example is a consecutive file in report-oriented format:

```
13217BOS, BLANCHE     100 JERICHO RD  BUTLER          NJ07405
14281CLEMENS, CORA    304 VASSAR RD   POUGHKEEPSIE    NY12642
14359FARNABY, ANNIE   243 23RD ST     CUYAHOGA FLS    OH44223
15692IONA, HELEN      23  E ELM AVE   QUINCY          MA02170
```

8.5.3 Record Access, Relative File, and Record-Oriented Format

The file is displayed record by record. Records are displayed in the order in which they appear in the file. Empty record slots are not displayed. Zero-length records, however, are displayed; the phrase "<empty>" appears beneath the record number to indicate that the record contains no data. You can use PF10 to display the record in the ASCII or hexadecimal representation. Pressing PF10 converts the record display to the opposite of the current representation. You can print a hard copy in this format for all or part of the file.

The following example is a relative file in record-oriented format (ASCII representation). (Because Record 2 represents an empty record slot, it is not displayed. Record 5 is a zero-length record.)

```
RECORD 1
   0    13217BOS, BLANCHE      100 JERICHO RD     BUTLER
  64              NJ07405
RECORD 3
   0    14281CLEMENS, CORA     304 VASSAR RD      POUGHK
  64    EEPSIE     NY12648
RECORD 4
   0    14359FARNABY, ANNIE    243 23RD ST        CUYAHO
  64    GA FLS    OH44223
```

```
RECORD 5
  <empty>
RECORD 6
   0   15692IONA, HELEN        23 E ELM AVE       QUINCY
  64              MA02170
```

8.5.4 Record Access, Relative File, and Report-Oriented Format

The file is displayed line by line as in Section 8.5.1. <MISSING RECORD> indicates an empty record slot. <EMPTY RECORD> indicates a record that is present but contains no data (a zero-length record). The options for manipulating the display operate on a line-by-line basis. You can perform horizontal and vertical scrolling. You can print a hard copy in this format for all or part of the file. The following example is a relative file in report-oriented format:

```
13217BOS, BLANCHE     100 JERICHO RD  BUTLER          NJ07405
  <MISSING RECORD>
14281CLEMENS, CORA    304 VASSAR RD   POUGHKEEPSIE    NY12648
14359FARNABY, ANNIE   243 23RD ST     CUYAHOGA FLS    OH44223
  <EMPTY RECORD>
15692IONA, HELEN      23  E ELM AVE   QUINCY          MA02170
```

8.5.5 Record Access, Indexed File, and Record-Oriented Format

The file is displayed record by record in primary key sequence. You can change the access path of an indexed file that has alternate indexes.

You should always specify the first record that you want displayed when you change the access path or select a record by key or text string. (Do this to ensure that the entire file is read because an indexed file always displays the next available record. It does not automatically go back to the beginning of the file.)

You can alternate between ASCII and hexadecimal by pressing PF10. When you press PF10, the present code that is displayed is converted to the other code. You can print a hard copy in this format for all or part of the file. The following example is an indexed file in record-oriented format:

```
KEY = 132
   0   13217BOS, BLANCHE      100 JERICHO RD     BUTLER
  64              NJ07405
KEY = 142
   0   14281CLEMENS, CORA     304 VASSAR RD      POUGHK
  64   EEPSIE     NY12648
```

```
KEY = 143
   0   14359FARNABY, ANNIE     243 23RD ST        CUYAHO
  64   GA FLS    OH44223
KEY = 156
   0   15692IONA, HELEN        23 E ELM AVE       QUINCY
  64             MA02170
```

8.5.6 Block Access

Records are displayed in 2-KB (2048-byte) blocks in logical block sequence, starting with the first block in the first extent of the file. Each line of the block contains 64 bytes.

Because Block access mode displays an exact image of each block of the file as it is recorded on disk, control characters (i.e., tab, return, indent) are displayed as special characters. You can alternate between ASCII and hexadecimal by pressing PF10. When you press PF10, the present code that is displayed is converted to the other code. Data blocks and index blocks are displayed for an indexed file. You can print a dump in this format for all or a part of the file. The following example is a file in block format:

```
Block 0
   0    ▶13217BOS, BLANCHE                100 JERICHO RD            BUTLER
  64             NJ07405◀ ▶14281CLEMENS, CORA              304 VASSAR RD
 128              POUGHKEEPSIE   NY12648◀ ▶14359FARNABY, ANNIE
 192      243 23RD ST              CUYAHOGA FLS   OH44223◀ ▶15692IONA,
 256    HELEN               23 E ELM AVE              QUINCY            MA
 320    02170
 384
 448
 512
 ...
 ...
1984
```

8.6 A SAMPLE DISPLAY PROCEDURE

You can control DISPLAY processing through the VS Procedure language. While you can specify the input to DISPLAY, you cannot specify the menu options because there are no GETPARM requests. For a complete list of DISPLAY GETPARMs refer to Appendix A. Refer to the *VS Procedure Language Reference* for details about Procedure language syntax.

The following example shows a procedure to run DISPLAY. Using this procedure you can display, manipulate, and print a hard copy of the file in Record access mode.

```
PROCEDURE
    RUN DISPLAY
        ENTER INPUT FILE=SAVE3, LIBRARY=DJDDATA,
            VOLUME=NEWSYS, ACCESS=RECORD
        ENTER EOJ 16
RETURN
```

If the input file is consecutive or relative, you can specify ACCESS = PRINT instead of ACCESS = RECORD as shown in the example. ACCESS = PRINT displays the file in report-oriented format and does not display the Display Options menu first. You cannot use ACCESS = PRINT for a file in record-oriented format. The end-of-job option terminates the DISPLAY utility.

CHAPTER 9
THE EZFORMAT UTILITY

9.1 INTRODUCTION

The EZFORMAT utility enables you to develop program screens that can easily be used in your program applications. You can perform the following functions with EZFORMAT:

- You can dynamically create a workstation screen image and generate an Assembly language, BASIC, COBOL, or RPG II source code program that reproduces the screen display that you created.
  - You can specify messages and input requests on the screen.
  - You can copy the source code that EZFORMAT generates into your program through the external copy feature of the EDITOR utility during program development. Refer to the *VS Program Development Tools Reference* for information about the EDITOR utility.
  - You can experiment with several screen formats before EZFORMAT generates the source code for the final screen format.
- You can create a complete Assembly language or RPG II menu program that associates specified programs with program function (PF) keys. You can design the workstation screen image that describes this menu.
- EZFORMAT processing can be controlled through the VS Procedure language (described in Section 9.14).
- You can create a COBOL or RPG II data entry program (with the Data Entry option) from a screen display that you design.

NOTE
This manual does not describe the Data Entry option of EZFORMAT because the Data Entry option is used exclusively in conjunction with the CONTROL utility. For information about the Data Entry option, refer to the VS File Management Utilities Reference.

Figure 9-1 shows an overview of the EZFORMAT processing.

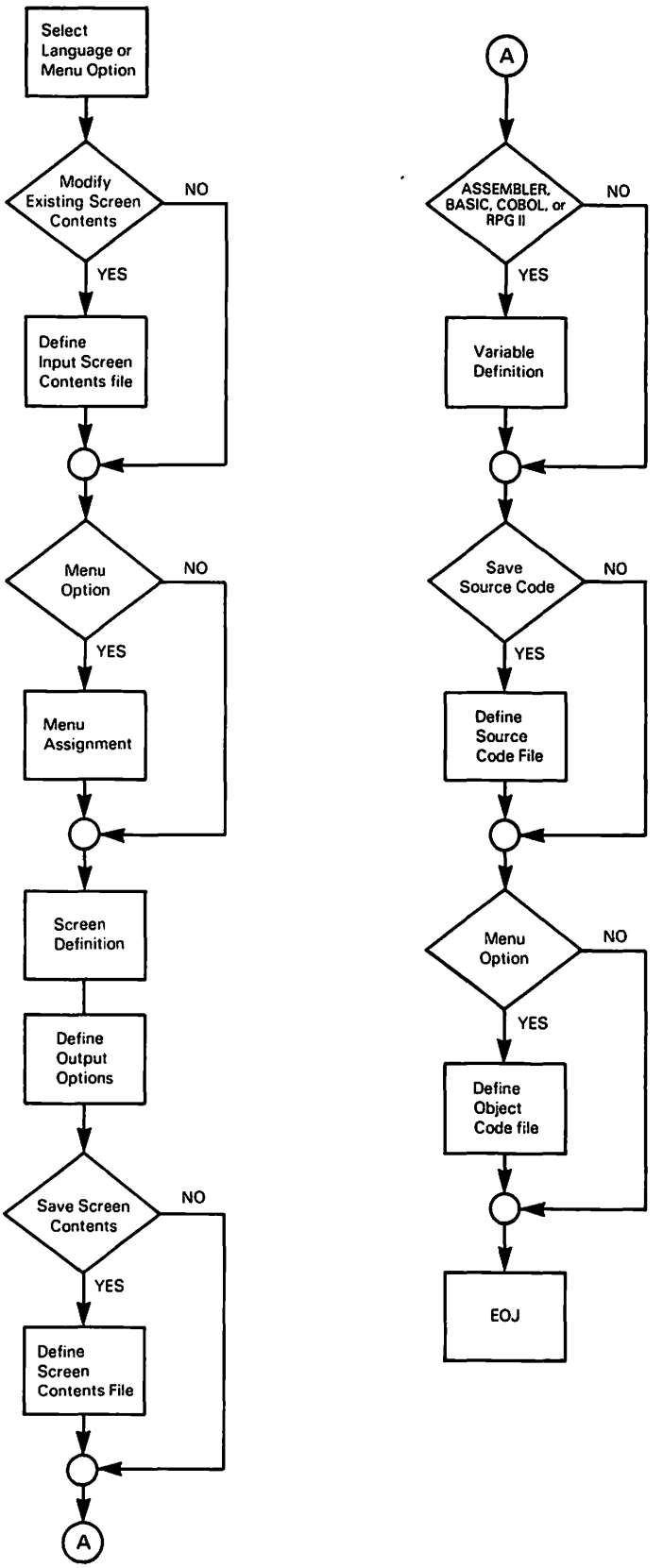


Figure 9-1. EZFORMAT Processing

9.2 EZFORMAT PROCESSING

This chapter describes the procedures to run EZFORMAT. You can run the EZFORMAT utility by pressing PF1 (RUN Program or Procedure) from the Command Processor menu or you can run EZFORMAT through the CONTROL utility. Refer to the *VS File Management Utilities Reference* for details on the CONTROL utility. EZFORMAT processing involves the following steps:

1. Select a programming language or menu option. (For the menu option, define the functions to be performed by the menu program.)
2. Define the screen image.
3. Define the output files. (For the menu option, define the language to be used to create the menu program.)

9.3 EZFORMAT INPUT OPTIONS

The EZFORMAT Input Options screen (Figure 9-2) appears when processing begins. Specify a programming language or menu option by typing in one of the listed languages (or MENU) in the Option field. You can also specify just the first letter of the option that you choose.

```
                WANG VS SCREEN FORMAT UTILITY - VERSION x.xx.xx

  This utility generates the statements that are needed to create the screen
      format established via interaction with the workstation operator.
The statements may be used as a source copy file with the language specified.
  This utility can also be used to create a data entry program in conjunction
      with the 'CONTROL' utility, with COBOL as the source language, or
         a program menu screen with ASSEMBLER as the source language.
          Please select the option desired and the appropriate PFkey.

                      Format Definition Input Options

                       OPTION = C*********    ('COBOL', 'ASSEMBLER', 'BASIC',
                                               'RPG', 'DATA ENTRY', OR 'MENU')

                       PFkey          Action

                        PF2           Create New Format Definition
                        PF3           Use previously saves screen contents
                                      as input for format definition

                        PF16          Exit without generating format definition
```

Figure 9-2. EZFORMAT Input Options Screen

The Option field determines the program code that is generated to reproduce your screen image design as part of a user or menu program. A description of the EZFORMAT options is as follows:

Option	Description
ASSEMBLER	Specify ASSEMBLER or A in the Option field to create a screen image in Assembly language source code. The source code is converted into an Assembly language source file that can be copied into an Assembly language program to be used as part of the program application.
BASIC	Specify BASIC or B in the Option field to create a screen image in BASIC source code. The source code is converted into a BASIC source file that can be copied into a BASIC program to be used as part of the program application.
COBOL	Specify COBOL or C in the Option field to create a screen image in COBOL source code. The source code is converted into a COBOL source file that can be copied into a COBOL program to be used as part of the program application.
DATA ENTRY	Specify DATA ENTRY of D to create a data entry screen image in conjunction with the CONTROL utility. For more information about the Data Entry option, refer to the *VS File Management Utilities Reference*.
MENU	Specify MENU or M in the Option field to create a program menu screen image. The program menu screen image uses Assembly or RPG II as the source language and can be used for initiating system and program applications.
RPG	Specify RPG or R in the Option field to create a screen image in RPG II source code. The source code is converted into an RPG II source file that can be copied into an RPG II program to be used as part of the program application.

After you have specified an option in the Option field, select PF2 (Create New Format Definition) if you are creating a new screen image, or PF3 (Use Previously Saved Screen Contents) if you want to modify an existing screen image. The second function is also useful if you want to copy and convert an existing screen image into another programming language. You can cancel processing from the EZFORMAT Input Options screen by pressing PF16 (Exit Without Generating Format Definition).

9.4 THE PROGRAMMING LANGUAGE OPTIONS

If you specify a programming language in the Option field (instead of the Menu option) on the EZFORMAT Input Options screen (Figure 9-2) and press PF2 (Create New Format Definition), then the EZFORMAT Screen Format Options screen (Figure 9-3) appears.

```
                        Screen Format Options

These formats are used to arrange and define the fields on the screen.

   Field Type                 Valid Formatting Options
   Text                       Must have double quotes ["].
   Numeric Modifiable         Must be [0 - 9], [+], [-], or [.].
   Alphanumeric Modifiable    Must begin with an alphabetic character.

The following characters are included in the length of a modifiable field.
   *   This is a required alphanumeric field (cannot be blank).
   +   This is a zoned numeric field, result may be [+] or [-].
   -   This is a zoned numeric field, result may be space or [-].
   .   This is a decimal point (position) in a numeric field.

Source and Object data-names are generated for all modifiable fields.
Source and Object data-names can be defined after screen format definition.
Value clauses are generated for modifiable fields where the first
 character is not one of the following: [X]  [*]  [9]  [+]  [-]  [.]

A printed copy of these instructions plus the menu of options can be
 obtained by pressing PF13 from the next menu.

                    **  Press ENTER to continue  **
```

Figure 9-3. EZFORMAT Screen Format Options Screen

The EZFORMAT Screen Format Options screen describes the options that you have when designing a program language screen image. You can create user-modifiable fields on these screens. When a program is run with these screens, you can input data requested by the program.

NOTE

Screen images that are created through EZFORMAT are defined in terms of fields, which are groups of characters of less than one line long. The Menu and language options support different types of fields:

- *The Menu option supports displayed text only.*
- *The language options support displayed text and modifiable alphanumeric and numeric fields.*

ALL modifiable text fields must be created in uppercase characters.

The EZFORMAT Screen Format Options screen is not displayed for the Menu option because only the text field type is available. The following field types are available for language option screen images: text fields, numeric modifiable fields, and alphanumeric modifiable fields.

9.4.1 Text Fields

Wherever text fields are placed on the screen, those fields cannot be modified. No input is allowed in a text field when the screen is displayed.

When creating a screen image, the text specified on the screen is a protected field if the text is enclosed in double quotation marks. A quotation mark occupies a screen position, but is not displayed on the generated screen. You do not need to use quotes for the text fields when you define a screen for the Menu option because text fields are the only type of fields that you can define on the screen image.

9.4.2 Numeric Modifiable Fields

Numeric modifiable fields accept only numeric input when the screen image that you design is presented in your program. A numeric field is identified by a 9, plus sign, minus sign, or any numeric character as the initial character in the field. A numeric field is ended by a space, the end of the line, or the double quote indicator of a Text field. Numeric field definitions are described as follows:

- For the COBOL and RPG II options, an initial plus sign generates a field that converts any entered negative data to positive values. For example, if you enter −44 into a field specified as +99, b44 (a space followed by 44) is stored. You can enter negative, positive, or unsigned values into a field with an initial specification of a minus sign.
- If the field specification consists of an initial 9, a plus sign followed by a 9, or a minus sign followed by a 9, then pseudoblanks are displayed in the resulting field.
- If you specify the field with an initial numeric character other than 9, or if you specify the initial plus or minus sign followed by a number other than 9, then the specified number is displayed as a default value when the screen is displayed.
- The initial plus or minus sign is displayed as a pseudoblank when the screen design program is run, regardless of whether it is followed by a default value or pseudoblanks.

9.4.3 Alphanumeric Modifiable Fields

Alphanumeric modifiable fields accept alphanumeric input when the screen image that you design is presented in your program. An alphanumeric field is identified by an initial character other than a number, a plus sign, or a minus sign.An alphanumeric field is ended by a space, the end of the line, or the double quote indicator of a Text field. Alphanumeric field definitions are described as follows:

- An asterisk (*) that occurs before any line in a BASIC or COBOL program specifies that this field is a required alphanumeric field; that the field cannot be left blank when the screen is processed.
- If the field specification contains an initial X, or an * followed by an X, the program displays pseudoblanks in the resulting field.
- If you specify the field with an initial character other than X, or if you type the asterisk (*) followed by a letter other than X, the specified character string is displayed (and is modifiable) as a default value.
- An initial asterisk (*) is always displayed as a pseudoblank when the screen design program is run.

9.5 THE MENU OPTION

If you specify MENU (or M) in the Option field on the EZFORMAT Input Options screen and press PF2 (Create New Format Definition), the EZFORMAT Menu-Generator screen (Figure 9-4) appears.

```
                        MENU GENERATOR SCREEN

Please associate the names of the programs which will be executed from the
     menu to be defined, with the PFkey list displayed on this screen.
Special functions:  Enter if desired beside PFkey 'EXIT' to exit from menu,
  'LOGOFF' to logoff, or 'USERPROG' to run program not listed on the menu.
 On the blank format definition screen, define the menu exactly as you wish
     it to be displayed (Do not delimit the text with double quotes).

PFKEY   PROGRAM NAME      PFKEY   PROGRAM NAME      PFKEY   PROGRAM NAME
ENTER     USERPROG        PF1       SORT****        PF2       ********
PF3       ********        PF4       DISPLAY*        PF5       COPY****
PF6       ********        PF7       ********        PF8       LISTVTOC
PF9       IBMCOPY*        PF10      ********        PF11      ********
PF12      ********        PF13      ********        PF14      ********
PF15      ********        PF16      LOGOFF**        PF17      ********
PF18      ********        PF19      ********        PF20      ********
PF21      ********        PF22      ********        PF23      ********
PF24      ********        PF25      ********        PF26      ********
PF27      ********        PF28      ********        PF29      ********
PF30      ********        PF31      ********        PF32      EXIT****
          DISABLE HELP KEY DURING MENU EXECUTION  YES  (YES OR NO)
                       **  Press ENTER to continue  **
```

Figure 9-4. Sample EZFORMAT Menu-Generator Screen

9.5.1 PF Key Assignment

Before you can define the screen image, you must associate the PF keys to be used in the menu with specified programs or functions. You can assign functions to the ENTER key and the PF1 through PF32 keys through the EZFORMAT Menu-Generator screen (Figure 9-4). You can also disable the HELP key during menu program processing. Any program or procedure that resides in the same library as the menu object code or in the system library (@SYSTEM@) can be directly associated with a PF key. To associate a program or procedure with a PF key, you type the file name in the field corresponding to the PF key.

Figure 9-4 illustrates the assignment of programs and functions to specified PF keys. For example, if you use the SORT, DISPLAY, COPY, LISTVTOC, and IBMCOPY utilities fairly often and you often have to switch between utilities, an EZFORMAT menu can simplify your tasks. With an EZFORMAT menu, you can press a single PF key and automatically run a specified utility. You can further simplify your tasks by automatically calling up the EZFORMAT menu program whenever you log onto the system through the VS Procedure language. (Refer to the *VS Procedure Language Reference* for more information about the VS Procedure language. A sample Procedure language procedure is provided in Section 9.14.)

Without a user-created menu, you have to run all the utilities by pressing PF1 (RUN Program or Procedure) from the Command Processor menu. Then you must specify the utility or program in the PROGRAM field, and sometimes you must also specify the library and volume on which it resides.

By creating a menu, and assigning PF keys to specific programs or functions, you can gain immediate access to the program that you specified by simply pressing the PF key that corresponds to the desired function. In Figure 9-4, SORT is assigned PF1, and so on. Thereafter, when the menu program is run, pressing PF1 automatically accesses the SORT utility. (Refer to Chapter 22 for information about the SORT utility.) The generated menu performs only those programs or functions that you associate with workstation PF keys.

You can run any user program from the menu if you specify the USERPROG option as a key function. The USERPROG option is especially handy if you want to design a screen with only a few PF key assignments but still want to have the option of running any other program from your particular screen design. When you press the PF key that is associated with USERPROG, the RUN Program or Procedure screen (PF1 from the Command Processor menu) appears.

You can terminate the menu program if you specify EXIT as a key function. If you specify LOGOFF as a key function, LOGOFF returns control to the process one link level up. For example, if the menu is running under control of a link from another program, when the menu LOGOFF function is performed, the menu user is returned to the calling program and not logged off. You can log off the system from the menu program if the menu program is the only program running.

You can disable the HELP key if you specify YES in the HELP key field. The HELP key field has a default value of NO.

9.5.2 The Created Screen Image

After you define the menu keys, press ENTER to create the screen image that displays the menu keys. With the Menu option, EZFORMAT incorporates your screen image design as the actual menu display for the generated menu program.

NOTE

You can only view the menu options that you defined on the EZFORMAT Menu-Generator screen (Figure 9-4) through the screen image that you create. If you exit the screen image definition screen without any text (that is, blank), EZFORMAT considers the blank screen to be the screen image that you created.

When you run your screen image program, the PF keys that you defined on the EZFORMAT Menu-Generator screen (Figure 9-4) still operate even though a blank screen is displayed on your workstation. This is because an executable menu program is generated from the PF key values that you specified on the EZFORMAT Menu Generator screen. The screen image is only a tool to help you remember which PF keys to press to accomplish the tasks that you defined.

When you exit from defining the screen image, you can select either the ASSEMBLER or the RPG II option to specify the language in which the menu is to be generated. An Assembly language or RPG II menu program can perform any or all of the following functions from an EZFORMAT-generated menu:

- Execute a specific program
- Allow the menu user to run a program
- Exit the menu
- Log the menu user off the system

9.6 CREATING AN EZFORMAT SCREEN IMAGE

After you specify the menu assignments and press ENTER (or select one of the programming languages and press PF2), EZFORMAT processing continues by displaying the EZFORMAT Screen Manipulation Options screen (Figure 9-5).

```
                    SCREEN MANIPULATION OPTIONS

  Functions 'ENTER' thru PF9, PF12, and PF14 are the functions
   available to the user while in the screen definition mode.
     Functions PF12 thru PF16 are available from this menu.
          Target row is determined by cursor position.

      PFKEY          ACTION
      ENTER          Display menu of screen manipulation options.

       PF2           Show cursor column position.
       PF3           Move target row up to previous line.
       PF4           Move target row down to next line.
       PF5           Copy target row up to previous line.
       PF6           Copy target row down to next line.
       PF7           Roll up Row 23 thru target row.
       PF8           Roll done Target row thru Row 23.
       PF9           Center contents of target row.
       PF12          Set tabs and uplow.
       PF13          Print copy of instructions.
       PF14          Redisplay the list of PFkeys selected.
       PF16          End screen format definition.

        **  Press ENTER to define the screen  **
```

Figure 9-5. EZFORMAT Screen Manipulation Options Screen

The EZFORMAT Screen Manipulation Options screen appears the same for the programming language and menu options **except** that PF14 performs a different function for each option. Also, if you chose the programming language option, an extra PF key is listed (PF1, Display Menu of Screen Format Options).

You can dynamically design a screen image by entering field specifications in the desired screen locations. You can define Rows 1 through 24 of the workstation screen for the ASSEMBLER, BASIC, COBOL, and RPG options. You can define Rows 1 to 23 for the menu option (Row 24 is reserved for message display during program execution).

Each row on the screen begins with a nondisplayed Field Attribute Character (FAC) to designate the row's default field type. As a result, only 79 columns of the workstation screen are available for designing the screen image. For the language options, the Screen Definition image initially shows all pseudoblanks. For the Menu option, the initial display is blank.

The cursor control keys (HOME, TAB, the arrow keys, etc.) simplify screen creation. In addition to the cursor control keys, the EZFORMAT Screen Manipulation options provide the following functions for screen design:

PF Key	Function	Description
ENTER	Display Menu of Screen Manipulation Options	Press ENTER to alternate between the EZFORMAT Screen Manipulation Options screen (Figure 9-5) and the screen image.
1	Display Menu of Screen Formatting Options	Press PF1 (used with the programming language option only), to display the EZFORMAT Screen Format Options screen. This screen displays the field types that are available to you when you design a program screen image. This function is not available for the Menu option because the Menu option only allows text fields.
2	Show Cursor Column Position	Press PF2 to show the current column position of the cursor in the upper righthand portion of the screen (temporarily overwriting any field defined in that location). The cursor position display does **not** destroy any data.
3	Move Target Row Up to Previous Line	Press PF3 to move the entire contents of the target row (specified by the cursor position) to the same positions in the previous row. The contents of the previous row are overwritten and destroyed. The original row is replaced by pseudoblanks (or blanks, for the Menu option). No action occurs if you attempt to move Row 1. The function is illustrated as follows; Row 7 is the target row in the example:
4	Move Target Row Down to Next Line	Press PF4 to move the entire contents of the target row (specified by the cursor position) to the same positions in the following row. The contents of the following row are overwritten and destroyed. The original row is replaced by pseudoblanks (or blanks, for the Menu option). No action occurs if you attempt to move Row 24 (Row 23 for the Menu option).
5	Copy Target Row Up to Previous Line	Press PF5 to copy the entire contents of the target row (specified by the cursor position) to the same positions in the previous row. The contents of the target row are unchanged. The contents of the previous row are destroyed. The target row and the previous row are then identical. No action occurs if you attempt to copy Row 1. This function is illustrated as follows; Row 7 is the target row in the example:

Row #	Data Before	Data After
5	AAA	AAA
6	BBB	CCC
7	CCC	(pseudo)blanks
8	DDD	DDD

Row #	Data Before	Data After
5	AAA	AAA
6	BBB	CCC
7	CCC	CCC

PF Key	Function	Description
6	Copy Target Row Down to Next Line	Press PF6 to copy the entire contents of the target row (specified by the cursor position) to the same positions in the following row. The contents of the target row are unchanged. The contents of the following row are destroyed. The target row and the following row are then identical. No action occurs if you attempt to copy Row 24 (Row 23 for the Menu option).
7	Roll Up Row 24 Thru Target Row	Press PF7 to move the contents of each row to the row above (roll up). However, this function only moves rows up from the target row through Row 24 (Row 23 for the Menu option). The contents of the target row are destroyed by copying the row below into the target row. Subsequent rows are copied upwards and Row 24 is overwritten with pseudoblanks (blanks on Row 23 if you are using the Menu option). The roll-up function is illustrated as follows; Row 20 is the target row in the example for a language option.
8	Roll Down Target Row Thru Row 24	Press PF8 to move the contents of each row to the row below (roll down). However, this function only moves rows down from the target row through Row 24 (Row 23 for the Menu option). The contents of Row 24 (or 23 for the Menu option) are destroyed by copying the previous row into it. Subsequent previous rows are copied down to the next row until the target row is copied one row down. The target row is then overwritten with (pseudo)blanks. If the target row is placed at Row 24 (Row 23 for the Menu option), the row is overwritten with (pseudo)blanks.
9	Center Contents of Target Row	Press PF9 to center the contents of the target row (horizontally).
12	Set Tabs And UPLOW	Press PF12 to display a screen that enables you to set up to ten tab positions and define the input in all uppercase letters or uppercase and lowercase (UPLOW) letters.

Row #	Data Before	Data After
18	RRR	RRR
19	SSS	SSS
20	TTT	UUU
21	UUU	VVV
22	VVV	WWW
23	WWW	XXX
24	XXX	(pseudo)blanks

NOTE
ALL modifiable text fields must be created in uppercase characters.

Set the tabs by typing a nonblank character underneath the desired column position. Specify UPLOW in the Mode field for uppercase and lowercase input (UPPER if you want only uppercase input). Text fields and default values for modifiable fields can have lowercase values, but lowercase input is not accepted in modifiable fields during program execution.

PF Key	Function	Description
13	Print Copy of Instructions	Press PF13 to create a print file that contains the information given on the EZFORMAT Screen Format Options screen (Figure 9-3) and the EZFORMAT Screen Manipulation Options screen (Figure 9-5).
14	Display the Screen as it will Appear in Program	Press PF14 (used with the programming language options only) to display the screen as it will appear in your program. If you press ENTER from the finished screen display, the EZFORMAT Screen Manipulation Options screen reappears. If you are using the Menu option, PF14 has another function — Redisplay the list of PF keys selected.
14	Redisplay the List of PF Keys Selected	Press PF14 (used with the Menu option only) to display the selections that you made on the EZFORMAT Menu-Generator screen (Figure 9-4). If you press ENTER, the EZFORMAT Screen Manipulation Options screen (Figure 9-5) reappears.
16	End Screen Format Definition	Press PF16 to end the screen definition process.

When you have finished designing the screen image, press PF16 from the EZFORMAT Screen Manipulation Options screen (Figure 9-5). EZFORMAT then creates output files according to your instructions.

9.7 SAVING AN EZFORMAT SCREEN IMAGE

After you create a screen image, EZFORMAT can save the screen contents and generate source code that reproduces the screen. For the menu option, object code that executes the constructed menu can also be generated. To save the screen image, press PF16 (End Screen Format Definition) from the EZFORMAT Screen Manipulation Options screen and the EZFORMAT Output Options screen (Figure 9-6) appears.

```
***  MESSAGE 0001 BY EZFORM

                    RESPONSE REQUIRED BY  PROGRAM  EZFORMAT
                          TO SELECT FUNCTION
                       ACTIVE SUBPROGRAM IS EZFORMAT

                       WANG VS SCREEN FORMAT UTILITY


                      EZFORMAT OUTPUT OPTIONS

                       PFKEY      ACTION

                        PF1       Return to the function selection screen
                                  without saving screen contents
                        PF2       Save generated output only
                        PF3       Save screen contents only
                        PF4       Save screen contents and generated output

                        PF16      Exit without saving any files

NOTE:  THE SCREEN CONTENTS, IF SAVED, CAN BE REUSED AS INPUT TO EZFORMAT.
```

Figure 9-6. EZFORMAT Output Options Screen

The generated output differs according to the selected function but, in general, the output is the source or object code, or both, that corresponds to the specified screen format. Whether you specified one of the programming language options, or the Menu option, the EZFORMAT Output Options screen offers the following options:

Key	Action
1	Return to the EZFORMAT Input Options screen (Figure 9-2) to select a function
2	Save generated output only
3	Save screen contents only
4	Save screen contents and generated output
16	Exit without saving any files

Saving the screen contents saves the image of the screen in a file. Saving the generated output saves the source code that recreated the screen image in the programming language that you specify.

If you save the screen contents by pressing PF key 3 or 4 on the EZFORMAT Output Options screen, an EZFORMAT Save Screen Image Definition screen appears. Specify a file, library, and volume in which to save the screen image. EZFORMAT supplies a default library name by concatenating your user ID with SAVE (MMSAVE, for example), but you can override this value if you want it saved in another library or volume. The SAVE file is used as input to EZFORMAT if you want to modify the screen display (refer to Section 9.13 for details).

The file that holds the screen image contains the row-by-row data (a picture-image) for the screen image design. For the Menu option, the file contains the screen image and also includes the workstation key assignments (refer to Section 9.5.1). If you save the screen contents, you can modify the screen image at a later date (refer to Section 9.13 for information about modifying existing screen images).

If you save only the screen contents, EZFORMAT processing terminates after you name the file that contains the screen contents. If you save the generated output, subsequent processing differs for each EZFORMAT option. The following sections describe the procedure (for each generated output) to save your screen image.

9.8 ASSEMBLY LANGUAGE-GENERATED OUTPUT

If you specify OPTION = ASSEMBLER on the EZFORMAT Input Options screen (Figure 9-2) and save the generated output, EZFORMAT creates an Assembly language source file. When you use the source file with an Assembly language program, the source file recreates the screen image at the workstation as part of the program. You can copy this source file into an Assembly language program through the XCOPY function of the EDITOR utility (refer to the *VS Program Development Tools Reference*).

NOTE
The source file consists entirely of Defined Constant statements; you must code any corresponding WRITE or READ statements in the main assembler program.

When you save the generated output (PF key 2 or 4 on the EZFORMAT Output Options screen), EZFORMAT prompts you to specify the file, library, and volume names for the Assembly language source file. The library name defaults to the value obtained by concatenating your user ID with COPY. You can override this value if you want to store it in another library or volume. Processing terminates after the Assembly language source file is created. EZFORMAT return codes are listed in Appendix B.

9.9 BASIC LANGUAGE-GENERATED OUTPUT

If you specify the BASIC option on the EZFORMAT Input Options screen (Figure 9-2) and save the generated output, EZFORMAT creates a BASIC source file. When you use the source file with a BASIC program, the source file recreates the screen image at the workstation as part of the program and also reads any data entered into any modifiable fields. EZFORMAT also defines and dimensions fields of the appropriate data type corresponding to the modifiable fields on the screen. The resulting BASIC source file contains appropriate DISPLAY, ACCEPT, and DIM statements.

NOTE
While the generated source code runs when compiled, the BASIC-generated screen image is displayed only once. Data entered on the generated screen is lost unless you modify the source code before compilation of the program.

9.9.1 BASIC Field Names File

Field names for BASIC screen images are generated by the EZFORMAT utility. The EZFORMAT utility enables you to change the generated field names to more meaningful values, and to set upper and lower bounds for acceptable input. To save the field names and definitions, the EZFORMAT Field Name File Definition screen (Figure 9-7) appears. Specify the file, library and volume in which to store the field definitions and press ENTER (if you are using an existing field name file) or PF2 (if you want to create a new field name file).

The Field Name Definition file is especially useful if you redefine a screen image by adding or deleting a modifiable field or change the screen image (fields are row and column sensitive). For example, if you created a screen image with 30 modifiable fields, and later added one field, you could call up this file and simply add the new field to the others; otherwise, you would have to respecify all of the field names.

```
***  MESSAGE 0001 BY EZFORM

                INFORMATION REQUIRED BY  PROGRAM  EZFORMAT
                          TO DEFINE FIELDS
                     ACTIVE SUBPROGRAM IS EZFORMAT

Information required to define field name file.


Please specify the location of the data file to save/initialize the
source and object field names.
Ranges for the BASIC or COBOL option will also be saved.

     FILE     = ******** IN LIBRARY  = ******** ON VOLUME   = ******


Press ENTER to use an existing field name file
or press PF2 to create a field name file.
```

Figure 9-7. EZFORMAT Field Name File Definition Screen

9.9.2 Defining the BASIC Data-Name and Range Fields

After you specify the file in which the fields are to be stored, the EZFORMAT utility displays the EZFORMAT BASIC Data-Name and Range Definition screen (Figure 9-8) through which you can alter the default-generated field definitions.

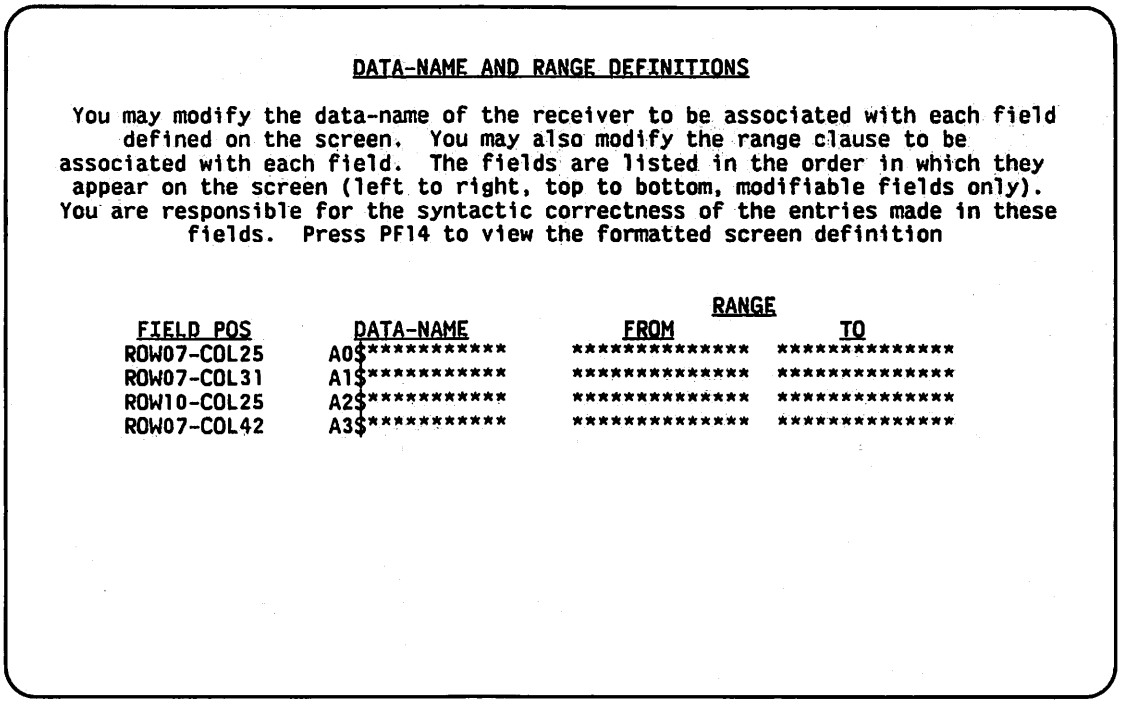


Figure 9-8. EZFORMAT BASIC Data-Name and Range Definitions Screen

The EZFORMAT BASIC Data-Name and Range Definitions screen displays the field position, data name, and default input range for each modifiable field defined in the screen format. You cannot modify the field position but you can change the data name and range to match fields used in the program into which the screen design is to be copied. From the EZFORMAT BASIC Data-Name and Range Field Definitions screen, you can

- Press ENTER to process subsequent EZFORMAT BASIC Data-Name and Range Definitions screens from the current screen.
- Press PF1 from the current screen to return to the first EZFORMAT BASIC Data-Name and Range Definitions screen to alter any previous field definitions.
- Press PF14 to view the screen display during processing of the EZFORMAT BASIC Data-Name and Range Definitions screen. Field definition resumes when you press ENTER from the design screen.
- Press PF16 after you finish defining the data names and ranges, to define the output file in which the screen is to be contained.

The modifiable fields on the EZFORMAT BASIC Data-Name and Range Definition screen(s) are described as follows:

Field	Description
DATA-NAME	Specify the field name by which the generated source code references the modifiable field. The data name for an alphanumeric field defaults to Xn$, where n varies from 0 to 9, for each X varying from A to Z. Thus, EZFORMAT generates up to 260 (A0$, A1$, ..., Z8$, Z9$) unique default character names. If you create more than 260 alphanumeric fields, you must change the corresponding data names for the excess fields to avoid overwriting earlier fields. In a similar way, integer and fractional fields default to data names of A0% through Z9%. Because the default numeric field name contains a % (indicating an integer field), you must change the data name for fractional numeric data. **NOTE** *VS BASIC supports 64-character data names, but data names that you assign through EZFORMAT cannot exceed 14 characters.*
RANGE	Specify the upper and lower bounds that determine acceptable input values. Default range values are displayed for those fields defined in the screen format. These ranges can be made more restrictive (or eliminated altogether), or upper and lower bounds can be applied to other fields by supplying values in the FROM and TO range fields. You can define only one set of range fields for each DATA-NAME field.

When you finish defining the data name and ranges, press PF16. EZFORMAT then prompts you to specify the file, library, and volume names for the source file that contains the formatted screen. The default value for the library name is obtained by concatenating your user ID with COPY (MMCOPY, for example). You can override this value if you want the source file stored in another library. After you name the output file, EZFORMAT creates the BASIC source file that corresponds to your screen design and field definitions, then terminates processing. EZFORMAT return codes are listed in Appendix B.

9.10 COBOL LANGUAGE-GENERATED OUTPUT

If you specify the COBOL option on the EZFORMAT Input Options screen (Figure 9-2) and save the generated output, EZFORMAT creates a COBOL source file that contains the Data Division definition of the screen image record. The record contains data names, data types, and initial values corresponding to the screen design. EZFORMAT does not generate Procedure Division statements to process this record; you must write the code in the main COBOL program into which the generated code is to be copied. When you use the source file with a COBOL program, the source file recreates the screen image at the workstation as part of the program and also reads any data entered into any modifiable fields.

9.10.1 COBOL Field Names File

Field names for COBOL screen images are generated by the EZFORMAT utility. The EZFORMAT utility enables you to change the generated field names to more meaningful values, and to set upper and lower bounds for acceptable input. To save the field names and definitions, the EZFORMAT Field Name File Definition screen appears (refer to Figure 9-7). Specify the file, library and volume in which to store the field definitions and press ENTER (if you are using an existing field name file) or PF2 (if you want to create a new field name file).

The field name file is especially useful if you redefine a screen by adding or deleting a modifiable field or change the screen image (fields are row and column sensitive). For example, if you created a screen image with 30 modifiable fields, and later added one, you could call up this file and simply add the new field to the others; otherwise, you would have to respecify all of the field names.

9.10.2 Defining the COBOL Source, Object, and Range Fields

After you specify the file in which the fields are to be stored, the EZFORMAT utility next displays the EZFORMAT COBOL Source, Object, and Range Field Definitions screen (Figure 9-9) through which you can alter the default-generated field definitions.

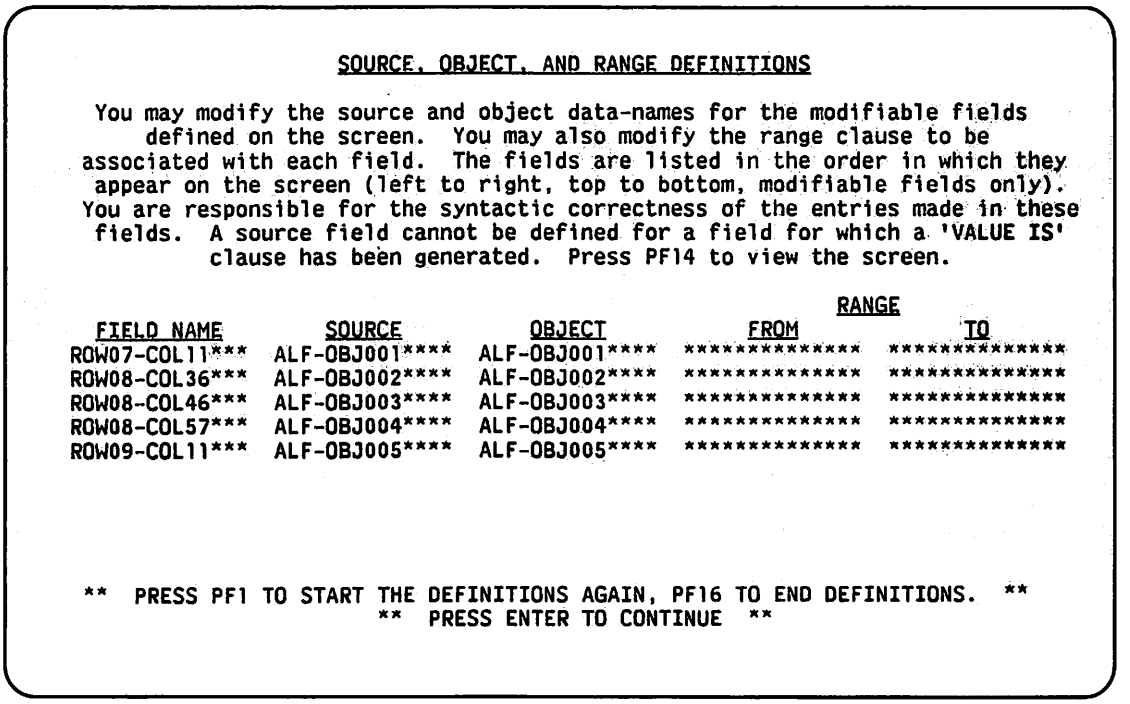


Figure 9-9. EZFORMAT COBOL Source, Object, and Range Field Definitions Screen

EZFORMAT generates default values for Field Name, Source, and Object fields. EZFORMAT also generates default values for the Range field for each modifiable field that you defined on your screen image design. You can modify the default values to correspond to fields used in the main program. The following options are available from the EZFOMAT COBOL Source, Object, and Range Definition screen:

- Press ENTER to proceed to subsequent EZFORMAT COBOL Source, Object, and Range Definitions screens from the current screen.
- Press PF1 from the current screen to return to the first EZFORMAT COBOL Source, Object, and Range Definitions screen to alter any previous field definitions.
- Press PF14 to view the screen display during processing of the EZFORMAT COBOL Source, Object, and Range Definitions screen. Field definition resumes when you press ENTER from the design screen.
- Press PF16 after you finish defining the data names and ranges to define the output file in which the screen is to be contained.

The modifiable fields on the EZFORMAT COBOL Source, Object, and Range Definitions screen(s) are described as follows.

Field	Description
FIELD NAME	Optionally, specify a name (used by the COBOL-generated source code) for a field's screen location. EZFORMAT supplies a default name in the form ROWxx-COLyy (where xx and yy are numeric row and column positions).
	NOTE *The name you specify in FIELD NAME must not duplicate a source name, an object name, or a COBOL reserved word.*
SOURCE	Optionally, specify a name for the Source field from which FIELD NAME obtains its initial value when the screen is displayed. EZFORMAT supplies a default name that reflects the data type and the field occurrence. For example, ALF-OBJOO1 reflects an alphanumeric modifiable field which is the first input request. NUM-OBJ005 reflects a numeric modifiable field, which is the fifth input request.
	NOTE *If you have supplied a default value for the modifiable field during screen definition, neither you nor EZFORMAT can assign a Source name. The default value specified during screen image definition is always displayed by the COBOL program.*
OBJECT	Optionally, specify a name for the Object field to which the value in FIELD NAME is moved when the ENTER key (or any defined PF key not in an ON clause of a DISPLAY AND READ statement) is pressed from the displayed screen. EZFORMAT supplies a default name constructed in the same manner as the Source field's default name.
RANGE	Specify the upper and lower bounds of values to be accepted as input by the COBOL program. The COBOL collating sequence determines the interpretation of the upper and lower bounds (refer to the *VS COBOL 74 Reference* for details). RANGE values are optional.

When you finish the source, object, and range definitions and press PF16, EZFORMAT prompts you to specify the file, library, and volume names for the source file that contains the formatted screen image. The library name defaults to the value obtained by concatenating your user ID with COPY (MMCOPY, for example), but you can override this value if you want the file stored in another library. After you name the output file, EZFORMAT creates the COBOL source file that corresponds to your screen image and your field definitions, then terminates processing. EZFORMAT return codes are listed in Appendix B.

9.11 RPG II LANGUAGE-GENERATED OUTPUT

If you specify the RPG II option on the EZFORMAT Input Options screen (Figure 9-2) and save the generated output, EZFORMAT creates an RPG II source file that displays the screen image that you designed. When you use the source file with an RPG II program, the source file recreates the screen image at the workstation as part of the program and also accepts any data entered into a modifiable field. In a subroutine, EZFORMAT defines and assigns initial values to fields corresponding to the modifiable fields defined on your screen image. The RPG II source code also produces calculations to enable (activate for processing) the ENTER and PF16 keys and to terminate the program when PF16 is pressed.

EZFORMAT prompts you to specify the file, library, and volume names of the RPG II source file to be generated. The library name defaults to the value obtained by concatenating your user ID with COPY, but you can override this value if you want to store it in another library. After you name the output file, EZFORMAT displays the EZFORMAT RPG II Source and Object Definitions screen (Figure 9-10).

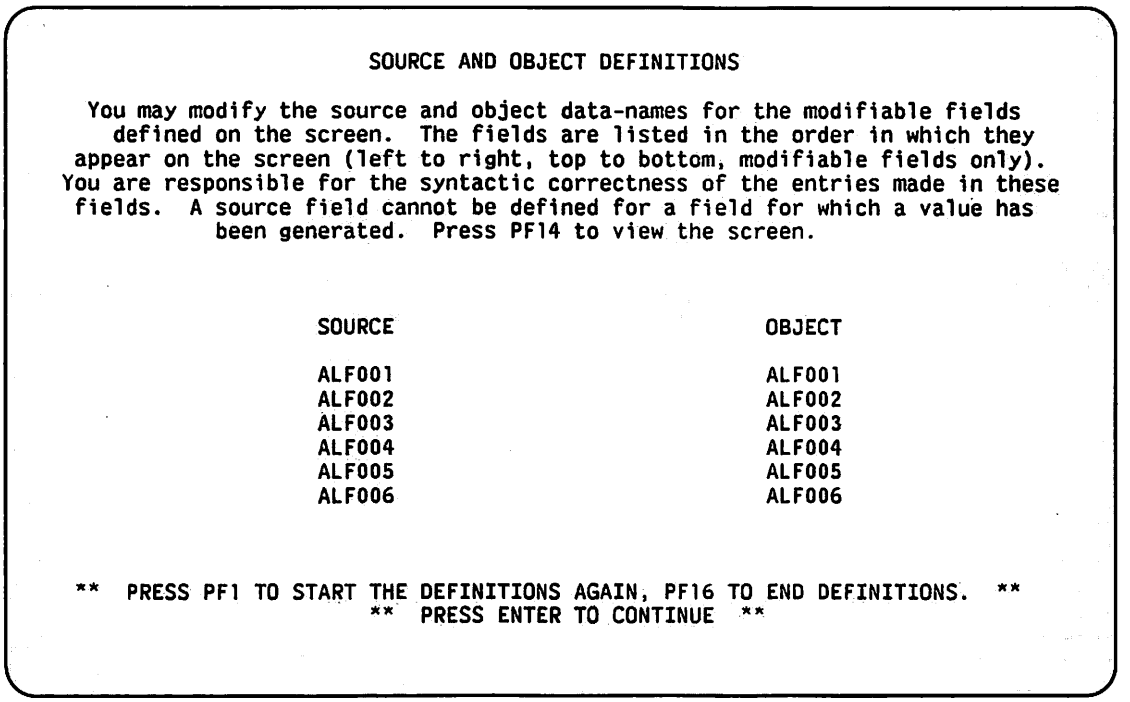


Figure 9-10. EZFORMAT RPG II Source and Object Definitions Screen

You can modify the default source and object field names on the EZFORMAT RPG II Source and Object Definitions screen (Figure 9-10) to correspond to field names used in the RPG II main program as follows:

- Press ENTER to reach subsequent EZFORMAT RPG II Source and Object Definitions screens from the current screen.
- Press PF1 from the current to return to the first EZFORMAT RPG II Source and Object Definitions screen to alter any previous field definitions.

- Press PF14 to view the screen display during processing of the EZFORMAT RPG II Source and Object Definitions screen. Variable definition resumes when you press ENTER from the design screen.
- Press PF16 after you finish defining the data names and ranges to define the output file in which the screen is to be contained.

The EZFORMAT RPG II Source and Object Definitions screen involves two fields, described as follows:

Field	Description
SOURCE	Optionally, specify the Source field from which the field's initial value is obtained when the screen is displayed. EZFORMAT supplies a default value that reflects the data type and modifiable field occurrence. For example, ALF001 reflects an alphanumeric field, which is the first input request on the screen. NUM005 indicates the screen's fifth input request, which is a numeric modifiable field. You cannot define a source field name if the screen format specifies an initial value for the field.
OBJECT	Optionally, specify the field that receives the value entered in the screen format field. EZFORMAT supplies a default name constructed in the same manner as the Source default name.

NOTE
VS RPG II supports 6-character field names, but source and object names that you assign through EZFORMAT cannot exceed 14 characters.

When you finish the source and object definitions and press PF16, EZFORMAT prompts you to specify the file, library, and volume names for the source file. The library name defaults to the value obtained by concatenating your user ID with COPY (MMCOPY, for example), but you can override this value if you want the file stored in another library. After you name the output file, EZFORMAT creates the RPG II source file that corresponds to your screen design and your field definitions, then terminates processing. EZFORMAT return codes are listed in Appendix B.

9.12 MENU-GENERATED OUTPUT

If you specify the Menu option on the EZFORMAT Input Options screen (Figure 9-2) and save the generated output, the EZFORMAT utility creates an executable program that corresponds to your screen image and the menu described by the EZFORMAT Menu Generator screen (Figure 9-4) selections. EZFORMAT also creates an Assembly language or RPG II source file for the menu program and a source listing print file that results from the menu assembly or compilation. You can further customize the menu program by editing the generated source file.

After you choose to save the generated output, EZFORMAT prompts you to specify the language in which EZFORMAT generates the menu program. If you press PF2, EZFORMAT generates the menu in Assembly language; if you press PF3, EZFORMAT generates the menu in RPG II. Menu programs created in RPG II can perform the same functions as menus created in Assembly language.

EZFORMAT then prompts you to specify the file, library, and volume names for the Assembly language or RPG II source file to be generated. The library name defaults to the value obtained by concatenating your user ID with COPY (MMCOPY, for example) You can override this value if you want to store it in another library. Next, EZFORMAT prompts you to specify the file fields for the menu object code.

NOTE
If user programs are associated with PF keys during menu assignment, place the menu program in the same library as the user programs. For RPG II menus, the CEXIT and LINK user subroutines must reside in the USERSUBS library on the system volume so that EZFORMAT can link them to the menu program. The print file is automatically assigned the same file name as the source file, and is placed in your print library.

Processing ends when the menu program is created. EZFORMAT return codes are listed in Appendix B.

9.12.1 Running the Menu Program

To run a menu program, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify the name of the menu program, the library and volume in which the program resides, and press ENTER.

The screen image that you created through EZFORMAT is displayed when the menu program is run. The functions, programs, and procedures are executed by the menu when the user presses the PF keys that were assigned when the menu was created. Control returns to the menu when a program terminates processing.

The USERPROG function does not retrieve the menu user's default RUNLIB and RUNVOL values. The USERPROG function supplies the file fields of the most recently run USERPROG as a default value. A message is displayed in Row 24 of the workstation screen that indicates the status of the most recently completed menu operation. The message states whether the program was cancelled or completed normally. An error is indicated if the program specified by the menu cannot be located.

Menu processing ends if you select and press a PF key that is associated with the EXIT or LOGOFF functions. If no EXIT or LOGOFF was assigned, you must cancel processing by pressing the HELP key, then PF16 to terminate processing.

9.13 MODIFYING AN EZFORMAT SCREEN IMAGE

With EZFORMAT, you can modify a previously defined screen image, provided that the screen contents have been saved in a separate file (refer to Section 9.7 for details on saving a screen design). The screen contents file for the Menu option also contains previous menu assignments; therefore, you can also modify the menu functions.

Press PF3 from the EZFORMAT Input Options screen (Figure 9-2) to modify a previously designed screen. EZFORMAT then prompts you to specify the file, library, and volume names of the file that contains the screen contents or menu assignments (for the Menu option). The input library name defaults to your user ID concatenated with SAVE (MMSAVE, for example); you can override the default value if you saved the file in a different library. You can then modify the screen contents and menu assignments.

9.14 A SAMPLE EZFORMAT PROCEDURE

Although screen image definition in EZFORMAT is by nature an interactive process, a significant amount of workstation interaction in EZFORMAT processing can be eliminated or controlled. You can select the function, file creation options, and output file locations through a procedure. If the screen image has been previously defined, all workstation interaction, except BASIC, COBOL, or RPG II field definition, can be eliminated or controlled. A complete listing of the EZFORMAT GETPARM requirements is given in Appendix A. Refer to the *VS Procedure Language Reference* for details about Procedure language syntax.

Although the EZFORMAT Input Options screen does not request information through a GETPARM, the EZFORMAT utility contains a hidden GETPARM, identified by the Options parameter reference name (prname), that requests the same information as the EZFORMAT Input Options screen. The EZFORMAT function is selected through the LANGUAGE keyword; PF2 effects the creation of a new screen format, PF3 indicates that previously created screen contents are to be used as input to EZFORMAT.

The hidden GETPARM enables you to bypass screen definition completely through the PROCEDUR keyword value which is only considered when EZFORMAT processes previously created screen contents. The default value (YES) bypasses the screen definition phase of EZFORMAT and continues processing with the EZFORMAT Output Options screen. However, if the Menu option is selected, the EZFORMAT Menu-Generator screen is not bypassed and is displayed immediately before the EZFORMAT Screen Manipulation Options screen.

The following sample procedure runs the EZFORMAT utility, selects the COBOL function, and creates a new screen format. The procedure saves the screen contents and generated COBOL source file as output, and names the files containing the screen contents and COBOL source code (accepting default library names). All workstation interaction is eliminated, except for the actual screen definition and COBOL field definition processes.

```
PROCEDURE
    RUN EZFORMAT
        ENTER OPTIONS 2, LANGUAGE=COBOL
        ENTER FUNCTION 4
        ENTER SCREEN FILE=DATA, VOLUME=SYSTEM
        ENTER COPYLIBR FILE=DATA, VOLUME=SYSTEM
RETURN
```

CHAPTER 10
THE FASTLINK UTILITY

10.1 INTRODUCTION

The FASTLINK utility enables you to specify program files that you want the system to keep open permanently. System performance improves when frequently-run application and utility program files are kept open because there is a reduction in Volume Table of Contents (VTOC) input/output (I/O) and control block allocations and de-allocations. For systems with large main memory space, FASTLINK can also reduce paging by extending the possibility for sharing the executable code in main memory.

The FASTLINK utility reads the File Location and Use Block (FLUB) chain to display an open status. The LINK Supervisor Call (SVC) then opens a file permanently, or resets an already opened file (removes from a permanently open state). You can perform the following functions with FASTLINK:

- Set and list program files that are permanently open
- Reset permanently open program files (individually or as a group)
- Monitor the use of programs that are permanently open
- Create, edit, and store a program-name file that contains the names of the programs that you set as permanently open or reset from a permanently open state
- Set programs, listed in a previously specified program-name file, as permanently open (individually or as a group)
- Reset programs that are listed in a previously specified progam-name file (individually or as a group)
- Use GETPARM screens to provide a background procedure through VS Procedure language when you use FASTLINK routinely
- Use a GENEDIT option that enables you to set programs as permanently open at IPL time.

10.1.1 Using FASTLINK

The FASTLINK utility should be used to include only those programs or utilities that are run frequently. While the computer system benefits from any frequently-run program that uses FASTLINK, you achieve the greatest performance gains on short-lived programs.

NOTE
Short-lived programs are programs that run only a second or so in the computer. The word processing (WP) program can take from a couple of minutes to a number of hours depending on how long a person uses WP. The WP program is considered to be a long-lived program.

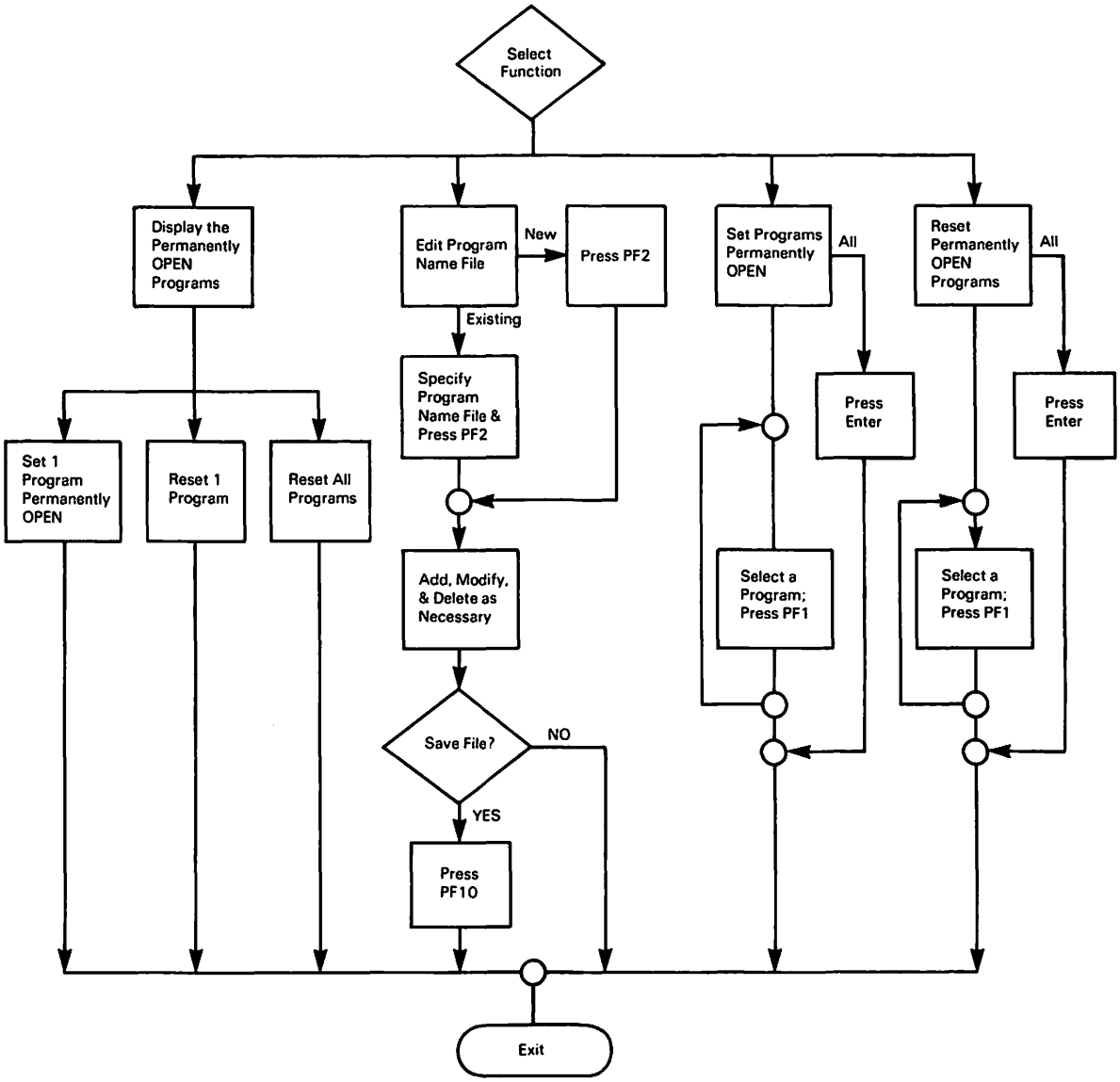


Figure 10-1. FASTLINK Processing

The COBOL compiler program is one example to illustrate the benefits of FASTLINK on frequently-run programs. If necessary, the executable code for the COBOL compiler is paged into memory each time a program is compiled. When the COBOL compiler is finished, the memory that contained the executable code is immediately returned to the system and made available for other processes unless another user has started compiling another COBOL program in the meantime, thereby using the executable code already in main memory. However, if that user needed the COBOL compiler 1 millisecond after the compiler had finished the first job, the compiler must be paged back into main memory. (For more information about paging, refer to Section 7.13 of the DISKINIT utility.)

The disk I/O and central processor (CP) savings from using FASTLINK in these cases are proportional to the size of the executable code and frequency of the use of executable code. FASTLINK enables a frequently-run program to remain in main memory (as long as the pages of the program are not considered by the operating system as the "best candidate" for removal, i.e., paged out).

By allowing the program to remain in main memory (using the previous example), the COBOL compiler would not have to be reloaded page by page each time you wanted to use it. On systems with sufficient memory, pages may not be selected for replacement for several minutes after completion of a job.

10.2 FASTLINK PROCESSING

Figure 10-1 shows an overview of FASTLINK processing.

To run the FASTLINK utility, press Program Function (PF) key 1 (RUN Program or Procedure) from the VS Command Processor menu. Specify FASTLINK in the Program field and press ENTER.

The FASTLINK Function Definition screen (Figure 10-2) prompts you to choose a particular FASTLINK function:

- You can set a single program as open through the Display Perm-Open Pgms (PF1) option discussed in Section 10.2.1.
- You can set groups of programs as open through the Edit "Program-Name" File (PF2) and Set Programs Perm-Open (PF3) options discussed in Sections 10.2.2 and 10.2.3.
- You can reset (i.e., remove from a state of being permanently open) individual programs or groups of programs through the Reset Perm-Open Programs (PF4) option discussed in Section 10.2.4.

You can also control permanently open files in the following ways:

- You can set files open permanently by using the GENEDIT option. (Refer to the *VS System Administrator's Reference* for details on the GENEDIT utility.)
- You can control FASTLINK processing through the VS Procedure language.

```
***  MESSAGE 1     BY FSTLNK

                INFORMATION REQUIRED BY PROCEDURE RK1LOGON
                          TO DEFINE FUNCTION
                       ACTIVE PROGRAM IS FASTLINK

<<<< FASTLINK - VS "Permanently Open Program File" Utility version x.xx.xx >>>>


        For option (2) enter FILE/LIB/VOL names to edit an existing file,
           or leave FILE/LIB/VOL blank to create a new file.

        For options (3) AND (4) FILE/LIB/VOL names must be provided.


        FILE     = FASTFILE LIBRARY  =  @SYSTEM@ VOLUME  = SYSTEM


(1) Display perm-open pgms      (2) Edit "program-name" file
                                (3) Set programs perm-open
                                (4) Reset perm-open programs     (16) Exit
```

Figure 10-2. Sample FASTLINK Function Definition Screen

The options on the FASTLINK Function Definition screen are described as follows:

PF Key	Option	Description
1	Display Perm-Open Pgms	Press PF1 to display all permanently opened program files with the current usage status of each program.
2	Edit "Program-Name" File	Specify the file, library, and volume names of the file that you want to edit and press PF2. If you want to create a new file, leave the file, library, and volume fields blank and press PF2.
3	Set Programs Perm-Open	Specify the file, library, and volume names of the program name file that you want to set as permanently open and press PF3.
4	Reset Perm-Open Programs	Specify the file, library, and volume names of the program name file whose permanently open status is to be changed and press PF4.
16	Exit	Press PF16 to terminate FASTLINK processing.

10.2.1 Displaying Permanently Open Programs

If there are already program files set as permanently open, the FASTLINK Open Files Display screen lists those files when you press PF1 from the FASTLINK Function Definition Screen. If you have not set any Function Definition screen, an emply FASTLINK Open Files Display screen appears.

The FASTLINK Open Files Display screen (Figure 10-3) shows files that are in a permanently open state. The first three fields give the names of the file, library, and volume of opened program files. The fourth field (Total Usage) keeps a count of the number of times the system has used a program since the program was permanently opened. The fifth field (Active Users) displays the number of tasks that are currently using a program. The Total Usage and Active Users fields enable you to monitor the use of your permanently open programs.

```
<<<< FASTLINK - VS "Permanently Open Program File" Utility version x.xx.xx >>>>
                                                    Perm-Open File Display Mode
      File                Library          Volume      Total Usage Active Users
-------------------------------------------------------------------------------
  17  BASIC               @SYSTEM@         SYSTEM            0            0
  18  FILEDISP            USERAIDS         SYSTEM            0            0
  19  MOUNT               USERAIDS         SYSTEM            0            0
  20  BACKUP              @SYSTEM@         SYSTEM            0            0
  21  @SYSCPR@            @SYSTEM@         SYSTEM           97           10
  22  @INFO               @SYSTEM@         SYSTEM            0            0
  23  @PROC@              @SYSTEM@         SYSTEM           52            8
  24  LINKER              @SYSTEM@         SYSTEM            1            0
  25  WE1EDIT             @SYSTEM@         SYSTEM            7            1
  26  DISKINIT            @SYSTEM@         SYSTEM            1            0
  27  ASSEMBLE            @SYSTEM@         SYSTEM            7            0
  28  @OPER@              @SYSTEM@         SYSTEM           40            5
  29  EDITOR              @SYSTEM@         SYSTEM            4            1
  30  DISPLAY             @SYSTEM@         SYSTEM            9            1
  31  COPY                @SYSTEM@         SYSTEM           10            0
  32  SCS                 @SCSLIB@         S63000           14            3

(2) First Screen          (4) Previous Screen     (7)  Reste File at Cursor
(3) Last Screen           (5) Next Screen         (8)  Reset All Files
                          (6) Set One Perm-Open   (16) Exit To Main Menu
```

Figure 10-3. Sample FASTLINK Open Files Display Screen

NOTE

In Figure 10-3, COPY is an example of a short-lived program in that it has a relatively high usage count and 0 active use counts. In other words, COPY is used frequently but not constantly but by having the executable code already in memory, COPY does not have to be constantly paged back in. Paging operations are saved by using FASTLINK for programs whose usage count frequently goes to 0.

You can set a program file as open permanently from this screen by pressing PF6 (Set One Perm Open). When you press PF6, the FASTLINK Define Open File screen (Figure 10-4) appears.

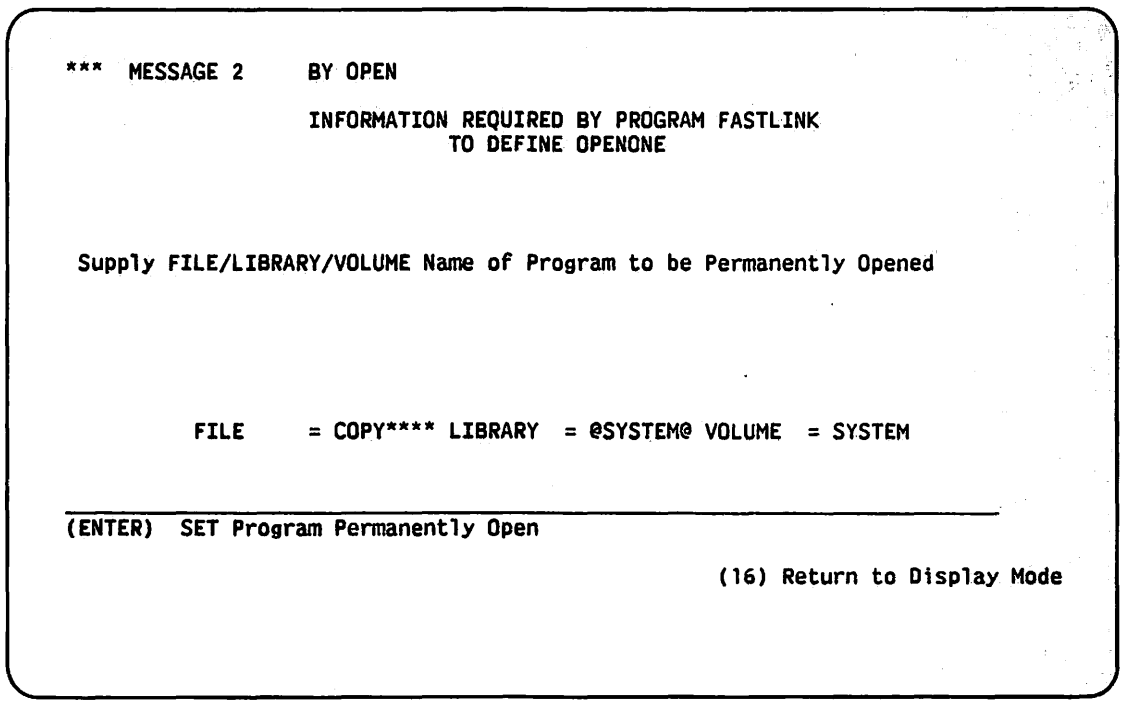


Figure 10-4. Sample FASTLINK Define Open File Screen

From the FASTLINK Define Open File screen (Figure 10-4), specify the file, library, and volume names of the file that you want to place into a permanently open state, then press ENTER. When you press ENTER, the FASTLINK Open Files Display screen (Figure 10-3) reappears and lists the file that you just opened. You can return to the FASTLINK Open Files Display screen without opening any files by pressing PF16.

PF7 on this screen resets the Open status and clears the Total Usage count for the program file line on which the cursor is positioned. PF8 resets the Open status and clears the Total Usage count for all of the permanently opened programs. When you reset the programs, the FASTLINK Open Files Display screen appears with no files listed.

10.2.2 Creating, Editing, and Storing the Program-Name File

A program-name file contains the names of a group of programs in one file that you want to place into a permanently open state. The Edit "Program-Name" File function (PF2) enables you to perform the following functions:

- Create a file that contains the program names that you want opened permanently.
- Read records from an existing file of program names.

To create a new program-name file, leave the file, library, and volume fields blank and press PF2. The Program-Name File screen appears with no files listed. Figure 10-5 shows a sample Program-Name File screen with files listed.

```
<<<< FASTLINK - VS "Permanently Open Program File" Utility Version x.xx.xx >>>>
                                                    "Pgm-name" File Edit Mode
      File                Library          Volume
-------------------------------------------------------------------------------
  17  COPY****            ********         ******
  18  DISPLAY*            ********         ******
  19  EDITOR**            ********         ******
  20  @OPER@**            ********         ******
  21  ASSEMBLE            ********         ******
  22  SCS*****            @SCSLIB@         S63000
  23  WE1EDIT*            ********         ******
  24  D*******            ********         ******
  25  @PROC@**            ********         ******
  26  MWSLOAD*            ********         ******
  27  @SYSCPR@            ********         ******
  28  ********            ********         ******
  29  ********            ********         ******
  30  ********            ********         ******
  31  ********            ********         ******
  32  ********            ********         ******

(2) First Screen          (4) Previous Screen      (9)  Modify Program Names
(3) Last Screen           (5) Next Screen          (10) Save "Pgm-Name" File
                                                   (16) Exit To Main Menu
```

Figure 10-5. Sample Program-Name File Screen

To enter program names into the file that is to be created, press PF9 (Modify Program Names) and specify the file, library, and volume names of the program that you want opened permanently. This program-name file can contain up to 128 records (8 screens of 16 entries each). If you create a program-name file with more programs that can fit on one screen, press PF5 to go to the next screen. You can display any one of the Program-Name File screens by pressing PF keys 2 through 5 for the desired screen (where applicable):

PF Key	Function
2	First Screen
3	Last Screen
4	Previous Screen
5	Next Screen

After you have entered all the program-name files that you want to set permanently open, press PF10 (Save "Program-Name" File) to retain the entries in the file. If you subsequently modify the entries, press ENTER to retain those changes, and then press PF10 to save them.

NOTE

If you do not press PF10 to save the program-name entries, a message appears, warning you that you have not saved your file. FASTLINK then enables you to return to the Program-Name File screen in Edit mode or exit to the FASTLINK Function Definition screen without saving the file.

If you save your program-name file by pressing PF10, the FASTLINK Disk Assignment screen (Figure 10-6) appears.

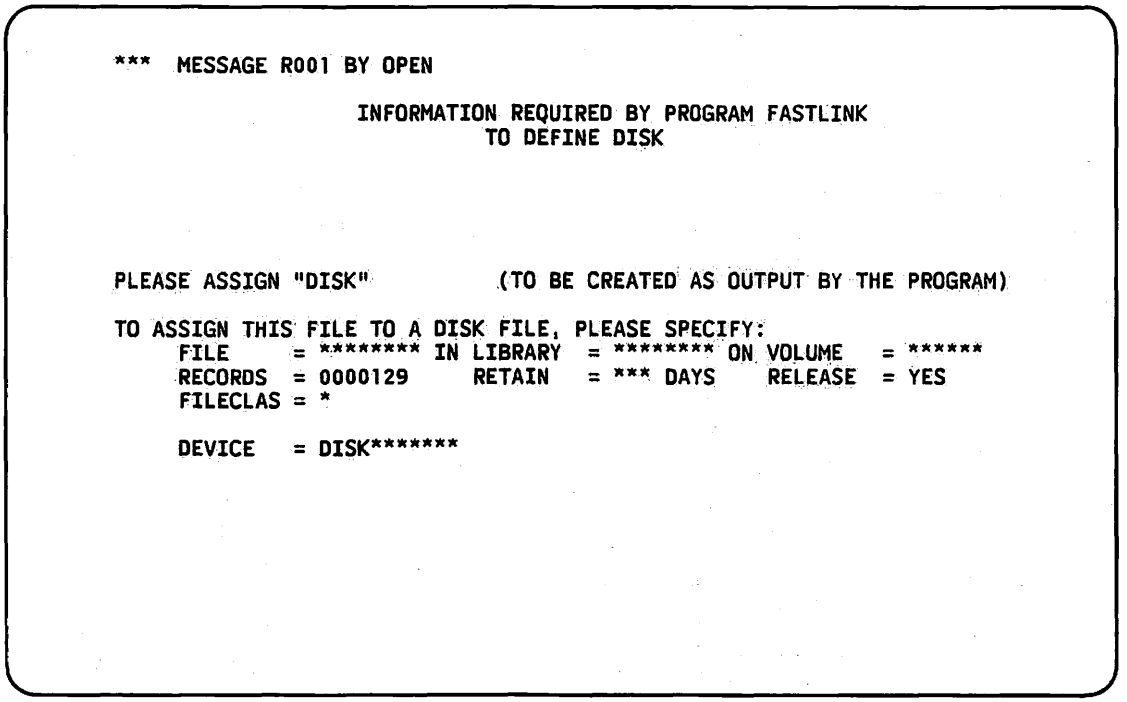


Figure 10-6. Sample FASTLINK Disk Assignment Screen

The FASTLINK Disk Assignment screen (Figure 10-6) prompts you to assign a file name to the program-name file. You are also prompted to specify the library and volume on which the program-name file is to reside. The other fields are described as follows:

Field	Description
Records	By default, the Records field indicates the number of records that your program-name file spans. You can specify a higher number if you want to allocate more space to the file on the disk. To increase system response further, you may want to locate a permanently open file near the most active areas on the disk. If you do not allocate more space than the file requires, space around the file may be used by other resources; the file is not able to accept any more programs in the file (and remain in the same area on the disk). If you want to increase your program-name file at a later date, you may have to do so at the cost of moving the file to another part of the disk. If you anticipate growth of your program-name file, you can allocate more records than what the default value allots. Increase the size of the output file by specifying the number of records to the number that you anticipate the file needs to support future updates. Also, specify NO in the Release field so that the extra space that you allocate is not released for other use.
Retain	Specify the number of days that the file is protected from deletion in the Retain field. A file can be protected for a maximum of 999 days and a minimum of zero days.

Field	Description
Fileclas	Specify the protection class of the output file. The protection class determines which users can access the file. The following file protection classes are available: #, $, @, blank, and A to Z. (For more information on file class options, refer to the *VS System User's Introduction*.)
Release	Specify (YES or NO) whether unused storage extents, previously allocated for data, can be released for use by other files. Specify NO if you have allocated more space than the file requires and want the file to keep that space for future updates to it. If you specify YES, it will not matter that you allocated more records than the default value; the extra space will be freed for use. The Release field defaults to YES.
Device	Specify the device on which the output file is to be located. The default value for the Device field is DISK.

After you specify the fields on the FASTLINK Disk Assignment screen, press ENTER. The Program-Name File screen (Figure 10-5) appears and you can either modify (PF9) the program-name file that you just created or exit (PF16) to the FASTLINK Function Definition screen.

10.2.3 Setting Program-Name Files Permanently Open

To place a program-name file in a permanently open state, specify the file, library, and volume names of the program-name file, and press PF3 from the FASTLINK Function Definition screen (Figure 10-2). When you press PF3, a screen appears giving you the following options for the program-name file that you specified:

- Press ENTER to set all the programs in the program-name file as open permanently.
- Press PF1 to select individual programs from the program-name file to be permanently open.

If you press PF1, the FASTLINK Set Open Programs screen (Figure 10-7) appears. Type any nonblank character in the space provided to the left of each program that you want to set to permanently open. When you have selected all the programs that you want opened permanently, press PF1 again to process the selected files. You can return to the FASTLINK Function Definition screen without setting programs by pressing PF16 from the FASTLINK Set Open Programs screen.

```
<<<< FASTLINK - VS "Permanently Open Program File" Utility version x.xx.xx >>>>
                                                    Set Selected Pgms Perm-Open
Type a nonblank character next to each program that you want Permanently-Opened

          ■ PROGRAM COPY****  IN ********  ON ******
          ■ PROGRAM DISPLAY*  IN ********  ON ******
          ■ PROGRAM EDITOR**  IN ********  ON ******
          ■ PROGRAM @OPER@**  IN ********  ON ******
          ■ PROGRAM ASSEMBLE  IN ********  ON ******
          ■ PROGRAM SCS*****  IN @SCSLIB@  ON S63000
          ■ PROGRAM WE1EDIT*  IN ********  ON ******
          ■ PROGRAM LINKER**  IN ********  ON ******
          ■ PROGRAM @PROC@**  IN ********  ON ******
          ■ PROGRAM MWSLOAD*  IN ********  ON ******
          ■ PROGRAM @SYSCPR@  IN ********  ON ******
          ■ PROGRAM ********  IN ********  ON ******
          ■ PROGRAM ********  IN ********  ON ******
          ■ PROGRAM ********  IN ********  ON ******
          ■ PROGRAM ********  IN ********  ON ******
--------------------------------------------------------------------------------
 (1) Process Files Selected on this Screen

                                                    (16) Return to Main Menu
```

Figure 10-7. Sample FASTLINK Set Open Programs Screen

10.2.4 Resetting Permanently Open Programs

To reset programs in a program-name file, specify the file, library, and volume names of the program-name file that you want to reset and press PF4 from the FASTLINK Function Definition screen. The reset option removes the program file from the state of being permanently open. The reset option does not close a program file automatically but rather the program file is able to close as a result of the reset option. When you press PF4 after specifying a program-name file, a screen appears that prompts you to choose one of the following options for the program-name file that you specified:

- Press ENTER to reset Open status for all of the programs in the program-name file.
- Press PF1 to reset Open status for specific programs in the program-name file.

If you press PF1, the FASTLINK Reset Open Program screen appears showing all of the programs in the program-name file. You can reset the status of any of the open files displayed. To reset files, type a nonblank character to the left of each program that you want to reset. Press PF1 again to process the selected files (and continue to the next screen if all the listed programs did not fit on the first screen).

After you reset the programs in the program-name file, you are returned to the FASTLINK Function Definition screen (Figure 10-2).

NOTE

When you create a program-name file, you must use the FASTLINK utility; you cannot use any other utility. When a program-name file is created through the FASTLINK utility, the file is formatted to be recognized as a file that contains the names of programs that may be opened permanently for processing. Although a program-name file must be created by FASTLINK, a correctly formatted program-name file can be copied or telecommunicated into your files. If you do not use correctly formatted files, the operating system cancels the program.

10.3 AUTOMATICALLY SETTING PERMANENTLY OPEN PROGRAMS AT IPL

A program-name file can be set to automatically open the programs listed inside the program-name file at every initial program load (IPL). To do this, simply place the program-name file in the system configuration file (@CONFIG@ in library @SYSTEM@) using the GENEDIT utility. (Refer to the *VS System Administrator's Reference* for details on the GENEDIT utility.)

10.4 USING FASTLINK ON YOUR SYSTEM

To determine the best use of FASTLINK for your system, perform the following steps:

1. Indefinitely open a number of program files (40-50) that you believe are frequently used.
2. Monitor the use of the programs by checking the FASTLINK Open Files Display screen's Total Usage and Active Users count (discussed in Section 10.2.1). (Short-lived programs typically have a high use count but a small number of active users.)
3. Close programs that your system does not use frequently, and open others.
4. After you have determined which programs are used frequently through monitoring the use of the programs, create program-name files (refer to Section 10.2.2) grouping those files.
5. If you want the same set of files used every day, put these in a program-name file. Put the program-name file in the system configuration using GENEDIT utility so that it is run every time the system is IPLed.
6. If you alternate sets of applications or utilities, group these into program-name files. Use procedures to set and reset the program-name files to accomplish different tasks.

NOTE
You cannot delete, rename, or patch permanently opened files. Also, you cannot dismount the volume on which permanently opened program files reside. You must first close the files to perform these functions on them.

10.5 SAMPLE FASTLINK PROCEDURES

You can control FASTLINK processing through the VS Procedure language. A list of FASTLINK GETPARMs is provided in Appendix A. Refer to the *VS Procedure Language Reference* for details about procedure syntax. In this procedure, FASTLINK calls upon Function 3 (Set programs in a program-name file permanently open) from the FASTLINK Function Definition screen (Figure 10-2) and supplies the file, library, and volume names. The second ENTER statement exits the program.

```
PROCEDURE SETFAST
    RUN FASTLINK
        ENTER FUNCTION 3
        ENTER SELECT FILE=X, LIBRARY=Y, VOLUME=Z
        ENTER FUNCTION 1
RETURN
```

The following procedure resets the named files by calling upon Function 4 (Reset programs in program-name file permanently open) from the Function Definition screen.

```
PROCEDURE RESETFAST
    RUN FASTLINK
        ENTER FUNCTION 4
        ENTER SELECT FILE=X, LIBRARY=Y, VOLUME=Z
        ENTER FUNCTION 16
RETURN
```

You can combine SETFAST and RESETFAST if the program-name files that you want to set permanently open change frequently. For example, the following program resets one program-name file and sets another:

```
PROCEDURE RESETANDSET
    RUN FASTLINK
        ENTER FUNCTION 4 FILE=X, LIBRARY=Y, VOLUME=Z
        ENTER SELECT
        ENTER FUNCTION 3 FILE=A, LIBRARY=B, VOLUME=C
        ENTER SELECT
        ENTER FUNCTION 16
    RETURN
```

CHAPTER 11
THE FLOPYDUP UTILITY

11.1 INTRODUCTION

You can use FLOPYDUP to create several copies of a single diskette or to back up a diskette to hard disk. Because FLOPYDUP simply transfers diskette image files to diskettes on a block-by-block basis, it can also duplicate non-VS diskettes. (Image files contain byte-by-byte copies of entire diskettes.)

NOTE
Only diskettes previously initialized as nonlabeled (NL) can receive FLOPYDUP diskette output. Any data residing on an output NL diskette (such as word processing documents) are destroyed when FLOPYDUP transfers output data to the diskette.

You can perform the following functions with the FLOPYDUP utility:

- Copy the contents of a diskette to another diskette through the Duplicate function.
- Copy the contents of a diskette to a disk image file through the Copy function.
- Copy the contents of a disk image file to a diskette through the Generate function.
- Support VS Procedure language control of workstation interaction.

Figure 11-1 shows an overview of FLOPYDUP processing.

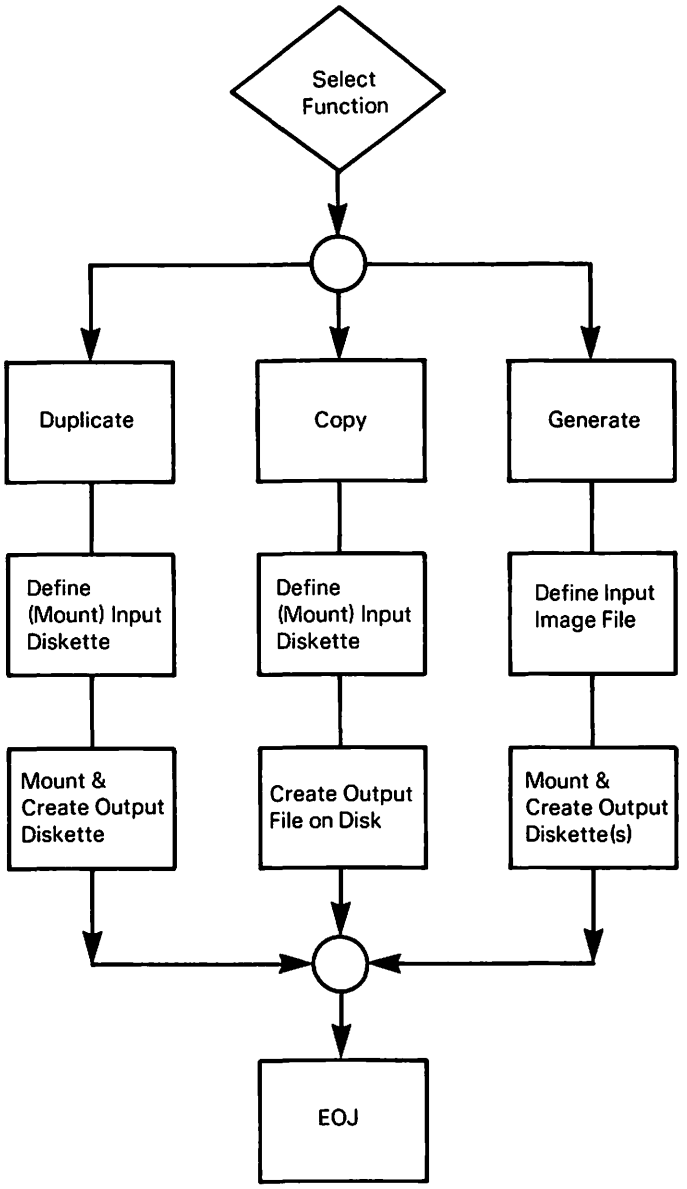


Figure 11-1. FLOPYDUP Processing

11.2 SELECTING A FUNCTION

The first screen that FLOPYDUP displays at your workstation is the FLOPYDUP Function Selection screen (Figure 11-2). Select D (Duplicate a diskette), C (Copy a diskette to a disk image file), or G (Generate a diskette from a disk image file). You can cancel processing by pressing PF16.

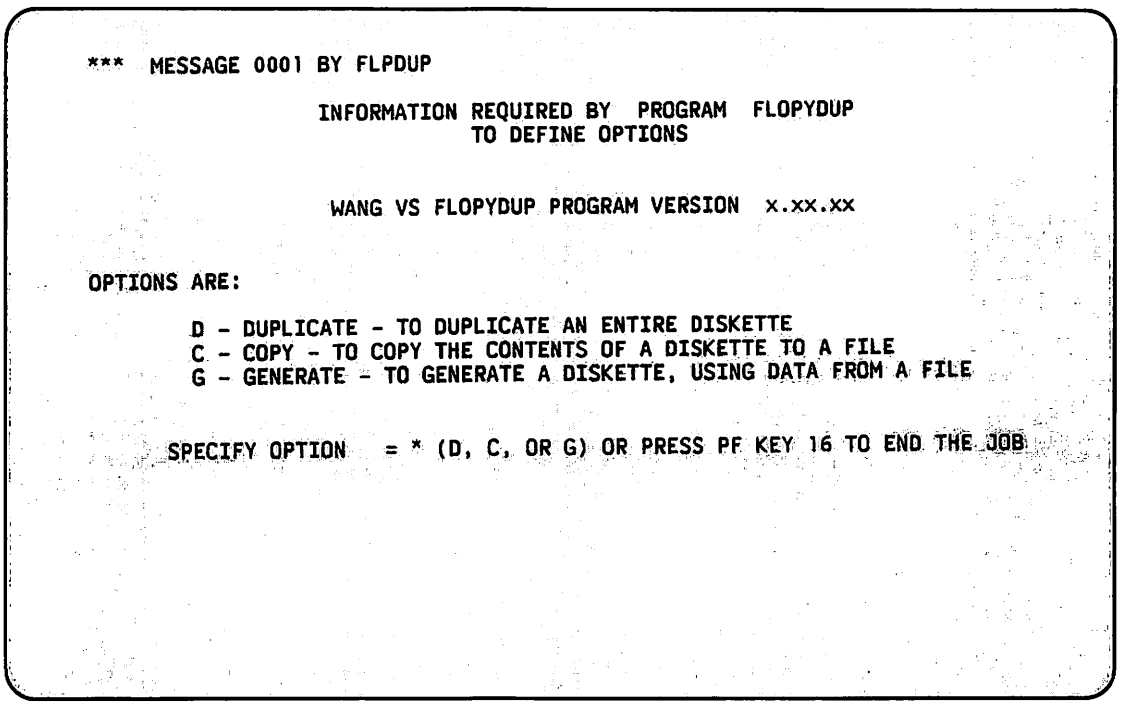


Figure 11-2. FLOPYDUP Function Selection Screen

11.3 THE FLOPYDUP DUPLICATE FUNCTION

If you specify D (Duplicate) on the FLOPYDUP Function Selection screen, the FLOPYDUP Input Diskette Definition Screen (Figure 11-3) prompts you to specify the name, label type, and device (diskette drive) of the diskette that you want to duplicate. If you have not mounted the diskette, FLOPYDUP automatically invokes the mount routine. The diskette can be mounted in Shared or Exclusive mode.

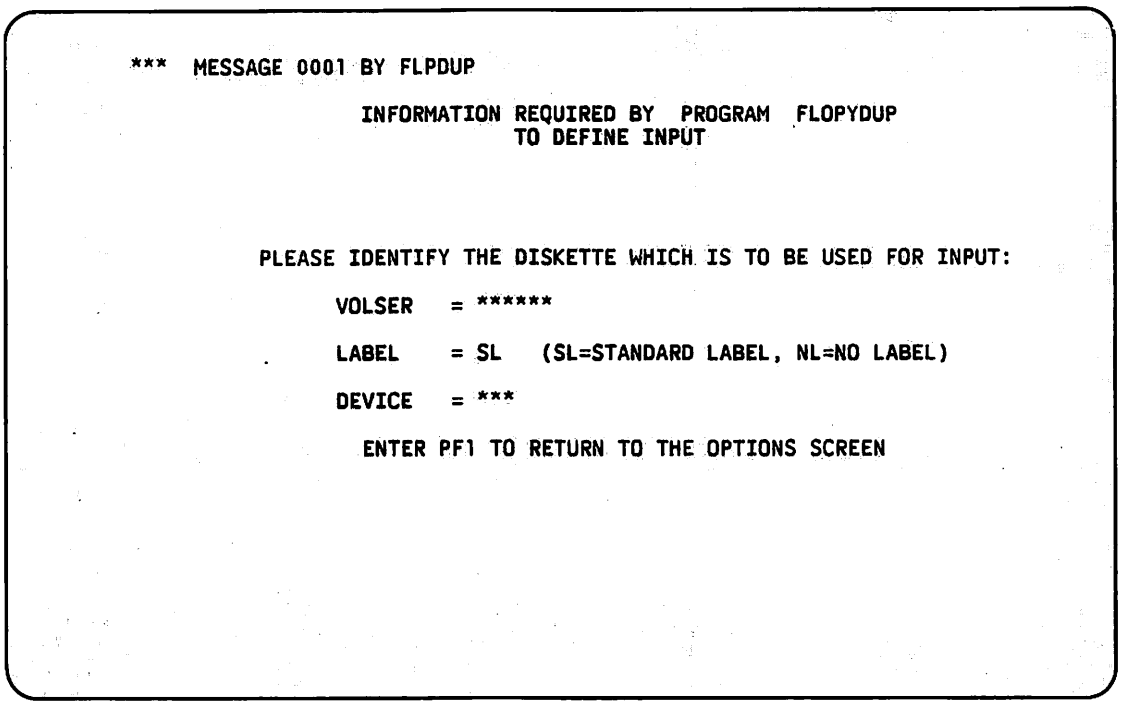


Figure 11-3. FLOPYDUP Input Diskette Definition Screen

FLOPYDUP copies a diskette onto the system disk, enables you to mount another NL diskette, and then copies the system disk's copy onto the new diskette. Even if the original diskette is not completely full, the program creates a complete image file (containing all the data of the diskette) in a work (temporary) file. The image file is always the size of a full diskette. The work file is on the system disk. The entire diskette is duplicated within a single work file, including the volume table of contents (VTOC) (if present), all libraries, and all files.

After the diskette image is in a work file, enter the name of the output NL diskette and the device number of the drive on which the diskette is mounted. You can use the same drive in which you copied the input diskette to mount the output diskette since the input diskette is no longer required. Any diskette already mounted in the specified drive is automatically dismounted by the program.

After you mount the new output diskette, the program copies the input diskette image file from the work file onto the output diskette. When the copying process is finished, the End-of-Job screen appears showing the results of the FLOPYDUP operation.

The End-of-Job screen enables you to end or restart FLOPYDUP processing, or to create another duplicate diskette with the same contents. All output diskettes contain exact, duplicate images of the input diskette.

NOTE

FLOPYDUP does not normally logically dismount a newly created diskette. However, if a second (or subsequent) duplicate diskette is generated, FLOPYDUP dismounts the first (or previous) diskette to enable you to use the same diskette drive to generate the second diskette. You must logically dismount the last generated diskette yourself through the Manage Devices function (PF6) from the Command Processor menu as you would if only one diskette were being duplicated.

Because the diskette is originally mounted as a nonlabeled (NL) diskette, the output diskette cannot be accessed by any utilities requiring VTOC information until the diskette is remounted as a standard label (SL) diskette.

11.4 THE FLOPYDUP COPY FUNCTION

The FLOPYDUP Copy function copies an image of the entire diskette into one file on the disk. This copy is a system disk file and not a work file. If you specify C (Copy) on the FLOPYDUP Function Selection screen, the FLOPYDUP Input Diskette Definition screen (Figure 11-3) appears.

Because the diskette image file is copied to a system disk file rather than a work file, you can use the Copy function to back up a diskette to the disk. Subsequently, the Generate function can use the diskette image file to create a duplicate of the original diskette.

The FLOPYDUP Input Diskette Definition screen prompts you for the name, label type, and device (diskette drive) of the diskette that you want to duplicate. If you have not mounted the diskette, FLOPYDUP automatically invokes the mount routine. The diskette can be mounted in Shared or Exclusive mode.

When you mount the input diskette, FLOPYDUP prompts you to specify the file, library, and volume names of the disk file that is to contain the diskette image. When the copying process is finished, the End-of-Job screen appears showing the results of the FLOPYDUP Copy operation. You can then select one of the following functions from the End-of-Job screen:

- Restart FLOPYDUP processing (PF1)
- Terminate processing (PF16)

11.5 THE FLOPYDUP GENERATE FUNCTION

If you specify G (Generate) on the FLOPYDUP Function Selection screen, the FLOPYDUP Input File Definition screen (Figure 11-4) appears. The Generate function creates a new diskette volume by copying a diskette image file from the system disk to an output NL diskette. This function is not restricted to image files that were originally created by any one means (for example, the image file can be created by the Copy function of FLOPYDUP, COPY2200, or by a user program). The Generate function can create new diskette volumes in formats not standard on the VS.

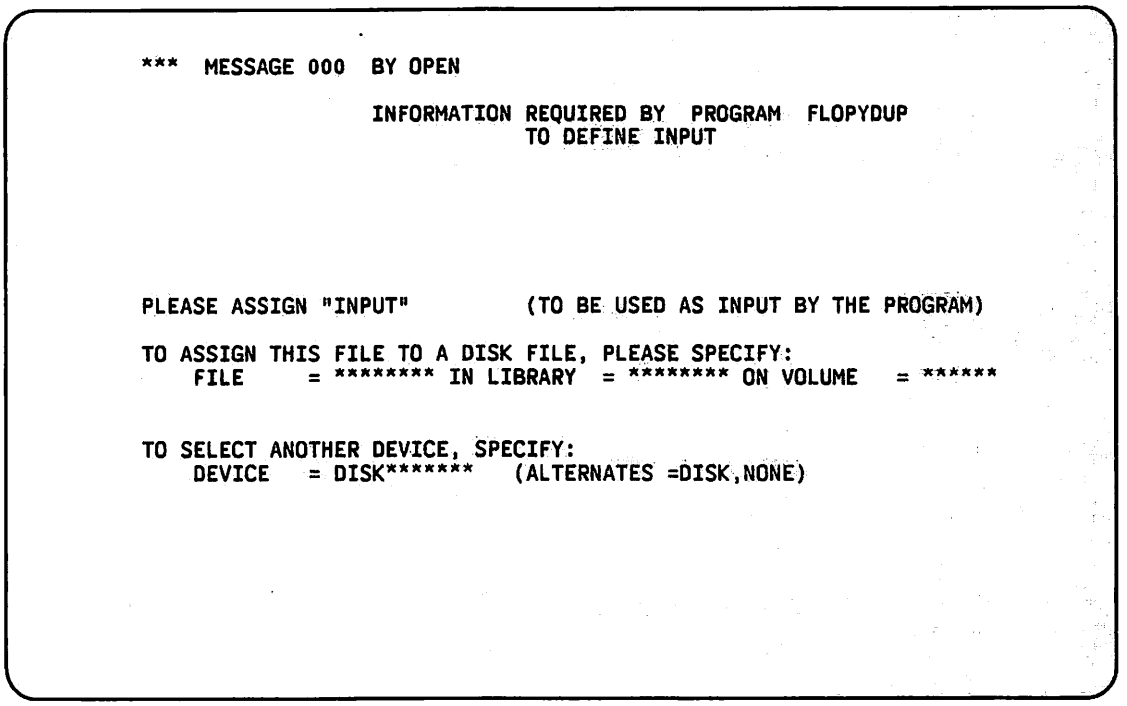


Figure 11-4. FLOPYDUP Input File Definition Screen

Specify the file, library, and volume fields to identify the disk file that contains the diskette image. Press ENTER (when the disk file has been identified) and an output definition screen appears.

Specify the name of the NL output diskette and the device number of the drive on which it is mounted. If you have not mounted the diskette, FLOPYDUP automatically invokes the mount routine.

When the diskette is generated, the End-of-Job screen appears showing the results of the FLOPYDUP Generate operation, and enables you to end or rerun FLOPYDUP processing. It also enables you to create another diskette with the same contents. The original diskette image file remains unchanged.

NOTE

When generating diskette contents from a disk image file, FLOPYDUP transfers the image file in 2-KB blocks (2048 bytes) each, regardless of the file's current data format. The utility does not decompress a compressed file before writing it to the diskette.

If the image file is created by the Copy function, the diskette image file is automatically the correct size. The file must contain an exact image of the contents of the diskette according to requirements for the output.

If you attempt to generate a diskette with a file that is too large, a message is displayed when the diskette is filled. You cannot continue the diskette generation on another volume. If you want to regenerate the large file to fit on two diskettes, split the file into parts that will fit on a diskette.

11.6 A SAMPLE FLOPYDUP PROCEDURE

You can control FLOPYDUP processing through the VS Procedure language. However, because the VS Procedure language MOUNT and DISMOUNT statements cannot be embedded in a RUN/ENTER (or DISPLAY) sequence, all Procedure language Mount or Dismount operations must be performed prior to or following automated FLOPYDUP processing. This restriction particularly affects the Duplicate function, which requires the output diskette to be mounted during the function's processing. A complete list of FLOPYDUP GETPARMs is given in Appendix A. Refer to the *VS Procedure Language Reference* for details about Procedure language syntax.

The following procedure copies the contents of a diskette to a disk image file and then generates an output diskette from the contents of the diskette image file. All mount specifications are supplied by the procedure. You need only place the diskettes in the drive when the mount message is displayed. The procedure exits and reenters FLOPYDUP each time a Mount operation is processed.

```
PROCEDURE
MOUNT DISK SLDSKT ON 014 WITH STANDARD LABEL FOR
  EXCLUSIVE USAGE
    RUN FLOPYDUP
        ENTER OPTIONS OPTION=C
        ENTER INPUT VOLSER=SLDSKT, LABEL=SL, DEVICE=14
        ENTER OUTPUT FILE=SLDSKT, LIBRARY=YOURLIB,
            VOLUME=SYSTEM
        ENTER EOJ 16
MOUNT DISK NLDSKT ON 014 WITH NO LABEL FOR EXCLUSIVE USAGE
    RUN FLOPYDUP
        ENTER OPTIONS OPTION=G
        ENTER INPUT FILE=SLDSKT, LIBRARY=YOURLIB,
            VOLUME=SYSTEM
        ENTER OUTPUT VOLSER=NLDSKT, DEVICE=014
        ENTER EOJ 16
DISMOUNT NLDSKT
RETURN
```

CHAPTER 12
THE FONTCNTL UTILITY

12.1 INTRODUCTION

The Font Control (FONTCNTL) utility enables you to manage printer fonts for text files and graphics files printing on font-loadable printers. If you have sufficient access privileges, you can install, delete, modify, make inquiries, and assign numbers to Wang-supplied fonts with the FONTCNTL utility.

12.1.1 Fonts and Font Files

A font is a complete set of type of one size and one face, provided in machine-readable form. Font files are loaded onto the system by the system administrator. Font files consists of the following parts:

- The Header Section — contains information about the font.
- The Translation Table — passes translated character information to the printer.
- The Font Data — contains the actual font characters that are printed.

Font characters are represented by hexadecimal values in the translation tables. When you install a font onto the system, there is usually a one-to-one correspondence between the hexadecimal value for a character and the actual character that is printed.

NOTE
Any user can access and print information about the fonts that are available on the system. However, for the protection of the system fonts and the users who use them, only users who have a write privilege that is equal to the protection class of the catalog file (@FONTCAT), can install, modify, or delete fonts.

This manual only details the Review Fonts and Print Catalog functions. For information about installing, modifying or deleting fonts, refer to the VS System Administrator's Reference.

12.1.2 FONTCNTL Options

The following options are supported by FONTCNTL and do not require special privileges:

- You can review the fonts that are available on your system. The font review displays the fonts for a particular printer, the file name of a particular font, and the file name of the corresponding translation table. For information about reviewing fonts, refer to Section 12.3.
- You can obtain a hardcopy of the font catalog. For more information about printing the font catalog, refer to Section 12.4.
- You can control FONTCNTL processing through the VS Procedure language. For a sample FONTCNTL procedure, refer to Section 12.5. Refer to the *VS Procedure Language Reference* for details about the VS Procedure language.

The following options are supported by FONTCNTL but require users to have a write privilege that is equal to the protection class of the catalog file. Refer to the *VS System Administrator's Reference* for information about the following options:

- You can install a font into the font catalog of any volume and assign numbers to specific fonts. The FONTCNTL utility supports international character sets.
- You can delete (remove) a font entry from the font catalog. (This function does not delete the font file.)
- You can also change font numbers.
- You can specify a default font, which is a particular font that is to be automatically used whenever you want a printout, unless another font is specified. You can also change (modify) the default font.
- You can change (modify) existing fonts through the translation table.

Figure 12-1 provides an overview of FONTCNTL processing.

12.1.3 FONTCNTL Screen Flow

When you run the FONTCNTL utility, you can traverse backwards and forwards through the FONTCNTL screens. To move forward, specify values in the fields on the screen (where applicable) and press ENTER (Continue). If a PF key option menu is displayed, press the PF key that corresponds to the action that you want to take. To view screens that have already been displayed, press PF1 (Return) to return to the previous screen (or the beginning of a FONTCNTL function). When displayed data cannot fit on a single screen, the data is listed on subsequent screens, which are called "extended screens", and are treated as extensions of the first screen. Traversing a series of screens in an extended screen is done by pressing PF4 (Previous) to view previous data, or PF5 (Next) to view subsequent data. When you press ENTER, the entire series of screens is processed.

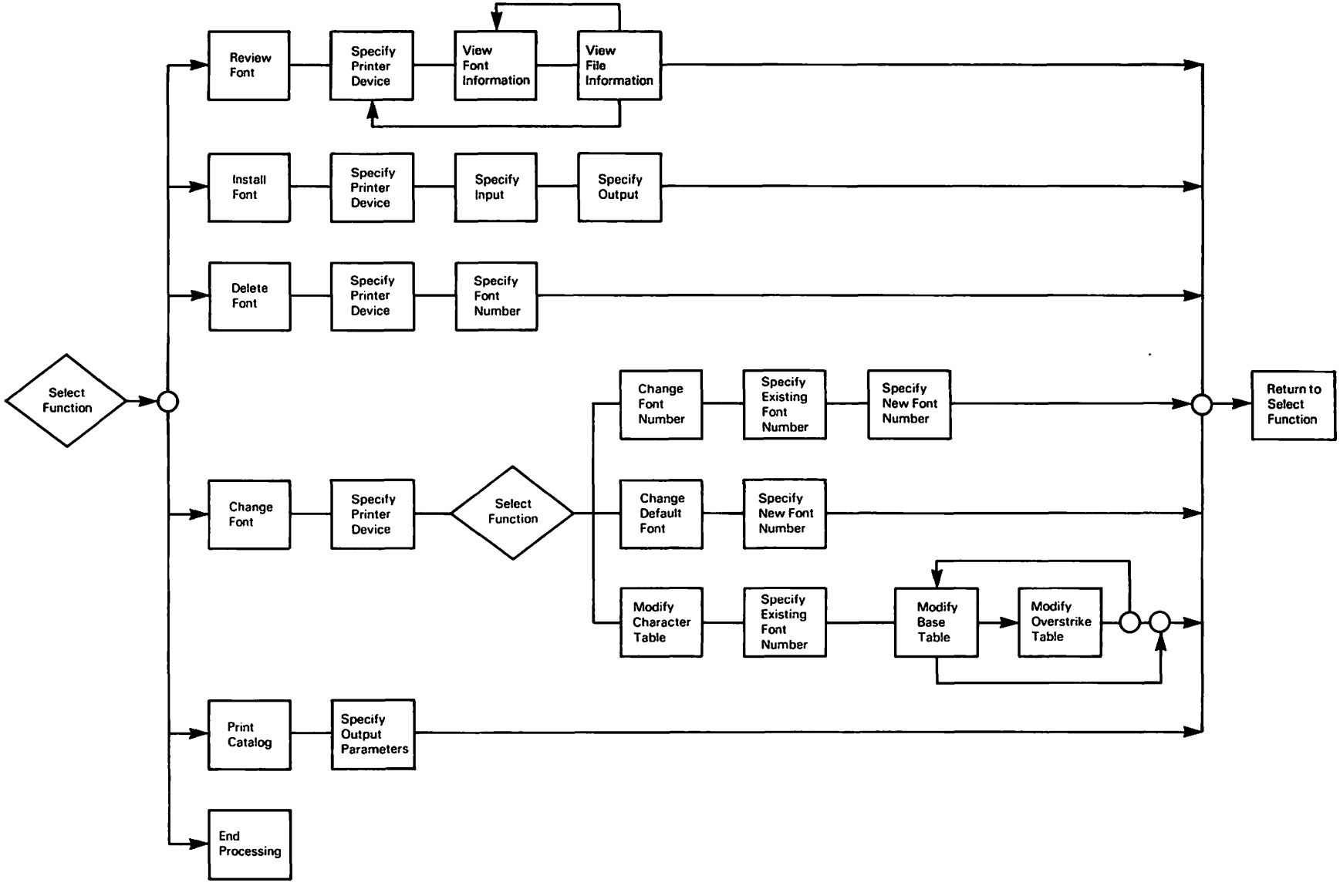


Figure 12-1. FONTCNTL Processing

12.2 RUNNING FONTCNTL

To begin FONTCNTL processing, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify FONTCNTL in the Program field (and the library and volume if FONTCNTL does not reside in @SYSTEM@ on the system volume) and press ENTER. When FONTCNTL processing begins, a status screen appears while the system initializes the utility. When the utility is initialized, the FONTCNTL Main Menu screen (Figure 12-2) appears.

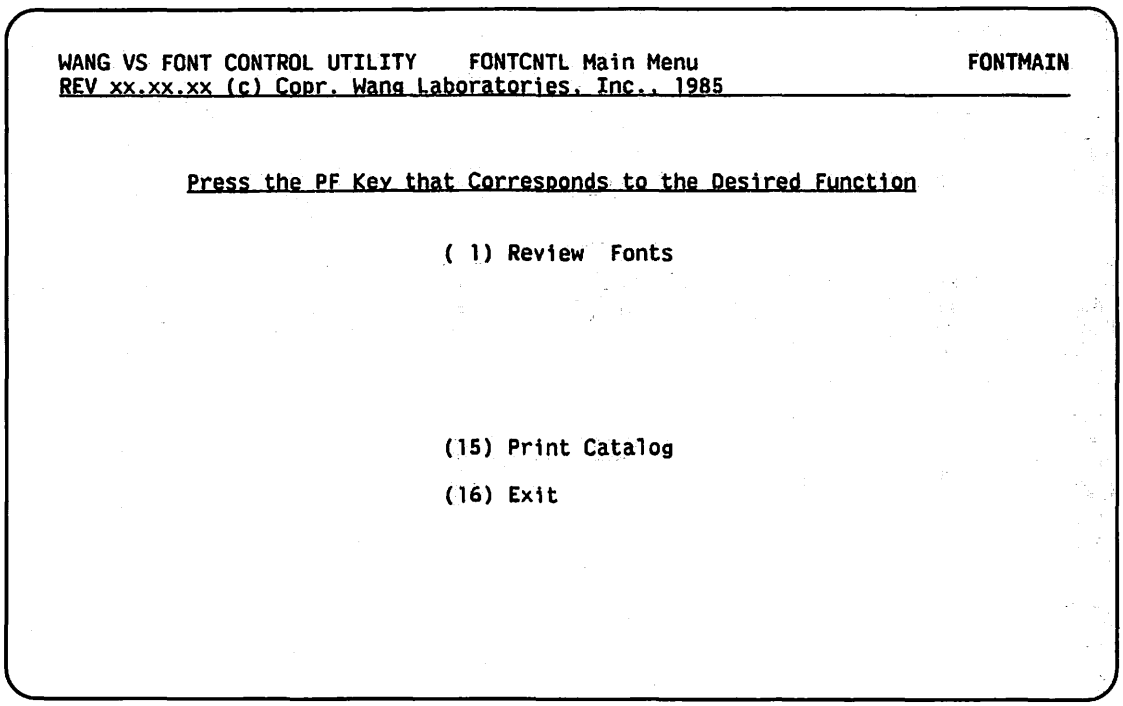


Figure 12-2. FONTCNTL Main Menu Screen

The functions on the FONTCNTL Main Menu screen are described as follows:

PF Key	Function	Description
1	Review Fonts	The Review Fonts function enables you to review the fonts that are available on the system.
15	Print Catalog	The Print Catalog function enables you to obtain a listing of the font catalog.
16	Exit	Press PF16 to terminate the current session of FONTCNTL processing.

NOTE

If you have security administrator's privileges or a write privilege that is equal to the protection class that is assigned to the font catalog, PF keys 2, 3, and 4 appear on your workstation screen enabling you to install, delete, or modify a font. For more information, refer to the VS System Administrator's Reference.

12.2.1 The FONTCNTL Device Selection Screen

When you select any of the functions on the FONTCNTL Main Menu screen, the FONTCNTL Device Selection screen (Figure 12-3) appears. The FONTCNTL Device Selection screen enables you to select the printer device on which you want to perform the function that you selected. You can select the device either by positioning the cursor in the space provided next to the device that you want and pressing ENTER, or by specifying the device number in the Device field and pressing ENTER. The Device field takes precedence over cursor position if both selection options are used.

```
FONTCNTL                        Device Selection                    DEVSCRN
(c) Copr. Wang

        Position Cursor Next to the Device Type and Press (ENTER)

            Device Type          Description

             * = 5577            Hi-Dens Matrx Ptr
             * = LIS12           12ppm Laser System
             * = LPS8            8ppm Laser System

                                  or
                        Enter a Device Type name

                          DEVICE    = LIS12*

(1) Return

                                                    (16) Main Screen
```

Figure 12-3. Sample FONTCNTL Device Selection Screen

12.3 REVIEW FONTS

When you press PF1 from the FONTCNTL Main Menu screen, the FONTCNTL Review Fonts screen (Figure 12-4) appears.

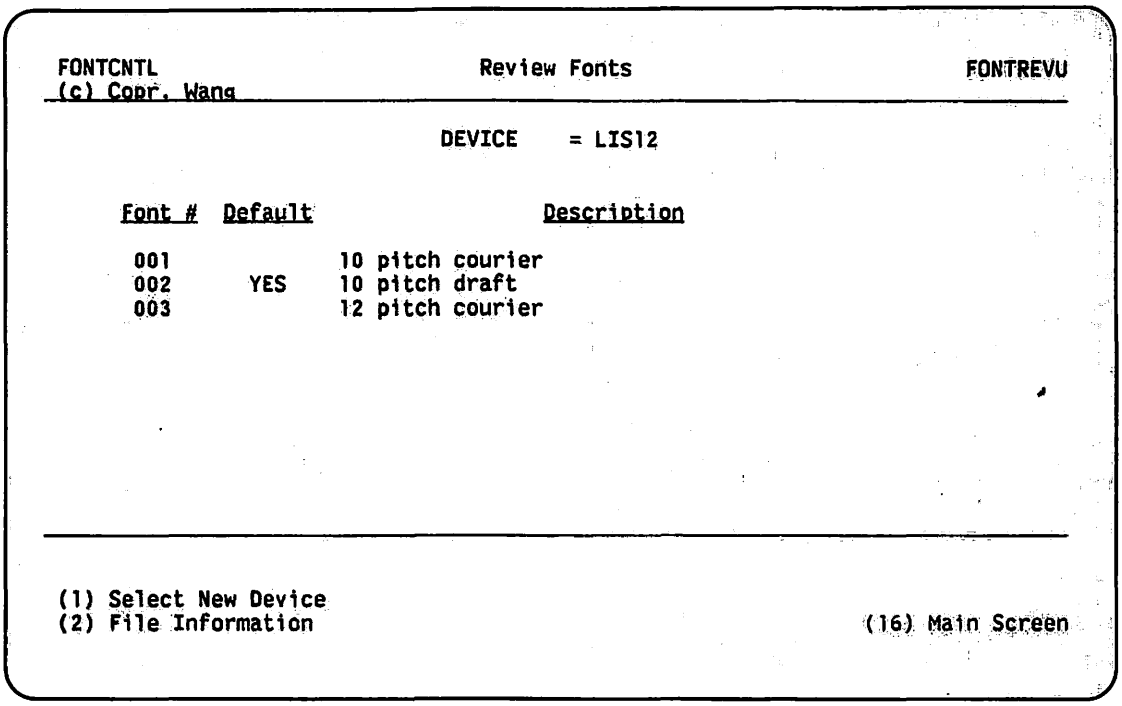


Figure 12-4. FONTCNTL Review Fonts Screen

The FONTCNTL Review Fonts screen lists the available fonts that are currently in the System Font Catalog for the device that you specified for review. The fields on the FONTCNTL Review Fonts screen are described as follows:

Field	Description
DEVICE	The Device field displays the device that you specified for review.
Font #	The Font# (Font number) field displays the number that is assigned to a particular font.
Default	The Default field is blank for all fonts except one. The one font that contains the word YES in this field is the default font for the device that you selected for viewing.
Description	The Description field displays a brief summary of the Wang-supplied or user-supplied description (i.e., pitch, character set) of a particular font.

If all of the fonts for the device that you specified for review cannot be displayed on one screen, you can view the subsequent screens of the extended screen by pressing PF5 (Next). You can view the previous screens of the extended screens by pressing PF4 (Previous).

To view the file information about the fonts that are listed on the FONTCNTL Review Fonts screen, press PF2 (File Information). When you press PF2, the FONTCNTL File Information screen (Figure 12-5) appears showing those fonts that were displayed on the FONTCNTL Review Fonts screen.

```
FONTCNTL                          Review Fonts                          FONTRVWX
(c) Copr. Wang

                                DEVICE    = LIS12

 F#    FSystem  FFile     FLib      FVol        XSystem  XTable    XLib     XVol

001             COURIER  @31310PN SYSTEM                COURIER  @31310P@ SYSTEM
002             DRAFT    @31310PN SYSTEM                DRAFT    @31310P@ SYSTEM
003             COURIER  @31312PN SYSTEM                COURIER  @31312P@ SYSTEM
004             ARTISAN  @31310PN SYSTEM                ARTISAN  @31310PN SYSTEM
005             DELEGATE @31310PN SYSTEM                DELEGATE @31310PN SYSTEM
006             DRSYMBOL @31310PN SYSTEM                DRSYMBOL @31310PN SYSTEM
007             DULGOTHC @31310PN SYSTEM                DULGOTHC @31310PN SYSTEM
008             NARRATOR @31310PN SYSTEM                NARRATOR @31310PN SYSTEM

(1) Select New Device    ( 4) Previous
(2) Description Menu     ( 5) Next                          (16) Main Screen
```

Figure 12-5. Sample FONTCNTL File Information Screen

The FONTCNTL File Information screen displays more information about each font that is assigned to the device that you specified for review. The fields on the FONTCNTL File Information screen are described as follows:

Field	Description
DEVICE	The Device field displays the device that you specified for review.
F#	The F# (Font number) field displays the number that is assigned to a particular font.
FSystem	The FSystem (Font system) field displays the name of the system on which a particular font file resides. If this field is blank, then the system on which the font file resides is the current system.
FFile	The FFile (Font file) field displays the name of the file in which a defined font resides.
FLib	The FLib (Font library) field displays the name of the library in which the corresponding font file resides.
FVol	The FVol (Font Volume) field displays the name of the volume on which the corresponding font library resides.

Field	Description
XSystem	The XSystem (translation system) field displays the name of the system on which a particular font translation file resides. If this field is blank, then the system on which the font translation file resides is the current system.
XTable	The XTable (translation table) field displays the name of the translation table file in which a specific font is defined.
XLib	The XLib (translation library) field displays the name of the library in which the corresponding translation table file resides.
XVol	The XVol (translation volume) field displays the name of the volume on which the corresponding translation library resides.

If all of the fonts for the device that you specified for review cannot be displayed on one screen, you can view the subsequent screens of the extended screen by pressing PF5 (Next). You can view the previous screens of the extended screens by pressing PF4 (Prev.) To review another device, press PF1. To return to the review fonts list, press PF2. If you want to return to the FONTCNTL Main Menu screen, press PF16.

12.4 PRINT CATALOG

The FONTCNTL Print Catalog screen (Figure 12-6) appears when you press PF15 from the FONTCNTL Main Menu screen. Although the primary function of the FONTCNTL Print Catalog screen is to print out a copy of the catalog, you can also assign it to a disk or tape file, or have it displayed on your workstation screen.

```
***  MESSAGE      BY WLROPE

                  INFORMATION REQUIRED BY PROCEDURE FONTRUN
                            TO DEFINE FONTLIST
                       ACTIVE PROGRAM IS FONTCNTL


Title : FONTLIST                                    Disposition : new
Mode : output                                       Access : sequential
I/O Format : stream                                 Organization : consecutive

To assign this file to a disk file   please specify:
     FILE     = FONTLIST  in  LIBRARY   = #MMPRT**  on   VOLUME   = SYSTEM
     RECORDS  = *****500     FILECLAS  = A

     DEVICE   = DISK**** (DISK,TAPE,PRINTER,WS)
     Tape Option :    FILESEQ  = ***1
```

Figure 12-6. Sample FONTCNTL Print Catalog Screen

The fields on the FONTCNTL Print Catalog screen are described as follows:

Field	Description
FILE	Specify the name of the file in which the font catalog is to be copied. A default name, FONTLIST, is provided.
LIBRARY	Specify the name of the library in which the font catalog copy is to reside. A default value is obtained from your workstation defaults, which are set through the SET Usage Constants function (PF2) of the Command Processor menu.
VOLUME	Specify the name of the volume on which the font catalog copy is to reside. A default value is obtained from your workstation defaults which are set through the SET Usage Constants function (PF2) of the Command Processor menu.
RECORDS	Specify the number of records to be allocated for the file. A default value is obtained through the sytem and more record space is allocated if necessary.
FILECLAS	Specify the print file class of the file. A default value is obtained from your workstation defaults which are set through the SET Usage Constants function (PF2) of the Command Processor menu.
DEVICE	Specify one of the following devices: DISK — When you select the DISK option, the print mode default is checked in your usage constants to determine which action (spool, hold, on-line, or keep) to take on the font catalog file. TAPE — When you select the TAPE option, the font catalog file is placed onto the mounted tape that you specify. PRINTER — When you select the PRINTER option, the font catalog file is spooled to the printer regardless of what your print mode default is in the usage constants. WS — When you select the WS (workstation) option, the font catalog file is displayed on your workstation screen.
FILESEQ	The Fileseq (file sequence) field is valid only when you specify TAPE in the Device field. Specify the Fileseq number where you want the font catalog file to reside.

12.5 A SAMPLE FONTCNTL PROCEDURE

You can control FONTCNTL processing through the VS Procedure language. If you create a VS Procedure language procedure to run FONTCNTL you must specify the device name in the Device field on the FONTCNTL Device Selection screen (Figure 12-3).

Appendix A provides a complete list of FONTCNTL GETPARMs. For detailed information about the syntax of the VS Procedure language, refer to the *VS Procedure Language Reference*.

Although FONTCNTL can be run through a procedure, the FONTCNTL utility is predominantly an interactive utility and procedures are only recommended for the installation of fonts onto the system volume. *The VS System Administrator's Reference* has a sample procedure that installs a font for the LIS12 printer as Font 99 and displays the first screen of the review function. The following sample procedure displays a review screen for the LIS12 printer.

```
PROCEDURE
    RUN FONTCNTL
        ENTER FONTMAIN 1
        ENTER DEVSCRN DEVICE = LIS12
RETURN
```

CHAPTER 13
THE FORMCNTL UTILITY

13.1 INTRODUCTION

The Forms Control (FORMCNTL) utility enables you to create and maintain a file of electronic vertical form control definitions for VS serial printers. A vertical form control definition controls the format of a type of computer printing paper.

Early printers used a paper tape, prepared on a teletype machine, for printer form control. Each time the paper advanced one line, the tape also advanced one character. Punches in various locations indicated the Top-of-Form, Bottom-of-Form, and various vertical tab stops. In most installations, each unique form required its own format tape. Changing forms on the printer meant changing format tapes as well.

Current Wang Printers have replaced the mechanical paper tape reader with an electronic forms control unit. You specify the form in which a print job is to be printed and the central processor (CP) loads a form definition into the printer electronically. The FORMCNTL utility is used to create or modify these electronic form definitions. After the operating system loads a form definition, it prompts you to physically change the paper form, if necessary. A form control definition consists of the following information:

- Identification number
- Form length
- Horizontal character spacing (pitch)
- Vertical line spacing
- Top-of-form (channel assignment)
- Vertical line skip (channel assignment)
- Font selection (where applicable)

13.2 FORMCNTL PROCESSING

Figure 13-1 shows an overview of producing a form definition through FORMCNTL.

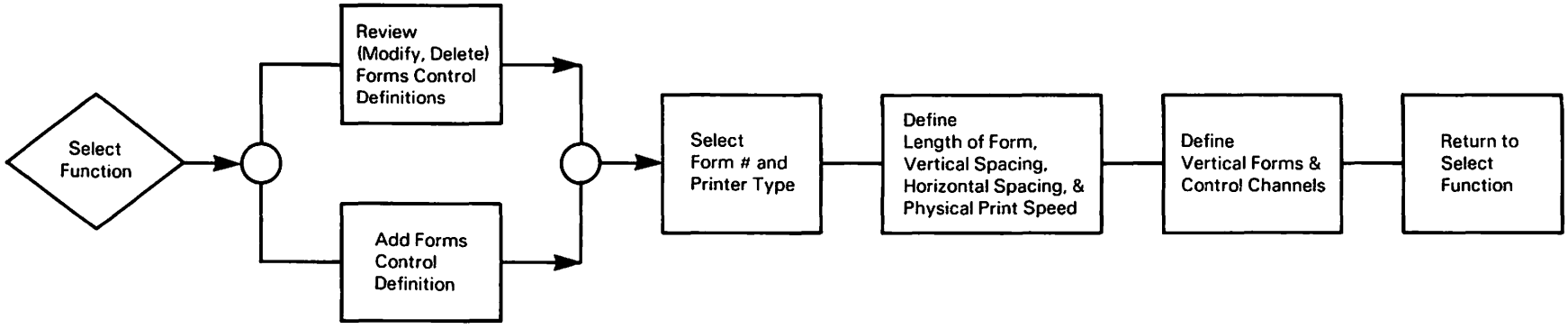


Figure 13-1. FORMCNTL Processing

To begin FORMCNTL processing, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify FORMCNTL in the Program field and press ENTER. The first screen to appear is the FORMCNTL Function Selection screen (Figure 13-2).

```
***  WANG VS PRINTER FORMS CONTROL DEFINITION UTILITY - VERSION  x.xx.xx  ***


          PRESS the PFKEY Corresponding to the Desired Function:

 (1)  REVIEW (Optional MODIFY or DELETE) Printer Forms Control Definition
      Starting with FORM # *** PRINTER TYPE ********
      (If Both Blank, the First Definition is Assumed).

 (2)  ADD a New Printer Forms Control Definition to The System.

(13)  INSTRUCTIONS.

(15)  PRINT Listing of Printer Forms Control Definitions.


(16)  END PROCESSING.
```

Figure 13-2. FORMCNTL Function Selection Screen

Select one of the following functions from the FORMCNTL Function Selection screen (Figure 13-2) by pressing the appropriate program function (PF) key:

PF Key	Function
1	Review printer forms control definition
2	Add a new printer forms control definition to the system
13	Review instructions during processing
15	Print a listing of printer forms control definitions
16	Terminate processing

NOTE
You cannot run the FORMCNTL utility through a procedure because it does not use GETPARM requests.

13.2.1 Review Printer Forms Control Definition (PF1)

If you press PF1 from the FORMCNTL Function Selection screen, the Forms Control Definition screen (Figure 13-3) appears. From this screen, you can review or modify the first forms control definition record. Forms control definition records reside in file FORMDFFN in library @SYSTEM@ on the system volume. FORMDFFN is created by the FORMCNTL utility. A forms control definition record is identified by both a form number and a printer type code.

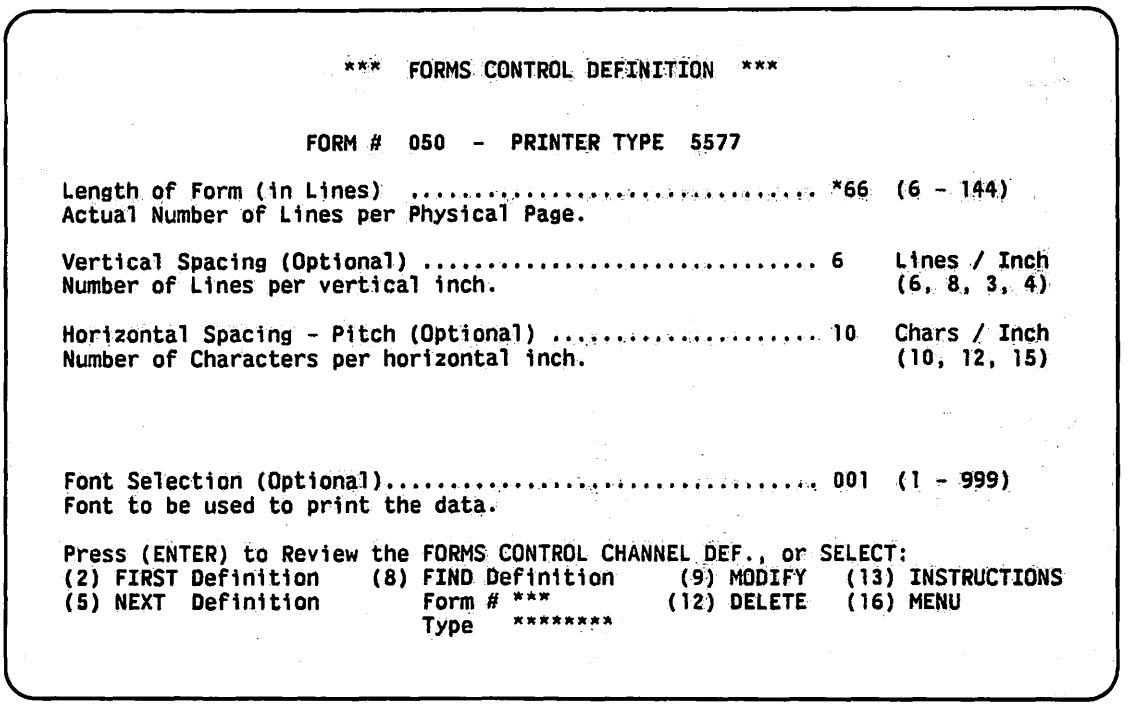


Figure 13-3. Sample Forms Control Definition Screen

You can review (or modify) a specific forms control definition record by specifying form number and printer type on the FORMCNTL Function Selection screen and then pressing PF1. The form number must be in the range of 0 to 254, and the printer type must be a valid model name. If you do not specify form number and printer type, the first record is displayed. If there are no records in the file, the utility indicates that the Forms Definition file (FORMDFFN) is currently empty (as it is when the system is first initialized; once FORMDFFN has records in the file, it retains those records between system initializations.)

When the Forms Control Definition screen appears, you can perform the following functions (on the currently displayed record) by pressing the corresponding PF key:

PF Key	Function	Description
ENTER	Review the Forms Control Channel Definition	If you press ENTER without first modifying a record, only a partial Vertical Forms Control Channel Definition screen appears which corresponds to the specific form number and printer type that was specified on the FORMCNTL Function Selection screen. If you press PF9 from the partial screen, you can modify the channel definition of the forms control definition that is currently displayed.
2	First Definition	Displays the first forms definition record from any record in the file.
5	Next Definition	Displays the record immediately following the record that is currently displayed.
8	Find Definition	Prompts you to specify the form number and the printer type fields of the record that you want displayed. After you specify the field information, pressing PF8 displays that record.
9	Modify	Highlights the forms definition record's values so that you can change them.

PF Key	Function	Description
12	Delete	Removes the current record from the file. When you press PF12, you are prompted to press ENTER to delete the file. This is a precaution against accidental deletion.
13	Instructions	Displays a description of the record attributes.
16	Menu	Returns you to the FORMCNTL Function Selection screen.

Modifying the Forms Definition Record

To modify the length of the form (in lines), the vertical spacing (optional), the horizontal spacing (optional), and the font selection, press PF9 from the Forms Control Definition screen. Next, type a new value over the current value that is displayed on the Forms Control Definition screen. A description of the Forms Control Definition screen fields follows. Valid values for these fields are dependent on the type of printer that you want to use; refer to Table 13-1 for the correct values for VS Serial printers.

Field	Description
Length Of Form	Specify the number of lines that can be printed on each physical page.
Vertical Spacing (Optional)	Specify the number of lines to be printed on each vertical inch of the form.
Horizontal Spacing (Optional)	Specify the number of characters to be printed on each horizontal inch of the form.
Font Selection (Optional)	Specify the number of the font that you want the printer to use. This determines the character set that is used when the form is printed. The default value is 01.

After you specify the field values, press ENTER to process the Forms Control Definition screen. The Vertical Forms Control Channel Definition screen is then displayed.

Table 13-1 lists valid VS serial printers that can be accessed by the FORMCNTL utility. Refer to Table 13-1 for specific printer attributes when using FORMCNTL.

Table 13-1. VS Serial Printers

Printer Model	Printer Description	Supported Length of Form	Vertical Spacing (LPI)	Horizontal Spacing (CPI)	Font Selection
Model 2233	Remote Matrix Printer, 100 CPS	6–144	6, 8	10, 12	—
Model 2235	Remote Matrix Printer, 180 CPS	6–144	6, 8	10, 12	—
Model 2233K	Remote Katakana Matrix Printer, 100 CPS	6–144	6, 8	10, 12	—
Model 2235K	Remote Katakana Matrix Printer, 180 CPS	6–144	6, 8	10, 12	—
Model 2273V1	Remote Band Printer, 250 LPM	6–126	6, 8	—	—
Model 5533	Matrix Printer, 100 CPS	6–144	6, 8	10, 12	01 (Primary) or 02 (Secondary)
Model 5535	Matrix Printer, 180 CPS	6–144	6, 8	10, 12	01, 02
Model 5573	Band Printer, 300 LPM	6–126	6, 8	—	—
Model 5574	Band Printer, 600 LPM	6–126	6, 8	—	—
Model 5575	High-Speed Band Printer, 1100 LPM	6–143	6, 8	—	—
Model 5577	High Density Matrix Printer	6–144	6, 8	10, 12	01, 02
Model 6581W	Daisywheel Printer, 40 CPS	6–144	6, 8, 3, 4	10, 12, 15	—
Model 6581WC	Wide Cartridge Daisywheel Printer 40 CPS	6–144	6, 8, 3, 4	10, 12, 15	—
Model DWOS-20	Daisywheel Printer, 20 CPS	6–144	6, 8, 3, 4	10, 12, 15	—
Model DWOS-55	Daisywheel Printer, 55 CPS	6–144	6, 8, 3, 4	10, 12, 15	—
Model DWR-20	Remote Daisywheel Printer, 20 CPS	6–144	6, 8, 3, 4	10, 12, 15	—
Model LIS12	Laser Image System, 12 PPM	6–144	6, 8, 3, 4	10, 12, 15	1–999
Model LPS8	Laser Printer System, 8 PPM	6–144	6, 8, 3, 4	10, 12, 15	1–999
Model OK555	Okidata Matrix Printer	6–144	—	—	—

Modifying a Vertical Forms Control Channel

When you press ENTER after modifying the forms definition record, the Vertical Forms Control Channel Definition screen appears (Figure 13-4). An X in the channel column indicates that the Skip to Channel command is defined for that line. The Vertical Forms Control Channel Definition screen enables you to specify the line(s) where vertical tab commands are executed. Vertical formatting determines the amount of space to be left blank between the printed lines of any type of output. A Skip to Channel command (or verticle tab) command is similar to an end-of-page indicator, except that it specifies an area within a page. Specifying a vertical format does not preclude the use of ordinary, single line feed commands, but simply helps to speed program execution and printer throughput.

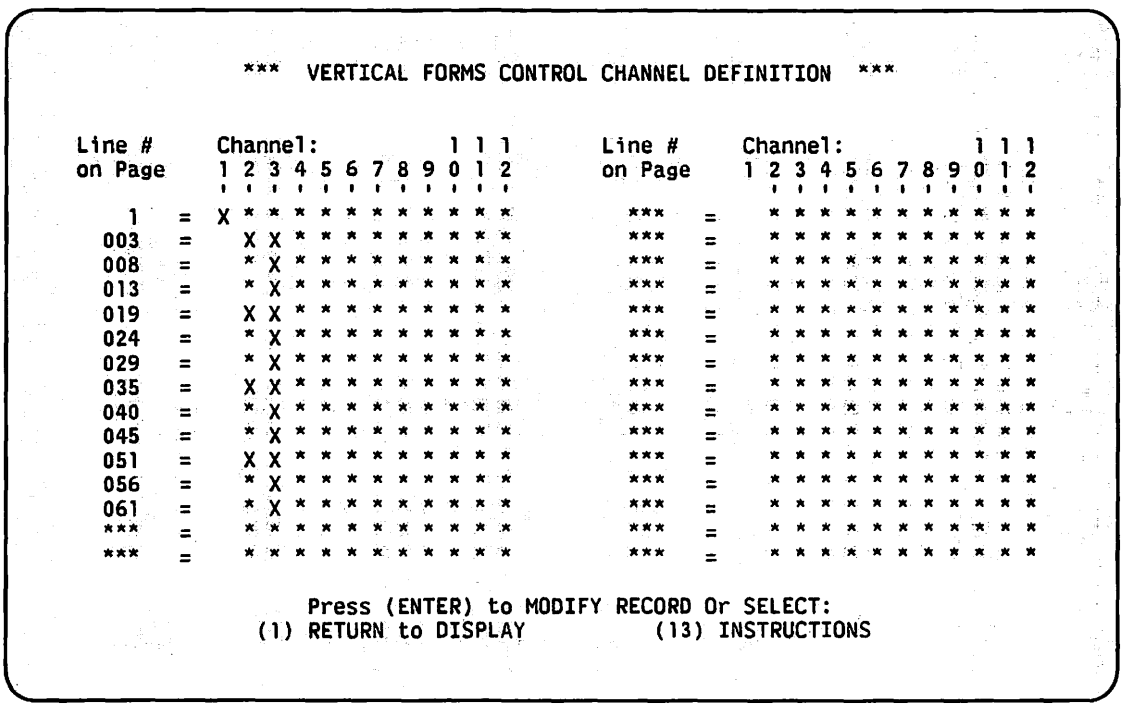


Figure 13-4. Sample Vertical Forms Control Channel Definition Screen

The following fields appear on the Vertical Forms Control Definition screen:

Field	Description
Line # On Page	Specify the lines on the page for which a channel entry is to be made. Values of Line # on Page cannot exceed the value of Length of Form specified in the Forms Control Definition screen. Line numbers need not be in ascending order nor in any particular location on the screen. Line 1 is always defined for Channel 1 (with an X) because this represents the Top-of-Form channel.
Channel	Specify the channel (or vertical tab) for each line number specified on this screen. A channel is defined by entering a nonblank character in the corresponding column. Channel 1 is always defined with an X for Line 1 to represent the Top-of-Form channel. Channel 1 cannot be defined on any other line.

NOTE
When a VS system is delivered, a definition for Form 000 (the default Forms Definition) does not exist. To establish a default Forms Definition for Form 000, you must run FORMCNTL, and the system administrator must ***add*** *a definition for Form 000 (not modify an existing one). The system default accommodates the 14 x 11-inch form size. If you do not request a special form, entries in the Print Queue are assigned Form 000. Refer to the* VS System Administrator's Reference *for more information.*

An Illustration of Using Forms Control Channels

Using the Sample Vertical Forms Control Channel Definition screen (Figure 13-4), the following example illustrates a report that is generated through the sample form.

Note the following conditions and refer to the Sample Printout Using the Forms Control Channels (Figure 13-5) as necessary:

- A Skip to Channel 2 command is issued to the printer whenever a new employee record is encountered in the report.
- A Skip to Channel 3 command is issued to the printer when the "Current Projects" section of the employee record is encountered or if the number of projects that have been printed equals 4 and there are more projects to be printed.

1. Cora Clemens is the first employee record to be printed. A Channel 1 command is issued to position the printer at the top of the first page. A Skip to Channel 2 command is issued to begin printing the first record at the first occurrence of a Channel 2 indicator (as defined on the Sample Vertical Forms Control Channel Definition screen; Figure 13-4). Cora Clemens' name, employee number and address are printed beginning at Line 3, after which, a Skip to Channel 3 command is issued to the printer.

   From the current position on the form, the next Channel 3 indicator (as defined) occurs at Line 8 so the printer begins printing Cora Clemens' current projects at Line 8. After 4 projects are printed, a second Skip to Channel 3 command is issued to the printer. The next Channel 3 indicator (as defined) occurs at Line 13. Cora Clemens' remaining 2 projects are printed on Lines 13 and 14. Cora Clemens' record printing is completed so a Skip to Channel 2 command is issued tothe printer for the next record to be printed.
2. The next Channel 2 indicator (as defined) occurs at Line 19. The printer advances to Line 19 and prints Annie Farnaby's name, employee number and address. A Skip to Channel 3 command is then issued to the printer to list Annie's projects. Annie has 10 projects; after each set of four projects a Skip to Channel 3 command is issued. The two sets of four projects are printed begining at Lines 24 and 29, respectively. A third Skip to Channel 3 command places the printer on Line 35 and the printer prints the remaining two projects. When Annie's record printing has completed, a Skip to Channel 2 command is issued to the printer.
3. The next Channel 2 (as defined) occurs at Line 51 (not 35, because the printer has already passed Line 35). The printer advances to Line 51 and prints Helen Johnson's name, employee number and address. A Skip to Channel 3 command is then issued to the printer to list Helen's three projects (beginning at Line 56). A Skip to Channel 2 command is issued to the printer for the next record to be printed.
4. The next Channel 2 (as defined) occurs at Line 3 and so a Top-of-Page command (Skip to Channel 1) is issued. The printer advances to the top of the next page and resumes printing operations on Line 3.

```
 1
 2
 3      Cora Clemens                                Employee #  51369
 4      123 Main Street                             5/12/85
 5      Lansing, MI  21854                          Project Status Report
 6
 7
 8      Project  1          Completed 5/24/85
 9      Project  2          In Progress
10      Project  3          In Progress
11      Project  4          On hold indefinitely
12
13      Project  5          Postponed until 7/85
14      Project  6          In Progress
15
16
17
18
19      Annie Farnaby                               Employee #  45336
20      90 Jefferson Blvd.                          5/12/85
21      Edison, NJ  08817                           Project Status Report
22
23
24      Project  1          In Progress
25      Project  2          In Progress
26      Project  3          Completed  5/12/85
27      Project  4          Completed  4/07/85
28
29      Project  5          Completed  3/24/85
30      Project  6          In Progress
31      Project  7          Reassigned
32      Project  8          In Progress
33
34
35      Project  9          In Progress
36      Project 10          Cancelled
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51      Helen Johnson                               Employee #  42330
52      2434 23rd Street                            5/12/85
53      Cuyahoga Falls, OH 44223                    Project Status Report
54
55
56      Project 1           Completed 5/17/85
57      Project 2           In progress
58      Project 3           In Progress
59
60
61
62
63
64
65
66
```

Figure 13-5. Sample Printout Using the Forms Control Channels

13.2.2 Add a New Forms Control Definition to the System (PF2)

If you press PF2 from the FORMCNTL Function Selection screen (Figure 13-2), the Add Forms Control Definition screen (Figure 13-6) appears. This function enables you to add new forms control definition information to the forms definition file (FORMDFFN). The Add Forms Control Definition screen prompts you for the Form # and Printer Type values. It also displays a list of printer types.

```
                    ***  ADD FORMS CONTROL DEFINITION  ***

FORM # ........ ***        - Supply the Installation Assigned Form Number
                             (000 - 254) to which this Forms Control Definition
                             is to apply.

PRINTER TYPE .. ********   - Supply the Printer Type to which use of this
                             Forms Control Definition is to be restricted.
                             Select from the following:

                    2273V1     -   250 lpm Rem Band Ptr
                    2233       -   100 cps Rem Mtx Ptr
                    2235       -   180 cps Rem Mtx Ptr
                    2233K      -   100 cps Katakana Ptr
                    2235K      -   180 cps Katakana Ptr
                    DWR20      -   20 cps Rem Daisy Ptr
                    DWOS20     -   20 cps Daisy Printer
                    6581W      -   35 cps Daisy Printer
                    6581WC     -   35 cps Wide Daisy
                    DWOS55     -   55 cps Daisy Printer
                    5573       -   250 lpm Band Printer

                      Press PF5 to Display MORE Printer Types
              Press (ENTER) to Add the Designated Forms Control Definition
                           Or Press PF1 to RETURN to MENU
```

Figure 13-6. Add Forms Control Definition Screen

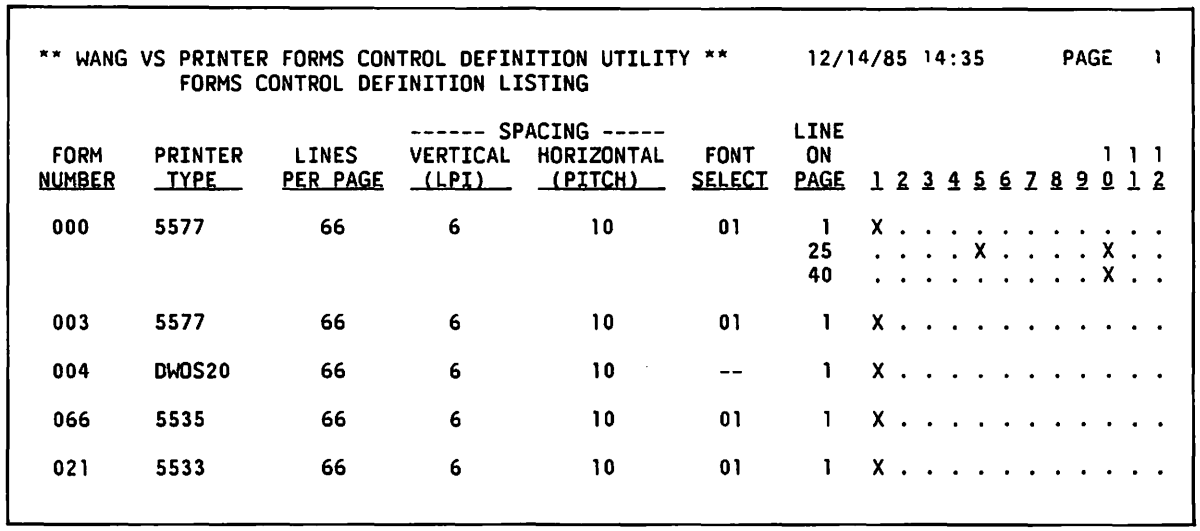
After you have specified the field values, press ENTER to add the new record to the file. The Forms Control Definition screen (Figure 13-3) then appears; you can review and modify it as necessary (as described in Section 13.2.1). After you have inspected (or modified) the new record, press ENTER and the Vertical Forms Control Channel Definition screen (Figure 13-4) is displayed for you to specify the line(s) where vertical tab commands are executed (refer to Section 13.2.1). When the new record is added, the FORMCNTL Function Selection screen (Figure 13-2) reappears. The values for Form # and Printer Type of the newly created record are displayed as the default values in the Review Printer Forms Control Definition field.

13.2.3 Review Instructions During Processing (PF13)

You can review instructions concerning the FORMCNTL Function Selection screen (Figure 13-2) or the Forms Control Definition screen (Figure 13-3) from your workstation by pressing PF13 from any screen. The instruction screens provide the information that you need to process those screens.

13.2.4 Print Listing of Printer Forms Control Definition (PF15)

You can produce a listing of all the printer forms control definition records by pressing PF15 from the FORMCNTL Function Selection screen (Figure 13-2). Figure 13-7 shows a sample Forms Control Definition Listing.



** WANG VS PRINTER FORMS CONTROL DEFINITION UTILITY ** 12/14/85 14:35 PAGE 1
FORMS CONTROL DEFINITION LISTING

FORM NUMBER	PRINTER TYPE	LINES PER PAGE	SPACING VERTICAL (LPI)	SPACING HORIZONTAL (PITCH)	FONT SELECT	LINE ON PAGE	1 2 3 4 5 6 7 8 9 10 11 12
000	5577	66	6	10	01	1	X . . . .
						25	. . . . X . . . . X . .
						40	. . . . X . .
003	5577	66	6	10	01	1	X . . . .
004	DWOS20	66	6	10	--	1	X . . . .
066	5535	66	6	10	01	1	X . . . .
021	5533	66	6	10	01	1	X . . . .

Figure 13-7. Sample Forms Control Definition Listing

CHAPTER 14
THE IBMCOPY UTILITY

14.1 INTRODUCTION

The IBMCOPY utility can transfer information between a Wang VS and any system using IBM-format soft-sectored diskettes. The IBMCOPY utility copies a single file, selected files, or a complete volume from IBM Data Exchange format diskette(s) to VS disk storage, or from VS disk storage to IBM Data Exchange format diskette(s). IBMCOPY converts copied files to and from the IBM and VS file formats. Optionally, IBMCOPY converts files containing only character data to and from the American Standard Code for Information Interchange (ASCII) and Extended Binary Coded Decimal Interchange Code (EBCDIC) character sets. Only IBMCOPY reads and writes IBM Data Exchange format diskettes.

NOTE
You must use the TAPECOPY utility to perform IBM and VS tape transfers (refer to Chapter 25).

The IBMCOPY utility supports the following functions:

- Mount an IBM Data Exchange format diskette
- Initialize a soft-sectored diskette to an IBM Data Exchange format
- Display or print an IBM diskette directory
- Transfer data from an IBM Data Exchange format diskette to a VS disk
- Transfer data from a VS disk to an IBM Data Exchange format diskette
- Control IBMCOPY workstation interaction through VS Procedure language

14.1.1 IBM Data Exchange Format Diskettes

IBM Data Exchange format diskettes are structured differently than VS diskettes. IBMCOPY accepts the following types of 8-inch, soft-sectored IBM diskettes that are available in the following forms:

- Single-sided, single-density (SSSD) diskettes in the IBM Basic Data Exchange format (IBM Diskette 1, Wang Green Label)
- Double-sided, single-density (DSSD) diskettes in the IBM Basic Data Exchange format (IBM Diskette 2, Wang Red Label)
- Double-sided, double-density (DSDD) diskettes in the IBM Type H Data Exchange format (IBM Diskette 2D, Wang Red Label)

NOTE
You cannot use hard-sectored diskettes for input or output of IBM Data Exchange formatted data. Hard-sectored diskettes have a fixed sector size. Due to this restriction, IBMCOPY requires you to use a diskette drive that can access soft-sectored diskettes.

Cylinder 0 of all IBM diskettes is reserved as the Index Cylinder. Track 0 of the Index Cylinder is referred to as the Index Track and describes the format and contents of the diskette. The Index Track contains

- An error map indicating defective and alternate tracks and sectors.
- A volume label indicating the physical and logical structure of the diskette.
- Nineteen data set header labels. For double-sided diskettes, Track 1 of the Index Cylinder contains additional data set header labels.

Double-density diskettes have sector sizes of 128 and 256 bytes. Single-density diskettes have sector sizes of 128 bytes. Track 0 of the Index Cylinder (the Index Track) is always recorded with 128 bytes for each sector, regardless of whether the diskette is single- or double-density. Track 1 of the Index Cylinder is recorded with 128 bytes per sector for single-density diskettes, and with 256 bytes per sector for double-density diskettes.

All IBM diskette files are consecutively organized with fixed-length records. Records must be unblocked, with logical records contained entirely within a single physical sector. The file's record length can vary from 1 to 128 on single-density diskettes, and from 1 to 256 on double-density diskettes.

Each file is stored in a single, consecutive extent. To locate the file, the data set header label for the file in the Index track contains Beginning-of-Extent (BOE) and End-of-Extent (EOE) extent limits, and an End-of-Data (EOD) pointer. The End-of-Data pointer always points to the next physical sector to be used for record storage.

Empty files are denoted by a header label in which the BOE and EOD pointers point to the same physical sector. Multivolume files are allowed, with sequence numbers optionally contained in the data set header labels. The EBCDIC character set is normally used for all character storage.

The maximum number of files that can be written to an IBM Data Exchange format diskette depends on the type of diskette. While the actual number of files that can be written to any given diskette depends on the size of the files, the following list indicates the maximum number of files that can be written to each type of diskette:

Diskette Type	Maximum Number of Files
Single-sided, single-density	19
Double-sided, single-density	45
Double-sided, double-density	71

14.1.2 IBMCOPY Processing

Figure 14-1 shows an overview of IBMCOPY processing.

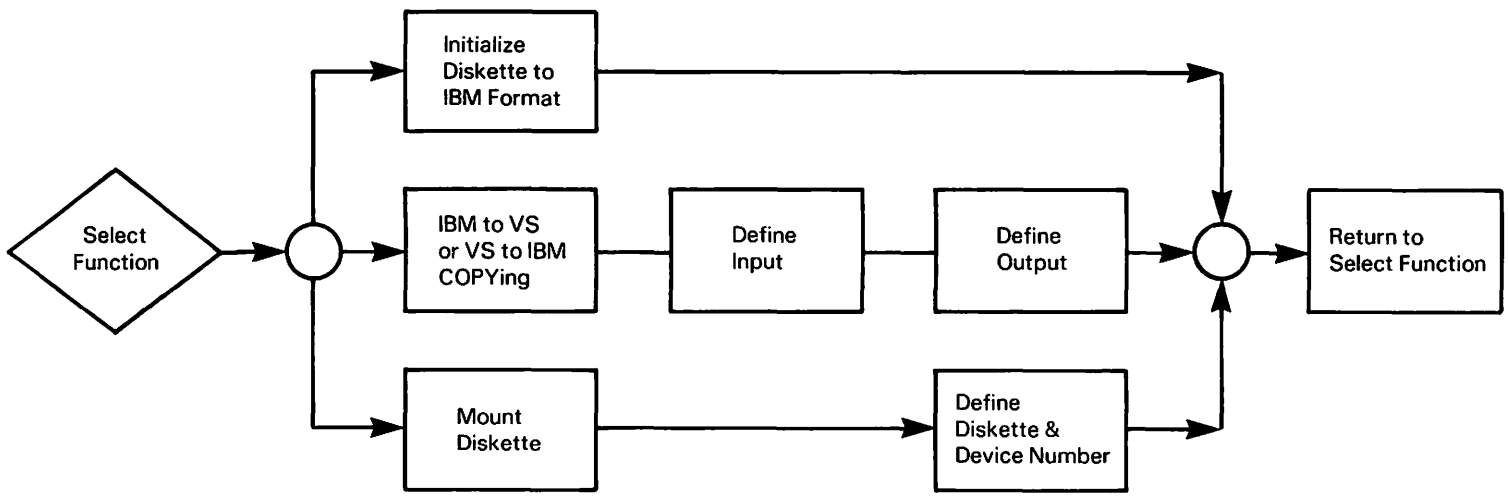


Figure 14-1. IBMCOPY Processing

When IBMCOPY processing begins, the IBMCOPY Function Selection screen (Figure 14-2) is displayed.

```
***  MESSAGE MN00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  IBMCOPY
                          TO DEFINE FUNCTION
                      ACTIVE SUBPROGRAM IS IBMCOPY

   ***  Wang VS - IBM Diskette Compatibility Utility - Version  x.xx.xx ***
                Copyright, Wang Laboratories, Inc. 1985

  IBMCOPY will allow files to be copied to and from IBM Basic or Type H
  Data Exchange Format Diskettes.  Standard soft sectored diskettes may
  also be initialized in an IBM format.

  Please press the appropriate PF key to select the desired function:

       PFKEY     FUNCTION

         (1)     Copy files from an IBM format diskette to VS disk files.
         (2)     Copy VS disk files to an IBM format diskette.
         (3)     Initialize the diskette using an IBM format.
         (4)     Mount an IBM format diskette.
         (5)     Display an IBM format diskette directory.

        (16)     End Processing.
```

Figure 14-2. IBMCOPY Function Selection Screen

From the IBMCOPY Function Selection screen, select the program function (PF) key that corresponds to the function that you want to perform.

14.2 COPY FILES FROM AN IBM FORMAT DISKETTE TO VS DISK FILES (PF1)

IBMCOPY displays the IBM to VS Options screen (Figure 14-3) when you press PF1 from the IBMCOPY Function Selection screen. You can return to the IBMCOPY Function Selection screen by pressing PF1 again.

If you select this function, IBMCOPY transfers IBM files to VS disk storage. IBMCOPY automatically converts IBM files to the VS file format and, at your request, converts the IBM character data from EBCDIC to ASCII. You can transfer a single file, selected files, or all files on the IBM diskette. You can also automatically transfer the contents of several IBM diskettes into a single VS file through separate IBMCOPY transfers.

```
***  MESSAGE IV00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  IBMCOPY
                          TO DEFINE OPTIONS
                       ACTIVE SUBPROGRAM IS IBMCOPY


                     IBM Diskette to VS Disk File Copy

Please enter the type of copy to be performed, the input volume name,
and the input file name (if appropriate):

     COPY     = FILE      Options: VOLUME - Copy all files on the volume.
                                   SELECT - Select files to be copied.
                                   FILE   - Specify one file to be copied.

     VOLUME   = IBM***    Note: The volume must be an IBM format diskette.

     FILE     = PROGRAMS  Note: May only be specified when COPY = FILE.

Or press PF1 to return to the function selection screen.
```

Figure 14-3. Sample IBM to VS Options Screen

Specify the IBM input from the IBM to VS Options screen. The IBM to VS Options screen fields are described as follows:

Field	Description
COPY	Specify VOLUME, SELECT, or FILE as the range of the copying operation. The default value of the Copy field is VOLUME. VOLUME — This value indicates that the entire contents of the IBM diskette that you select is copied. Each file on the diskette is copied to an equivalent file on the VS. Section 14.2.1 presents subsequent processing for volume copying. SELECT — This value indicates that only those files on the IBM diskette that you select are copied. Section 14.2.3 describes subsequent processing for selected file copying. FILE — This value indicates that a single file from the IBM diskette that you select is copied. Specify the name of the IBM file in the File field on this screen. Section 14.2.4 discusses subsequent processing for a single file copy.
VOLUME	Specify the name of the IBM input diskette. The Volume field defaults to the name of the volume last specified through this function of IBMCOPY. If no last name is available, the Volume field defaults to blanks. If the specified IBM diskette is not mounted, IBMCOPY prompts you to mount the diskette (refer to Section 14.6).
FILE	Specify the name of the file to be copied (only when copying a single file). Leave this field blank if you are copying more than one file during this operation.

14.2.1 Copying an IBM Diskette Volume to a VS Disk

If you specify VOLUME for the Copy field and an IBM format diskette as the input volume, the IBMCOPY Volume Output Definition screen (Figure 14-4) is displayed.

```
***  MESSAGE IV00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  MENU
                             TO DEFINE OUTPUT
                         ACTIVE SUBPROGRAM IS IBMCOPY


 Please enter the output library and translation options:

      LIBRARY  = XXXORG** on VOLUME   = SYSTEM

      Translate the files from EBCDIC to ASCII?  TRANSL   = NO  (YES or NO)

 Or Press the appropriate PF Key to select the desired option:

      PFKEY    FUNCTION

       (1)     Return to the input selection screen.
       (2)     Specify output attributes individually for each file.

      (16)     Abort this operation.
```

Figure 14-4. Sample IBMCOPY Volume Output Definition Screen

The Library and Volume fields are the destination into which you are copying the files. The default response for Library and Volume is your default output library (OUTLIB) and your default output volume (OUTVOL), which is set through the SET Usage Constants function (PF2) from the Command Processor menu.

The TRANSL field determines whether or not the files are to be translated from EBCDIC to ASCII code. The default value is YES, which performs the translation. If you rerun the IBM to VS copy sequence, the default value is the previously entered value for each field.

NOTE
IBMCOPY performs only character translation. For this reason, specify NO for the Transl field for files that contain numeric data. These files can be translated through the TRANSL utility after IBMCOPY has converted the files to the VS format (refer to Chapter 27).

If you specify the name of an output file that already exists on the volume, you can

- Rename the output file.
- Skip the current file and continue with the next file (PF2).
- Delete the existing file (PF3).
- Append the specified file to the existing file (PF5).

The file attributes that you specified must match those of the existing file, if you append the specified file to an existing VS file. You also have the following options from the IBMCOPY Volume Output Definition screen:

PF Key	Function
1	Return to the IBMCOPY Function Selection screen.
2	Specify the output attributes individually for each file.
16	Abort this operation.

14.2.2 The IBMCOPY End-of-Job Menu

When all the files on the volume have been copied, the IBMCOPY End-of-Job menu (Figure 14-5) appears. The IBMCOPY End-of-Job menu enables you to perform one of the following:

- Copy more IBM data to the VS by pressing PF1.
- Return to the IBMCOPY Function Selection screen by pressing PF2.
- Exit IBMCOPY by pressing PF16.

NOTE
You should dismount the IBM diskette through the Command Processor menu after exiting the IBMCOPY utility.

```
***  MESSAGE EJ01 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  MENU
                          TO DEFINE EOJ
                     ACTIVE SUBPROGRAM IS IBMCOPY

 IBM to VS copy completed.

 Please press the appropriate PF Key to select the desired function:

      PFKEY     FUNCTION

       (1)      Restart IBM to VS copy.
       (2)      Return to the function selection menu.

      (16)      End Processing.
```

Figure 14-5. IBMCOPY End-of-Job Menu

14.2.3 Copying Selected Files from an IBM Diskette Volume to a VS Disk

If you specify SELECT for the Copy field, specify an IBM format diskette as the input volume, and press ENTER from the IBM to VS Options screen (Figure 14-3), the IBMCOPY File Selection screen is displayed. You can return to the IBM toVS Options screen by pressing PF1.

The IBMCOPY File Selection screen lists all files on the specified IBM diskette and indicates how many files are on the diskette. If more than 20 files exist on the diskette, the file list continues on another screen. To select files to be copied, type any alphanumeric character in the field next to the desired file. Press ENTER when you have selected all the files that you want from this screen. When you press ENTER, any subsequent file selection screens are displayed.

If you complete file selection on the second screen out of six IBMCOPY File Selection screens, you can skip the subsequent screens and continue processing by pressing PF16. If all the files on the IBM diskette have been displayed, press ENTER to end file selection and continue processing.

After you select the files to be copied, the Selected IBM Files Output Definition screen is displayed. The Selected IBM Files Output Definition screen looks the same as the IBMCOPY Volume Output Definition screen (Figure 14-4).

Specify the output library and volume locations of the selected files and indicate whether the files are to be translated from EBCDIC to ASCII (refer to Section 14.2.1).

The Selected IBM Files Output Definition screen determines the output options for all the selected files. It does not allow you to change file organization, file location, or storage characteristics for individual files. If you press ENTER from the Selected IBM Files Output Definition screen, all selected files are copied with the output options that you specified on the Selected IBM Files Output Definition screen; each selected IBM file is transferred to the specified library and volume location with the IBM file name used as the VS file name.

If you press PF2 instead of ENTER from the Selected IBM Files Output Definition screen, you can specify the output options individually for each file because IBMCOPY displays an IBMCOPY Single File Output Definition screen (Figure 14-6) for each selected file. From this point, if you selected multiple files to be individually defined, you can perform multiple single file copy operations, described in the following section.

```
***  MESSAGE IV00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  MENU
                          TO DEFINE IOUTPUT
                     ACTIVE SUBPROGRAM IS IBMCOPY


Please enter the output file and options for input EXTSET on volume IBM:
     FILE     = PROGRAMS in LIBRARY  = XXXORG   on VOLUME   = SYSTEM
     TRANSL   = YES          (Translate the file from EBCDIC to ASCII ?)
     RECORDS  = 1000         (Number of records in the output file)
     RELEASE  = YES          (Release unused space in the output file ?)
     FILEORG  = C            (C - Consecutive or I - Indexed)
     COMPRESS = YES          (Compress the output file ?)
     FILECLAS = #            (File class of output file)

For indexed files, please specify:
     KEYLEN   = ***          (Primary key length)
     KEYPOS   = *****        (Primary key position from 1)
     IPACK    = 100 %        (Packing density for index blocks)
     DPACK    = 100 %        (Packing density for data blocks)

     Or press PF16 to abort this operation.
```

Figure 14-6. Sample IBMCOPY Single File Output Definition Screen

14.2.4 Copying a Single File from an IBM Diskette Volume to a VS Disk

If you specify FILE in the Copy field on the IBM to VS Options screen (Figure 14-3), and then specify the file and volume names of the IBM file, the IBMCOPY Single File Output Definition screen (Figure 14-6) is displayed.

From the IBMCOPY Single File Output Definition screen, you can change the file, library, and volume names of the output file, and you can specify the file organization and storage characteristics for the single output file. You can also select whether the input file is translated from EBCDIC to ASCII. After you specify the individual output file requirements, the file is copied. The fields on the IBMCOPY Single File Output Definition screen (Figure 14-6) are described as follows:

Field	Description
FILE	Specify the VS file name of the output file. Valid VS file names include alphanumeric values of up to eight characters. The default value is the IBM file name.
LIBRARY	Specify the VS library name in which the output file is to reside. Valid VS library names include alphanumeric values of up to eight characters.
VOLUME	Specify the VS volume name in which the output file is to reside. Valid VS volume names include alphanumeric values of up to six characters.
TRANSL	If you want the file to be translated from EBCDIC to ASCII, specify YES. If you specify NO, the file remains in the EBCDIC character set. The default value for the Transl field is YES. See the note concerning the Transl field following this field description list.

Field	Description
RECORDS	Specify the number of records to be allocated for the output file. (If you intend to copy the data from several IBM diskettes into a single VS file, you should specify the number of records required for the entire file and specify NO in the Release field.) The default value is the number of records in the IBM input file.
RELEASE	If you want to release unused, allocated space after the file is created, specify YES for this field. You should specify NO for cases where data from several IBM diskettes is to be transferred into a single VS file.
FILEORG	Specify C (consecutive) or I (indexed) file organization for the output VS file. If you specify C in the Fileorg field, the output file remains in consecutive organization. All files on an IBM Data Exchange format diskette are stored as consecutive files.
COMPRESS	If you want the VS output file to be stored using data compression to conserve storage space, specify YES for this field. Refer to the *VS Data Management System (DMS) Reference* for details on data compression. If NO is specified, data compression is not used. The default value is YES.
FILECLAS	Specify the VS file class of the output file. Valid file classes are A to Z, @, #, $, and blank. Refer to the *VS System User's Introduction* for details on file classes. The Fileclas field defaults to your default file class, which is set through the SET Usage Constants function (PF2) from the Command Processor menu or the VS Procedure language SET statement.

NOTE

If you are copying a multivolume (multiple IBM volume) file, specify the FILE option on the IBM to VS Options screen rather than the VOLUME option so that you can then specify the approximate file size on the IBMCOPY Single File Output Definition screen (Figure 14-6). Once the initial file has been created, the data is appended to it for subsequent diskettes.

When copying a multivolume file, specify the approximate number of records for the entire file, and then specify NO in the Release field so that the space is not released after the first diskette.

Data files should not be translated through the IBMCOPY utility.

If you specify I in the Fileorg field, IBMCOPY converts the IBM file to a VS indexed file. If you choose to create an indexed file, you can specify a primary key only and you **must** specify the Keylen, Keypos, Ipack, and Dpack fields (described in this list) to define the index of the file. The default value is C.

Field	Description
KEYLEN	Specify the length (in bytes) of the primary key field. The key length can be any value from 1 to the length of the record, but cannot exceed 255 bytes. Keylen defaults to blanks. You must specify a value in the Keylen field if you specify I in the Fileorg field.
KEYPOS	Specify the starting byte position of the key field. A valid key position can be any byte from Byte 1 to the last byte of the record. You must specify a value in the Keypos field if you specify I in the Fileorg field.

Field	Description
IPACK	Specify the packing density of the index blocks (stored on the output disk). The default value is 100. Refer to the *VS Data Management System (DMS) Reference* for further informationon packing densities. You must specify a value in the Ipack field if you specify I in the Fileorg field.
DPACK	Specify the packing density of the data blocks (stored on the output disk). The default value is 100. Refer to the *VS Data Management System (DMS) Reference* for further information on packing densities. You must specify a value in the Dpack field if you specify I in the Fileorg field.

NOTE
IBMCOPY performs only character translation. If the files contain numeric data, specify NO for the Transl field. Files containing numeric data can be translated through the TRANSL utility after IBMCOPY has converted the files to the VS format (refer to Chapter 27 for more information about the TRANSL utility).

After you define the output file fields, the IBM file is transferred to the VS. If you specify the name of an output file that already exists on the volume, you can

- Rename the output file.
- Skip the current file and continue with the next file (PF2).
- Delete the existing file (PF3).
- Append the specified file to the existing file (PF5).

If you append the specified file to an existing VS file, the file attributes that you specified must match those of the existing file.

When the files are copied, IBMCOPY displays the IBMCOPY End-of-Job menu. The IBMCOPY End-of-Job menu enables you to copy more IBM data to the VS by pressing PF1, return to the IBMCOPY Function Selection screen by pressing PF2, or exit IBMCOPY by pressing PF16. Dismount the IBM diskette through the Command Processor menu after exiting the IBMCOPY utility.

14.3 COPY VS DISK FILES TO AN IBM FORMAT DISKETTE (PF2)

If you select this function, IBMCOPY transfers VS disk files to an IBM diskette and displays the VS to IBM Input Definition screen (Figure 14-7). To return to the IBMCOPY Function Selection screen, press PF1.

IBMCOPY automatically converts the VS files to the IBM file format and, at your request, converts the VS files from ASCII to EBCDIC. You can transfer an entire VS library, selected files in a library, or a single file to the IBM diskette.

```
***  MESSAGE VI00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  IBMCOPY
                          TO DEFINE OPTIONS
                     ACTIVE SUBPROGRAM IS IBMCOPY


                        VS Disk to IBM Diskette File Copy

  Please enter the type of copy to be performed and the input specification:

       COPY     = LIBRARY   Options: LIBRARY - Copy all files in a VS Library.
                                     SELECT  - Select files to be copied.
                                     FILE    - Specify one file to be copied.

       VOLUME   = WP****

       LIBRARY  = DOCMNTR*

       FILE     = ********  Note: May only be specified when COPY = FILE.

  Or press PF1 to return to the function selection screen.
```

Figure 14-7. Sample VS to IBM Input Definition Screen

NOTE
Since IBM Data Exchange format diskettes can only contain consecutive files, IBMCOPY converts the copied VS indexed files to IBM consecutive files before the data is written onto the diskette.

The VS to IBM Input Definition screen enables you to identify the range and locations of VS files to be copied through the following fields:

Field	Description
COPY	Specify LIBRARY, SELECT, or FILE as the range of the copying operation. The default value for the Copy field is LIBRARY. LIBRARY — This value indicates that the entire VS library is to be copied to the IBM output diskette. Each file in the library is copied to a corresponding file on the IBM output diskette. You mustspecify the Library and Volume fields on the screen. Section 14.3.1 describes subsequent processing for library copying. SELECT — This value indicates that selected files of a VS library are to be copied to the IBM output diskette. You must specify the Library and Volume fields on this screen. Section 14.3.2 describes subsequent processing for selected file copying. FILE — This value indicates that a single file is to be copied from the VS to the IBM output diskette. You must specify the File, Library, and Volume fields on this screen. Section 14.3.3 describes subsequent processing for single file copying.
VOLUME	Specify the name of the VS disk volume on which the VS input file(s) resides. The default value is your default input volume (INVOL) which is set through the SET Usage Constants function (PF2) from the Command Processor menu.

Field	Description
LIBRARY	Specify the name of the VS library that is to be copied or that contains the file(s) to be copied. The default value is your default input library (INLIB).
FILE	Specify the name of the file to be copied. (Do not enter a file name when the input range spans more than one file.)

14.3.1 Copying a Library from a VS Disk to an IBM Diskette Volume

If you specify LIBRARY for the Copy field, identify the input library on the VS to IBM Input Definition screen (Figure 14-7) and specify the output diskette (along with the copy options); the Copy VS Library Output Definition screen (Figure 14-8) appears.

```
***  MESSAGE VI00 BY IBMCPY

                INFORMATION REQUIRED BY  PROGRAM  IBMCOPY
                          TO DEFINE OUTPUT
                    ACTIVE SUBPROGRAM IS IBMCOPY


 Please enter the output volume, pad character, and translation options:

      VOLUME   = ******

      PADCHAR  = *    (blank or zero) used to extend variable length records.

      TRANSL   = YES  (YES or NO ) Translate the files from ASCII to EBCDIC?

 Or press the appropriate PF key to select the desired option:

      PFKEY    FUNCTION

       (1)     Return to the input selection screen.
       (2)     Specify output attributes individually for each file.

      (16)     Abort this operation.
```

Figure 14-8. Sample Copy VS Library Output Definition Screen

From the Copy VS Library Output Definition screen, you are prompted to specify the following fields:

Field	Description
VOLUME	Specify the IBM output diskette.
PADCHAR	Specify (blank or zero) a pad character that is used to extend VS variable-length records to IBM fixed-length records
TRANSL	Specify (YES or NO) whether you want to translate the files from ASCII to EBCDIC.

The VS file name for each file in the library becomes the IBM output file name.

14.3.2 Specifying Single File Options When Copying Multiple Files

To respecify the file name or select copy options separately for each file, press PF2 from the Copy VS Library Output Definition screen. An IBMCOPY Single File Output Definition screen is displayed for each file in the library. The output options that you specified on the Copy VS Library Output Definition screen are displayed as default values on each IBMCOPY Single File Output Definition screen. If you want, you can change these values. You can cancel this operation and return to the IBMCOPY Function Selection screen by pressing PF16.

NOTE
If the IBM output diskette already contains files, you must either add the library's files to the diskette or delete all files already on the diskette before you copy the files.

The Newindex field appears on a separate screen immediately following the Copy VS Library Output Definition screen (Figure 14-8). If you define the output options separately for each file, the screen containing the Newindex field appears after the first IBMCOPY Single File Output Definition screen.

If you specify NO in the Newindex field (the default response), the files are added to the current diskette contents. If you specify YES in the Newindex field, any files existing on the diskette are deleted before the VS files are copied. If sufficient space is not available on the IBM output diskette to hold all the files in the library, you can

- Skip the file that is currently being processed (PF2).
- Continue the copy (making a multivolume file) by pressing (PF3).
- Mount a new diskette that can hold the entire file by specifying the name of the new diskette (ENTER).

If one diskette becomes full during a multivolume copy, IBMCOPY prompts you to mount a new IBM diskette. If any input file name duplicates a file name already on the diskette, IBMCOPY signals the error and enables you to perform one of the following functions:

- Respecify the input file name
- Skip the file and continue (PF2)
- Delete the file on the diskette (PF3)
- Terminate the copy (PF16)

After you copy the files, IBMCOPY displays the IBMCOPY End-of-Job menu. From the IBMCOPY End-of-Job menu, you can copy more data by pressing PF1, return to the IBMCOPY Function Selection screen by pressing PF2, or exit IBMCOPY by pressing PF16. Dismount the IBM diskette through the Command Processor after exiting IBMCOPY.

14.3.3 Copying Selected Files from a VS Disk to an IBM Diskette Volume

If you copy only selected files from a VS library, IBMCOPY displays a file selection screen. The file selection screen displays a listing of files in the VS library and the total number of files in the library. If more than 20 files reside in the library, the file listing (up to 20 files per screen) continues on a following screen. You can return to the VS to IBM Input Definition screen (Figure 14-7) by pressing PF1.

To select a file from the list to be copied, type any alphanumeric character in the field next to the desired file name. To display any subsequent file selection screen(s), press ENTER. To end input definition and to continue processing, press ENTER when the last file selection screen is displayed. You can complete your file selection without traversing subsequent screens; for example, if you complete your file selection on the second screen out of six file selection screens, you can skip screens three through six and continue processing by pressing PF16.

At this point, the Selected VS Files Output Definition screen appears. Identify the IBM output diskette, specify a pad character used to extend VS variable-length records to IBM fixed-length records, and select character set conversion. The VS file name for each selected input file is used as the output IBM file name.

NOTE
The Selected VS Files Output Definition screen is similar to the IBMCOPY Single File Output Definition screen (refer to Figure 14-6, discussed in Section 14.2.3), except that the file name is not defined in the Selected VS Files Output Definition screen.

Refer to Section 14.3.2 if you want to respecify the file name and/or select copy options separately for each file.

14.3.4 Copying a Single File from a VS Disk to an IBM Diskette Volume

After you select a single file copy option and identify the input file on the VS to IBM Input Definition screen, you specify the output file, along with copy options, on the Single File Output Definition screen (Figure 14-9). Because all IBM files on IBM Data Exchange format diskettes have fixed-length records, the Single File Output Definition screen enables you to select the pad character used to convert VS variable-length records. You can also select ASCII to EBCDIC character set conversion.

```
***  MESSAGE VI00 BY IBMCOPY

                INFORMATION REQUIRED BY  PROGRAM  IBMCOPY
                          TO DEFINE IOUTPUT
                     ACTIVE SUBPROGRAM IS IBMCOPY


 Please enter the output file, volume, pad character, and translate option for
      the input file FILENAME in LIBNAME  on VOLNAM:

      FILE     = FILENAME on VOLUME    = ******

      PADCHAR  = *     (blank or zero) used to extend variable length records.

      TRANSL   = YES   (YES or NO ) Translate the files from ASCII to EBCDIC?

      EXTRA    = 000   Number of extra records to be allocated for the file

   Or press PF16 to abort this operation.
```

Figure 14-9. Sample Single File Output Definition Screen

Specify the output file location and copy options through the Single File Output Definition screen fields, described as follows:

Field	Description
FILE	Specify the name of the output IBM file. You can supply an alphanumeric value of up to eight characters.
VOLUME	Specify the name of the output IBM diskette. If the specified IBM diskette is not mounted, IBMCOPY prompts you to mount the diskette, as described in Section 14.6. If the specified IBM diskette is not initialized, IBMCOPY enables you to initialize the volume without returning you to the IBMCOPY Function Selection screen (Figure 14-2). There is no default value unless you rerun the VS to IBM copy sequence. Then, the Volume field defaults to the previously entered value.
PADCHAR	Specify the character used to extend variable-length VS records to fixed-length IBM records. Pad characters are hexadecimal zeros or blanks; the Padchar field defaults to blanks.
TRANSL	Specify whether the file is translated from ASCII to EBCDIC. The file is translated if you specify YES in the Transl field. The file remains in the ASCII character set if you specify NO. The default value is YES. **NOTE** *Specify NO in the Transl field for files that contain numeric data because IBMCOPY performs only character translation. Files that contain numeric data can be translated through the TRANSL utility (refer to Chapter 27).*
EXTRA	Specify the number of extra records that are to be allocated for the file. The default value is 000.

Refer to Section 14.3.2. for information concerning the options that are available to you when copying a single VS file to an IBM output diskette that already contains files.

14.4 INITIALIZE THE DISKETTE USING AN IBM FORMAT (PF3)

IBM Data Exchange format diskettes differ in format from VS soft-sectored diskettes. Consequently, initializing an IBM diskette cannot be performed through the DISKINIT utility. However, the IBMCOPY utility can initialize and format soft-sectored diskettes to IBM Data Exchange formats. IBMCOPY initializes the following types of diskettes to the indicated formats:

Diskette Type		IBM Data Exchange Format	Bytes per Sector
SSSD	to	Basic	128
DSSD	to	Basic	128
DSDD	to	Type H	256

The Initialize IBM Diskette screen is displayed if you press PF3 from the IBMCOPY Function Selection screen (Figure 14-2).

Specify the name of the diskette to be initialized. The name that you specify becomes the IBM volume name. If the specified volume is not mounted, mount the diskette, as described in Section 14.6. The drive detects whether the disketteis single- or double-sided. IBMCOPY processing differs for each diskette type.

14.4.1 Initializing IBM Single-Sided Diskette Volumes

If you use a single-sided diskette volume, IBMCOPY displays a screen acknowledging that the diskette is single-sided and a warning that the initialization process destroys any data on the diskette. To return to the Initialize IBM Diskette screen, without initializing or destroying the contents of the diskette, press PF1. Press ENTER to acknowledge the message and continue the initialization.

IBMCOPY initializes the diskette to the IBM Basic Data Exchange format. When the initialization is complete, the IBMCOPY End-of-Job menu is displayed. The IBMCOPY End-of-Job menu enables you to initialize another diskette by pressing PF1, to return to the IBMCOPY Function Selection screen by pressing PF2, or to exit IBMCOPY by pressing PF16.

14.4.2 Initializing IBM Double-Sided Diskette Volumes

If you use a double-sided diskette volume, IBMCOPY displays a screen acknowledging that the diskette is double-sided and a warning that the initialization process destroys any data on the diskette. To return to the Initialize IBM Diskette screen, without initializing or destroying the contents of the diskette, press PF1. Press ENTER to acknowledge the message and continue the initialization.

Specify the desired density format through the Format field. (IBMCOPY formats double-sided diskettes to single- or double-density.) If you specify BASIC in the Format field (the default value), IBMCOPY initializes the diskette as an IBM Basic Data Exchange format diskette in single-density. If you specify H in the Format field, IBMCOPY initializes the diskette as an IBM Type H Data Exchange format diskette in double-density. PF1 can still be pressed at this point to return to the Initialize IBM Diskette screen.

Press ENTER to begin initialization. IBMCOPY initializes the diskette to the selected format. When the initialization is complete, the IBMCOPY End-of-Job menu is displayed. The IBMCOPY End-of-Job menu enables you either to initialize another diskette by pressing PF1, to return to the IBMCOPY Function Selection screen by pressing PF2, or to exit IBMCOPY by pressing PF16.

14.5 DISPLAY AN IBM FORMAT DISKETTE DIRECTORY (PF5)

You can display or print the contents of any IBM diskette directory by pressing PF5 from the IBMCOPY Function Selection screen (Figure 14-2). IBMCOPY can only transfer IBM Basic and Type H Data Exchange files, but it can read any IBM diskette directory. IBMCOPY also determines whether it can transfer the resident files by reading the directory.

If you press PF5, specify the IBM volume and the desired type of directory report on the resulting screen through the following fields:

Field	Description
VOLUME	Specify the name of the IBM volume diskette for which you want to generate a directory report. The Volume field defaults to the name of the volume last specified through this function of IBMCOPY. If no volume was last specified, the field is blank. If the specified diskette is not mounted, mount the diskette as described in Section 14.6.
DEVICE	Specify PRINTER or DISPLAY to determine whether you want to display the directory report at the workstation or create a print file. A value of PRINTER creates a print file. A value of DISPLAY specifies a displayed directory report. The Device field defaults to DISPLAY, or to the value last specified through this function of IBMCOPY. A printed report contains directory information for all fields (except those fields described by IBM as reserved or padded). A displayed report also contains all the volume information, but contains only those fields from the HDR1 (Header 1) records that are usually of interest.

NOTE
You can print the full report at any time while displaying the report. Therefore, you can view your report before printing it out.

IBMCOPY also contains a hidden GETPARM (identified by the prname Special) that enables you to include deleted HDR1 records in the directory report. The screen is not displayed because the deleted records are usually only required to analyze a problem. The deleted HDR1 records are included if DELETED = YES, and omitted if you accept the default value of NO.

The following VS procedure language routine enables you to display the hidden GETPARM. Section 14.7 has more on sample Procedure language implementation of IBMCOPY.

```
PROCEDURE
RUN IBMCOPY
DISPLAY SPECIAL
```

14.5.1 Printing an IBM Format Diskette Directory

If you specify PRINTER in the Device field, IBMCOPY creates a print file that contains a complete directory report for the IBM diskette. Each field in the report is described in Section 14.5.3. The print file name is constructed by concatenating the character string IBMC and a unique, 4-digit number. The file name is placed in SPOOLIB (your default print library, which is set through the SET Usage Constants function (PF2) from the Command Processor menu). The print file is printed according to your print mode defaults. Refer to the *VS System User's Introduction* for details about the effect of the default PRNTMODE value on print files.

14.5.2 Displaying an IBM Format Diskette Directory

If you specify DISPLAY in the Device field, the IBMCOPY Directory Dump menu (Figure 14-10) appears.

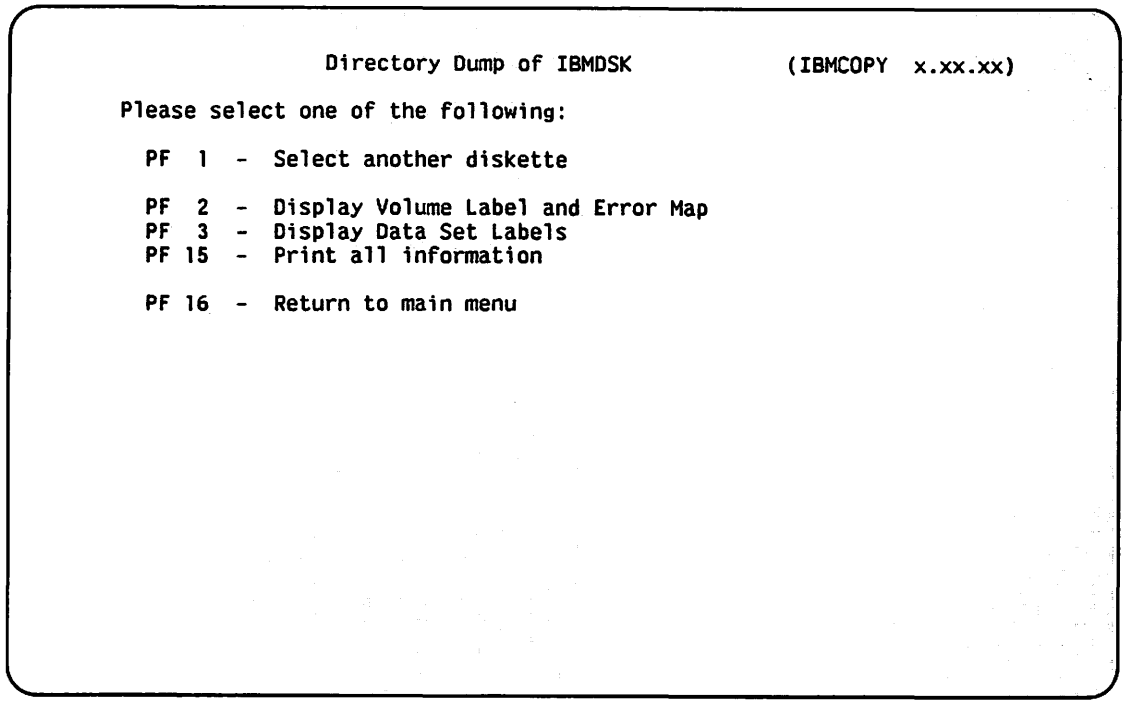


Figure 14-10. IBMCOPY Directory Dump Menu

You can perform the following functions from the IBMCOPY Directory Dump menu:

PF Key	Function	Description
1	Select Another Diskette	Generate a directory report for a different IBM diskette from the IBM Directory Display menu.
2	Display Volume Label and Error Map	Display volume label and error map directory information from the IBM Diskette Directory Display. The report contains the Volume ID, System Code, Owner ID, Volume Surface, Sector Length, Volume Access, Label Extension, Special Requirements, Extent Arrangement, and Sector Sequence fields. The report also indicates whether the volume contains any defective cylinders or alternative record relocation. Refer to Section 14.5.3 for a sample screen and details on each field in the directory.
3	Display Data Set Labels	Display directory information for each data set on the volume. The report contains the Label CCHSS Address, Dataset Name, Exchange Type, Multiple Volume Indicator, Volume Number, Block Length, Sector Length, Record Length, Beginning- of-Extent (BOE) Address, and End-of-Extent (EOE) Address fields, and indicates whether IBMCOPY can read the data set. Refer to Section 14.5.4 for a sample screen and details on each field in the directory.
15	Print All Information	Generate a printout that contains the entire directory report.
16	Return to the Main Menu	Cancel the display operation and return to the IBMCOPY Function Selection screen.

14.5.3 Analyzing the Volume Label and Error Map

When you press PF2 from the IBMCOPY Directory Dump menu, the IBMCOPY Volume Label and Error Map screen (Figure 14-11) appears.

```
                Directory Dump of IBMDSK              (IBMCOPY  x.xx.xx)

1-Menu  15-Print all information

            Volume  System Code     Owner Id              Volume     Sector
              ID                                          Surface    Length
            GLOP                                           DSDD        256

                   Volume  Label  Special  Extent  Sector
                   Access  Exten  Req'mnt  Arrang  Seq

ERMAP indicates no defective cylinders.

ERMAP indicates no alternative physical record relocation.
```

Figure 14-11. Sample IBMCOPY Volume Label and Error Map Screen

IBMCOPY directory reports contain volume information and data set information. Volume information consists of fields from the IBM diskette's VOL1 and ERMAP records, and data set information consists of fields from the diskette's HDR1 records. Printed reports (obtained by pressing PF15 from this screen) contain all of the available directory information. Displayed reports omit less useful fields in the HDR1 records. This section describes the fields in the VOL1, ERMAP, and HDR1 records.

Volume Information

The following fields are contained in both displayed and printed reports, and reside in the diskette's VOL1 record:

Field	Description
Volume ID	The Volume ID field displays the name of the volume when the diskette was initialized.
System Code	The System Code field displays the name of the system that initialized the diskette. Diskettes initialized through IBMCOPY contain the value WANGVS in this field.
Owner ID	The Owner ID field displays the name of the owner of the diskette.
Volume Surface	The Volume Surface field displays the type of diskette as recorded in the diskette directory. This field can contain a value of DSDD (double-sided, double-density), DSSD (double-sided, single-density), or SSSD (single- sided, single-density).

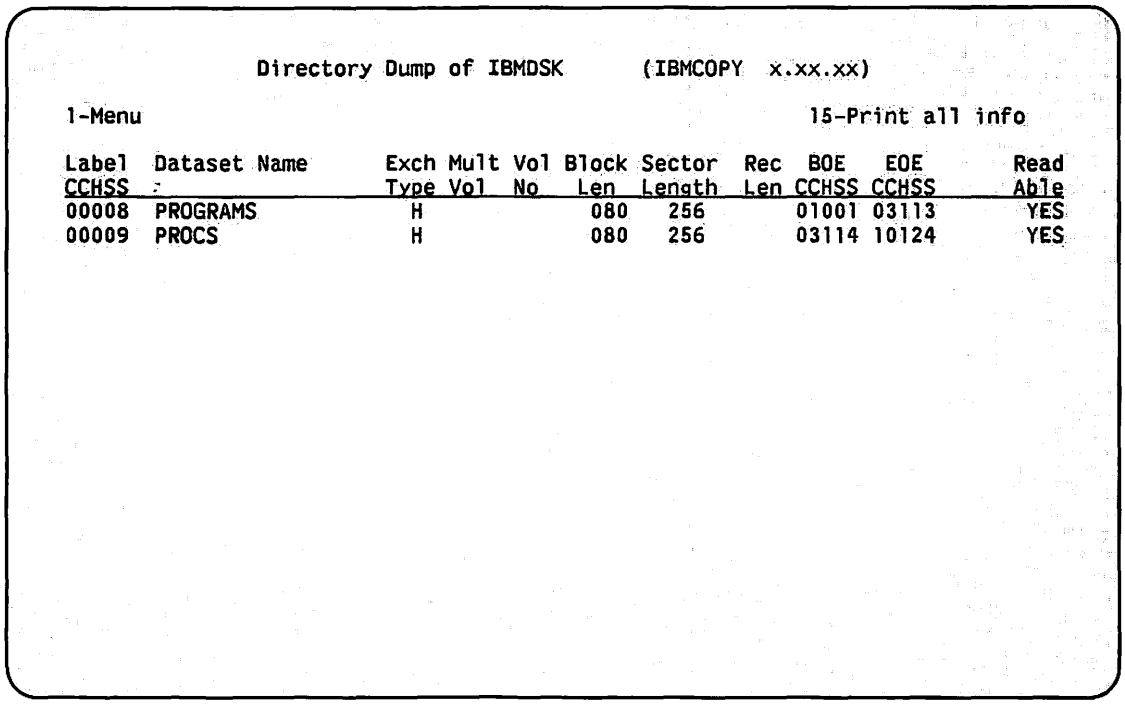
Field	Description
Sector Length	The Sector Length field displays the length of all sectors on the diskette on Cylinders 1 through 76. A value in parentheses indicates that the diskette contains an invalid value. Because IBMCOPY can only read Basic Data Exchange and Type H diskettes, it can only read diskettes with 128 or 256 sector lengths.
Volume Access	The Volume Access field displays a character which shows whether more qualifications are necessary to access the diskette.
Label Exten	The (Label Extension) field displays the number of additional cylinders reserved for the diskette directory. IBMCOPY can only copy data from diskettes with a space in this field.
Special Req'mnt	The Special Req'mnt (Special Requirement) field indicates whether special requirements must be met to access data on this volume. IBMCOPY can only copy data from the diskette if this field contains a space.
Extent Arrang	The Extent Arrang (Extent Arrangement) field indicates whether data sets and labels must be contiguous on the diskette.
Sector Seq	The Sector Seq (Sector Sequence) field displays a number that determines the logical sequence of the physical sectors on the diskette. IBMCOPY requires a value of spaces, 0, or 1 to read files on the diskette.

ERMAP Fields

The ERMAP record determines whether the diskette has any defective cylinders or any physical record relocation. If the diskette has defective cylinders, the directory report lists them in the format "Defective cylinders: xx xx." If thediskette contains physical record relocation, the directory report indicates the defective record method and lists the cylinder, head, and sector addresses of the defective records.

14.5.4 Analyzing the Data Set Labels

When you press PF3 from the IBMCOPY Directory Dump menu, the IBMCOPY Data Set Labels screen (Figure 14-12) appears.



```
                Directory Dump of IBMDSK      (IBMCOPY  x.xx.xx)

1-Menu                                                  15-Print all info

Label  Dataset Name      Exch Mult Vol Block Sector  Rec  BOE   EOE       Read
CCHSS                    Type Vol  No  Len   Length  Len CCHSS CCHSS      Able
00008  PROGRAMS            H              080   256       01001 03113      YES
00009  PROCS               H              080   256       03114 10124      YES
```

Figure 14-12. Sample IBMCOPY Data Set Labels Screen

HDR1 Fields

The following IBMCOPY Data Set Labels screen fields are contained both in the displayed and printed reports, and reside in the diskette's HDR1 records. The IBMCOPY Data Set Labels screen fields are described as follows:

Field	Description
Label CCHSS	The Label CCHSS (cylinder/cylinder/head/sector/sector) field displays the cylinder, head, and sector address of the data set's entry in the label.
Dataset Name	The Dataset Name field displays the IBM name of the data set.
Exch Type	The Exch Type (Exchange Type) field displays the type of diskette. IBMCOPY can read only diskettes in the Basic Data Exchange (BDE) or Type H (H) formats.
Mult Vol	The Mult Vol (Multiple Volume) field determines whether the data set spans more than one volume. If the field is blank, the data set resides only on this volume. A value of C indicates that the data set is continued to another diskette. A value of L indicates the last diskette in a series.
Vol No	The Vol No (Volume Number) field displays the sequence number of the current volume (if the data set spans more than one volume).
Block Length	The Block Len (Block Length) field displays the length of the blocks in the data set.
Sector Length	The Sector Length field displays the physical record length (128, 256, 512, or 1024 bytes).
Rec Length	The Rec Len (Record Length) field displays the length of the records in the data set.

Field	Description
BOE CCHSS	The BOE CCHSS (cylinder/cylinder/head/sector/sector) field displays the cylinder, head, and sector address of the beginning-of-extent (BOE) pointer.
EOE CCHSS	The EOE CCHSS (cylinder/cylinder/head/sector/sector) field displays the cylinder, head, and sector address of the end-of-extent (EOE) pointer.
Readable	This field determines whether IBMCOPY can read the data set.

The following IBMCOPY Data Set Labels fields are contained only in the printed reports:

Field	Description
R F	The R F field displays the record/block format that indicates fixed-length records residing in fixed-length blocks. This field contains a blank if the Exchange Type Indicator field is H or BASIC (therefore, IBMCOPY can only copy data from diskettes with a blank in this field). Valid R F field values are blank and H.
B I	The B I field displays the bypass indicator that determines if the data set is to be skipped in copy operations on IBM systems; IBMCOPY does not use this field. If the field is blank, the data set can be transferred; if the field contains a B, the data set is skipped when copying the diskette on an IBM system.
D S	The D S field displays the degree of data set security on IBM systems; IBMCOPY does not use this field.
W P	The W P field indicates whether the data set can be written. If the field is blank, the data set can be both read or written. If the field contains a value, the data set can only be read.
Creation Date	The Creation Date field displays the date that the data set was created.
Offset to Next Record Space	The Offset To Next Record Space field indicates the starting position of the next record space in the data set's last sector (EOD) for blocked records.
Expiration Date	The Expiration Date field indicates when the data set can be deleted or written over.
Verify/ Copy Indicator	The Verify/Copy Indicator indicates whether the data set has been verified (V) or copied (C).
Data Set Organization	The Data Set Organization field indicates whether defective records can be relocated.
EOD (End-Of-Data) Address	The EOD field displays the address of the next unused sector within the data set extent. If the address is the same as the BOE address, the data set contains no records.

14.6 MOUNTING IBM DATA EXCHANGE FORMAT DISKETTES (PF4)

IBM Data Exchange format diskettes cannot be mounted through the VS mount procedure (PF6 from the Command Processor menu). You must mount all IBM diskettes through IBMCOPY. IBMCOPY mounts diskettes either directly from the IBMCOPY Function Selection screen (Figure 14-2) or through a processing function (when necessary). You can dismount the IBM diskette through the Procedure language DISMOUNT statement (refer to Section 14.7) or through the Manage DEVICES option (PF6) from the Command Processor menu.

If you press PF4 from the IBMCOPY Function Selection screen, the Mount IBM Diskette screen is displayed. Specify the values for the following fields:

Field	Description
VOLUME	Specify the name of the diskette to be mounted. You must specify a Wang VS volume name, regardless of the IBM volume name of the diskette. However, if IBMCOPY has previously mounted an IBM diskette, the Volume field defaults to the name of the most recently mounted volume.
DEVICE	Specify the device number of the soft-sectored diskette drive. (If associated with an archiving workstation, the drive's device number is generally one greater than the workstation number.) If you are running IBMCOPY at an archiving workstation, the Device field defaults to the device number of the associated diskette drive. However, if IBMCOPY has previously mounted an IBM diskette, the Device field defaults to the number of the most recently used diskette drive.

NOTE
If you have specified the name of an unmounted IBM diskette as input to another IBMCOPY function, the resulting Mount IBM Diskette screen requests only a Device value. In addition, this Mount IBM Diskette screen enables you to return to theprevious screen by pressing PF1 to respecify the volume name.

After you specify the fields on the Mount IBM Diskette screen, insert the diskette into the drive. After the diskette is mounted, program control returns to the IBMCOPY Function Selection screen or to the processing function, depending on which mode initiated the mount.

14.7 A SAMPLE IBMCOPY PROCEDURE

You can control IBMCOPY processing through the VS Procedure language. You can specify all IBMCOPY options and mount operations through a procedure. A complete list of IBMCOPY GETPARMs is provided in Appendix A; consult the *VS Procedure Language Reference* for details concerning VS Procedure language syntax.

The following procedure mounts an IBM diskette and copies all files on the IBM diskette into a new library on the VS SYSTEM volume. Once the files are copied, the procedure exits IBMCOPY and dismounts the IBM volume.

NOTE
The procedure dismounts the volume after IBMCOPY processing terminates because VS Procedure language DISMOUNT statements cannot be embedded in a RUN, ENTER, or DISPLAY sequence. The IBMCOPY Mount routine can be embedded in a procedure since the Mount routine does not use the VS Procedure language MOUNT statement.

This procedure totally automates IBMCOPY processing; you only physically mount and dismount the IBM diskette.

```
PROCEDURE
    RUN IBMCOPY
        ENTER FUNCTION 4
        ENTER MOUNTCOM DEVICE=039, VOLUME=IBMVOL
        ENTER FUNCTION 1
        ENTER OPTIONS COPY=VOLUME, VOLUME=IBMVOL
        ENTER OUTPUT LIBRARY=YOURLIB, VOLUME=SYSTEM,
            TRANSL=NO
        ENTER EOJ 16
DISMOUNT DISK IBMVOL
RETURN
```

CHAPTER 15
THE IOELOG UTILITY

15.1 INTRODUCTION

The IOELOG utility enables you to examine the contents of an input/output (I/O) error log file. You can use this utility to evaluate the history and condition of equipment configured to a system. IOELOG enables you to review the types of errors that occur and how often they occur. The I/O error log resides in the @SYSLOG file in the system Work library (@SYSWORK) on the system volume. The error log file exists even if it contains no errors. The VS Operating System records each occurrence of an error and produces an error log entry that contains the date and the time that each error occurred.

IOELOG provides the following functions:

- You can analyze the entire I/O error log (or portion of it) at your workstation. If you analyze the error log, IOELOG summarizes the error log contents into the following categories:
  - Standard Errors (errors that are logged against any of the five device classes)
  - Nonstandard Errors (logged errors that are not specifically related to one of the five device classes)
  - System Initial Program Loads (IPLs)
- You can print the entire contents of the I/O error log (or a portion of it). The printout lists the errors sequentially by the date and the time they occurred. They are not sorted by the type of error.

To analyze or print the error log, you must first copy it into a separate file from the Operator Console menu. Since the current log must remain open to receive any newly detected errors, you cannot use the active error log file as input. You can purge the error log file later from the same menu. If you use the purge option, the system then creates a new error log file.

15.1.1 Types of Errors that are Logged

There are five types of standard I/O errors that are logged on the VS. Standard errors are logged against the following five system devices:

- Communications
- Disk drives
- Printers
- Tape drives
- Workstations

Standard I/O errors occur as a result of either a Start I/O (SIO) instruction or a Control I/O (CIO) instruction. (For more information about these instructions, refer to the *VS Principles of Operation*.) SIO instructions initiate data transfer or a control operation. The Write command (write data onto a disk) is an example of an SIO instruction that transfers data. The Seek command (search for data on a disk) is an example of an SIO instruction that controls the operation of an I/O device. CIO instructions have two primary purposes:

- To control memory diagnostics for I/O processors
- To control microcode reading, microcode loading, and programmable processing

An example of a memory diagnostic command is the Reference command, which reads specified memory in the I/O processor and compares what was read to a given data pattern. The Restart Bus Processor command and the Start Data Link Processor command are examples of microcode reading and loading, and programmable processor control.

Both types of Standard I/O errors are classified as hard or soft errors. A soft error occurs when a command is successfully completed after one or more failures. If this is the case, the error is considered "recoverable." A hard error occurs when a command is not successfully completed after a number of attempts, in which case, the error is not recoverable.

Nonstandard I/O errors (refer to Section 15.5) include missing interrupts (delayed device response), lost extents to the free extent list in the VTOC on a disk (reported by the Supervisory Calls (SVCs) SCRATCH and UPDATFDR), page-in/page-out errors, redundant VTOC compromises (an I/O command to the VTOC failed due to a hard disk error), and machine check errors. The error log also includes an IPL summary, which lists the dates and times of each system IPL.

15.1.2 The IOELOG On-line Help Facility

IOELOG provides you with access to an on-line help facility for information about the utility. By pressing Program Function (PF) key 13, you can enter the Help facility from most of the IOELOG screens. Pressing PF13 from a given screen brings you to the on-line Help text section for that screen. Pressing PF1 from within the Help facility returns you to IOELOG.

15.2 COPYING THE ERROR LOG

To copy an I/O error log file, press PF14 (SYSTEM Options) from the Operator's Console menu. When you press PF14, the Systems Options screen appears. From the System Options screen, press PF9 to copy the I/O error log. The Copy I/O Error Log screen (Figure 15-1) appears and prompts you to specify the names of the file, library, and volume into which the I/O error log is to be copied. Also specify in the Purge field whether you want to purge the error log after copying it. When you press ENTER, the error log is copied to the file and a completion message appears at the top of the screen. Press HELP to return to the Operator's Console menu.

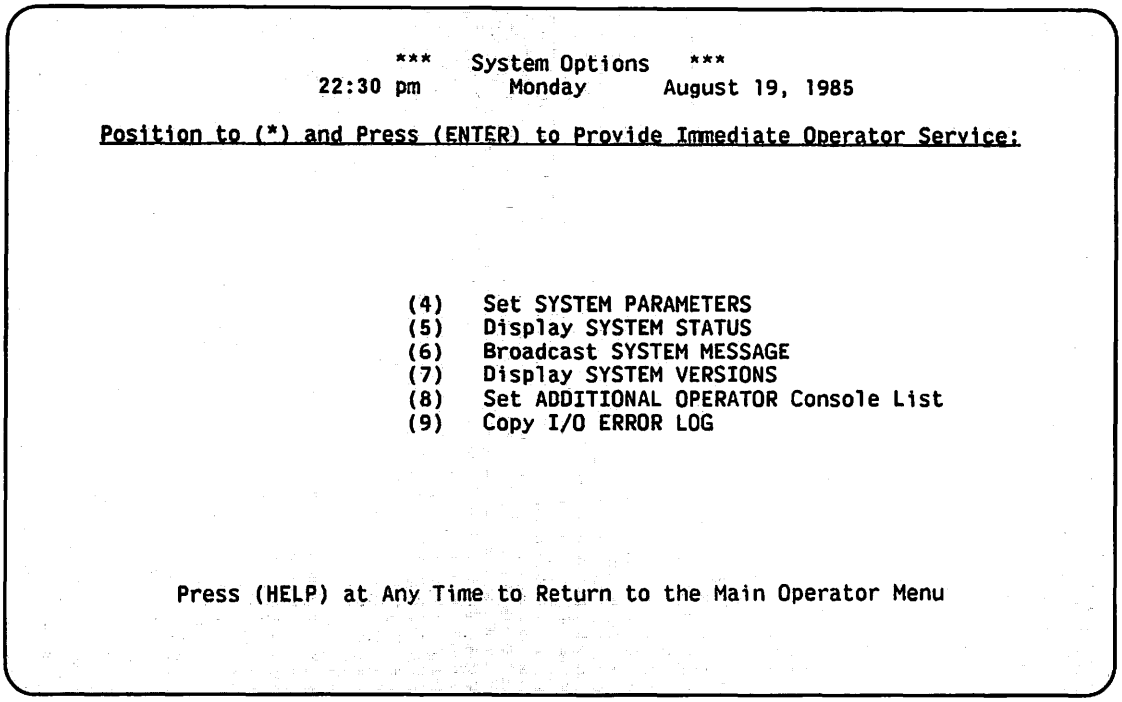


Figure 15-1. Copy I/O Error Log Screen

NOTE

*It is important to **print** and **then purge** the I/O error log on a regular basis. In doing this, you can reduce the amount of disk space that is occupied by the log and be regularly informed of potential problems before they become serious.*

15.3 IOELOG PROCESSING

Figure 15-2 shows an overview of IOELOG processing.

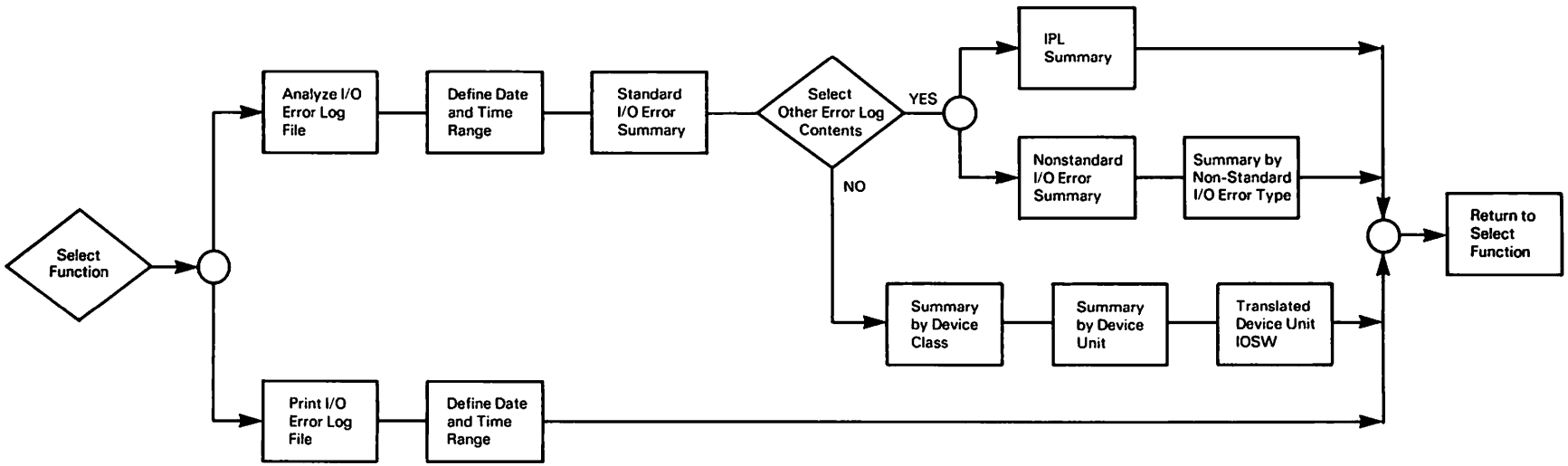


Figure 15-2. IOELOG Processing

To run IOELOG, press PF1 (RUN Program or Procedure) from the Command Processor menu, specify IOELOG in the Program field, and press ENTER. The IOELOG Function Definition screen (Figure 15-3) then appears.

```
***  MESSAGE I001 BY IOELOG

                INFORMATION REQUIRED BY  PROGRAM  IOELOG
                            TO DEFINE FUNCTION


                ***   Wang VS I/O Error Log Program   ***

IOELOG is used to analyze and print the contents of an I/O Error Log.

Please specify the name of an I/O error log file to be analyzed or printed
and select the function to be performed.

       FILE     = ERRLOG** in LIBRARY  = LOG***** on VOLUME   = SYSTEM

Functions:

 (1) Analyze - Analyze the contents of an I/O Error Log file.
 (2) Print   - Print the contents of an I/O Error Log file.
(13) Info    - Obtain help information.
(16) Exit    - End the program.
```

Figure 15-3. Sample IOELOG Function Definition Screen

A description of the IOELOG Function Definition screen fields are as follows:

Field	Description
FILE	Specify the name of the copied I/O error log file that you want to analyze or print.
LIBRARY	Specify the name of the library in which the copied I/O error log file resides. The Library field defaults to your input library (INLIB) that you set previously through a procedure or through the SET Usage Constants (PF2) function of the Command Processor menu.
VOLUME	Specify the name of the volume that contains the input library. The Volume field defaults to your input volume (INVOL) that you set previously through a procedure or through the SET Usage Constants function (PF2) of the Command Processor menu.

15.4 ANALYZING THE CONTENTS OF AN I/O ERROR LOG FILE

To analyze an I/O error log at your workstation, enter the log file, library, and volume names and press PF1 from the IOELOG Function Definition screen. When PF1 is pressed, the IOELOG Range Definition screen (Figure 15-4) appears and displays the name and the location of the error log file to be analyzed. The date and time range of current error log entries is also displayed, including the number of records in the range.

```
***  MESSAGE I002 BY IOELOG

                INFORMATION REQUIRED BY  PROGRAM  IOELOG
                            TO DEFINE RANGE


                ***   Wang VS I/O Error Log Program   ***


                                Analysis Option

File to be analyzed is:       ERRLOG   in LOG      on SYSTEM
Date and time range is:       1/02/85  @ 11:29     to   5/09/85  @  9:40
# records in range is:       10202

Please specify the desired time range of the file to be analyzed:

                STARTDAY = *5/08/85       MM/DD/YY
                STARTIME = 00:01          HH:MM  (24 hour clock)
                ENDDAY   = *5/09/85       MM/DD/YY
                ENDTIME  = *9:40          HH:MM  (24 hour clock)

              Press PF1 to return to the function selection menu.
```

Figure 15-4. Sample IOELOG Range Definition Screen

The IOELOG Range Definition screen provides fields that default to the date and time range of the error log. To view only a portion of the error log, just modify the following fields:

Field	Description
STARTDAY	Startday refers to the starting date used in error log analysis, in the form MM/DD/YY. The default value for Startday is the date of the first entry in the error log.
STARTIME	Startime refers to the starting time used in error log analysis, in the form HH:MM as based on a 24-hour clock. The default value for Startime is the time of the first entry in the error log.
ENDDAY	Endday refers to the end date used in error log analysis, in the form MM/DD/YY. The default value for Endday is the date of the last entry in the error log.
ENDTIME	Endtime refers to the end time used in error log analysis, in the form HH:MM as based on a 24-hour clock. The default value for Endtime is the time of the last entry in the error log.

You must use the format provided for Startday and Endday as it appears on the screen. If you use another format or if you enter values greater than 23:59 in the Startime or Endtime fields, an error message appears (and the field in error flashes). You can then respecify a correct value.

When you have defined the time range in which you are analyzing the I/O error log, press ENTER to continue processing. You can press PF1 to return to the IOELOG Function Definition screen if you do not want to continue.

After you press ENTER, the Standard I/O Error Summary screen appears. This screen contains a summary of standard I/O errors logged against the five device classes. You can obtain more specific information about I/O errors on specific devices. The following levels of information can be obtained for each device class:

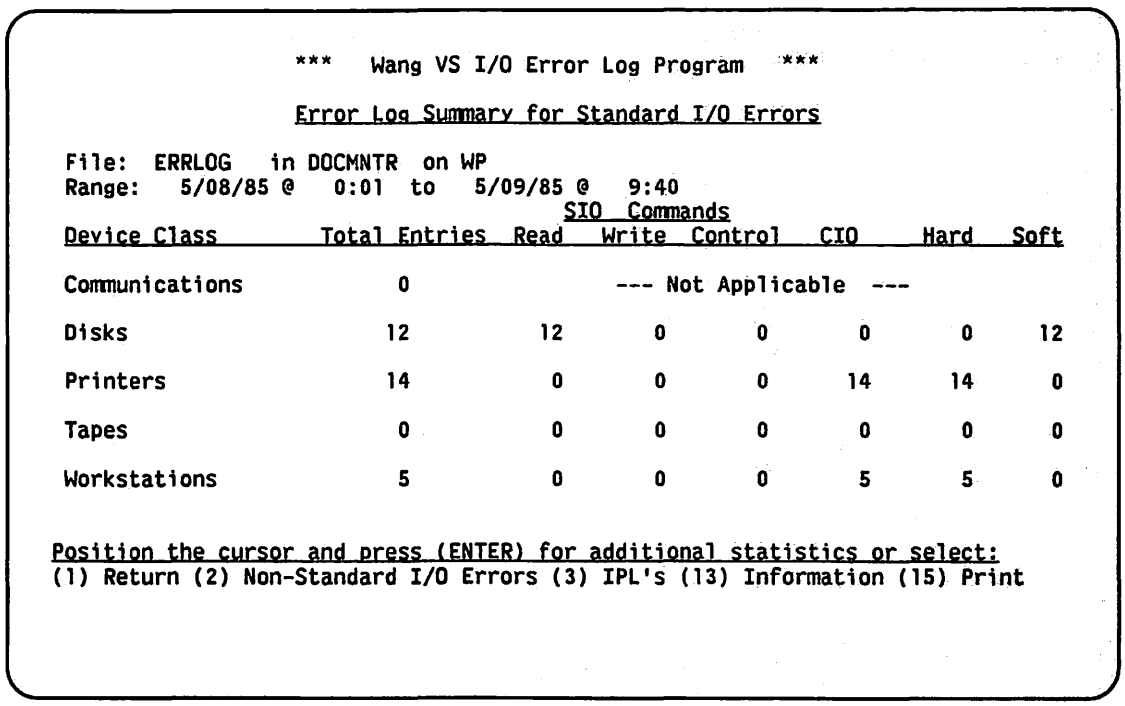
Level	Description
All Device Classes	The Standard I/O Error Summary displays the total number of errors for each of the five device classes.
Specific Device Class	The (Device) I/O Error Summary displays the total number and type of errors for all the devices of the same class.
Specific Device Unit	The (Device) Unit Summary displays information about specific errors on a specific device.
Device Unit IOSW	The Translated (Device) IOSW displays specific information about what caused a specific error to occur on a device.

15.4.1 The Standard I/O Error Summary

The Standard I/O Error Summary screen (Figure 15-5) displays the number of errors that are logged against each of the five device classes during the specified time range:

- Communications
- Disks (drives)
- Printers
- Tapes (drives)
- Workstations

The name and location of the error log file appear at the beginning of the summary, as well as the specified time range of the error log. The total number of Standard I/O errors logged against each device class is divided into two categories: errors that occurred as a result of an SIO or CIO instruction, and those errors that are classified as hard or soft. SIO instruction errors are divided into three categories: read, write, and control. The total number of CIO instructions for each device appears in one column. Hard and soft errors for each device are listed in separate columns.



```
                    ***   Wang VS I/O Error Log Program   ***

                    Error Log Summary for Standard I/O Errors

File:  ERRLOG   in DOCMNTR  on WP
Range:   5/08/85 @   0:01  to   5/09/85 @   9:40
                                          SIO  Commands
Device Class          Total Entries  Read   Write  Control   CIO     Hard   Soft

Communications              0                --- Not Applicable  ---

Disks                      12          12      0        0       0       0      12

Printers                   14           0      0        0      14      14       0

Tapes                       0           0      0        0       0       0       0

Workstations                5           0      0        0       5       5       0


Position the cursor and press (ENTER) for additional statistics or select:
(1) Return (2) Non-Standard I/O Errors (3) IPL's (13) Information (15) Print
```

Figure 15-5. Sample Standard I/O Error Summary Screen

The Standard I/O errors contained in the log are categorized as follows:

Error Type	Description
SIO Read	An SIO Read error count refers to the total number of errors that occurred for a given device class while reading data as a result of a Start I/O instruction.
SIO Write	An SIO Write error count refers to the total number of errors that occurred for a given device class while writing data as a result of a Start I/O instruction.
SIO Control	An SIO Control error count refers to the total number of errors that occurred for a device class during a control operation (such as, SEEK or FORMAT on a disk as a result of a Start I/O instruction). No transfer of data was involved.
CIO	A CIO error count refers to the total number of errors that occurred for a given device class while initiating memory diagnostic functions, loading or reading microcode, or performing processor control functions.
Hard	A hard error count refers to the total number of unrecoverable errors that occurred for a device class.
Soft	A soft error count refers to the the total number of recoverable errors that occurred for a device class.

NOTE
"Not Applicable" appears in the Communications device class because the errors cannot always be broken down into the categories listed. Only the total number of errors is given.

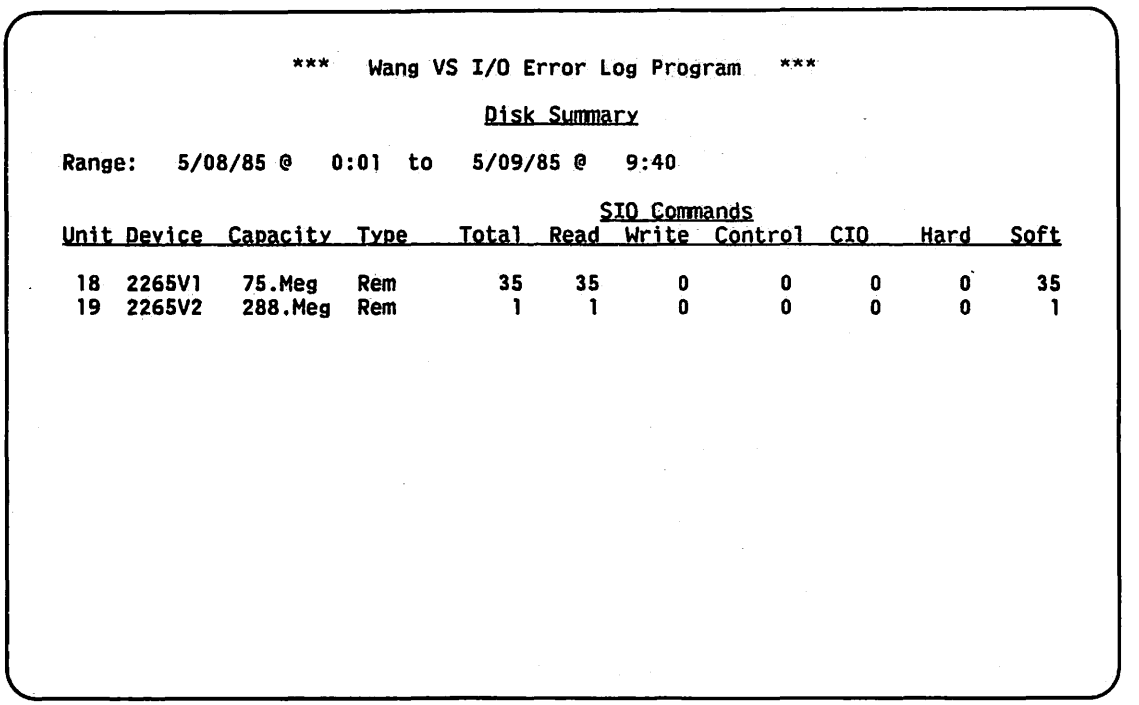
You can perform the following functions from the Standard I/O Error Summary screen by pressing the corresponding PF key:

PF Key	Function	Description
ENTER	Display Error Summary	Position the cursor next to a listed device class and press ENTER to display an error summary that lists the total number of Standard I/O errors that occurred for a particular device class.
1	Return	Return to the IOELOG Range Definition screen (Figure 15-4).
2	Nonstandard I/O Errors	Display the Nonstandard I/O Errors Summary, which lists I/O errors that are found in the error log and not reported against one of the five device classes. (For a description of these errors, refer to Section 15.5.)
3	IPLs	Display the IPL Summary, which lists the date and time of each IPL of the system. (For a description of the IPL summary, refer to Section 15.6.)
13	Information	Display on-line help information.
15	Print	Print the contents of the Standard I/O Error Summary screen.

NOTE
For illustration purposes, disk devices will be used as examples in the following sections when describing the levels of errors occurring on specific device classes. Other devices are just as applicable.

15.4.2 The IOELOG Device Class Summary

To obtain a summary about a specific device class, position the cursor next to the desired device class on the Standard I/O Error Summary screen and press ENTER. For example, the IOELOG System Disks Error Summary screen (Figure 15-6) lists all of the disks on the system and reports the total number of specific errors that occurred on each disk.



```
                    ***  Wang VS I/O Error Log Program  ***

                                  Disk Summary

Range:   5/08/85 @   0:01  to   5/09/85 @   9:40

                                                        SIO Commands
Unit Device  Capacity  Type    Total  Read  Write  Control  CIO   Hard  Soft
 18  2265V1   75.Meg   Rem       35    35      0        0     0     0    35
 19  2265V2  288.Meg   Rem        1     1      0        0     0     0     1
```

Figure 15-6. Sample IOELOG System Disks Error Summary Screen

The IOELOG System Disks Error Summary screen also displays the total number of SIO and CIO instruction errors, the total number of hard and soft errors, and the total number of errors that occurred for each device unit. The unit number, device name, capacity, and type are also listed for each disk unit and are described as follows:

Column Head	Description
Unit	The Unit column displays the unit number of the disk drive.
Device	The Device column displays the name that the system recognizes for a given disk unit.
Capacity	The Capacity column displays the maximum amount of disk storage available on that disk.
Type	The Type column lists the types of disks that can be mounted on a specific disk drive. They include: Fix (Fixed), Rem (Removable), and Fix/Rem (Fixed/Removable).
Total	The Total column displays the number of errors on a particular disk drive as recorded by the error log.
SIO Read	An SIO Read error count refers to the total number of errors that occurred for a given disk drive while reading data as a result of a Start I/O instruction.
SIO Write	An SIO Write error count refers to the total number of errors that occurred for a given disk drive while writing data as a result of a Start I/O instruction.
SIO Control	An SIO Control error count refers to the total number of errors that occurred for a disk drive during a control operation (such as, SEEK or FORMAT on a disk as a result of a Start I/O instruction). No transfer of data was involved.

Column Head	Description
CIO	A CIO error count refers to the total number of errors that occurred for a given disk drive while initiating diagnostic functions, loading or reading microcode, or performing processor control functions.
Hard	A hard error count refers to the total number of unrecoverable errors that occurred for a given disk drive.
Soft	A soft error count refers to the the total number of recoverable errors that occurred for a given disk drive.

From the IOELOG System Disks Error Summary screen (Figure 15-6) and from subsequent screens, you can perform the following functions from each screen:

PF Key	Function	Description
ENTER	Display Next Error Level	Press ENTER (after positioning the cursor next to the item for which you want more detail) to display the next error level summary.
1	Return	Return to the previous error level summary.
2	First	Display the first screen of errors in the current error level summary (if more than one screen is required).
3	Last	Display the last screen of error types in the current error level summary (if the list requires more than one screen).
4	Prev	Display the previous list of error types in the current error level summary (if more than one screen is required).
5	Next	Display the next list of error types in the current error level summary (if the list requires more than one screen).
13	Information	Display on-line help information for the current error level summary.
15	Print	Print the screen contents of the current error level summary.

15.4.3 The IOELOG Device Unit Summary

If you want to analyze errors logged against a specific disk unit that is displayed on the IOELOG System Disks Error Summary screen, position the cursor next to a unit number and press ENTER. When this is done, the IOELOG Disk Unit Summary screen (Figure 15-7) appears and lists all the errors for that unit, allowing you to further analyze the types of errors recorded. The Sample IOELOG Disk Unit Summary screen includes the total number of errors that occurred on specific cylinders of the disk. In the sample shown in Figure 15-7, ten errors occurred in Cylinder 3 of Unit 18 (nine in Block 4 and one in Block 1).

```
                    ***   Wang VS I/O Error Log Program   ***

                                  Disk Unit Summary

Range:   5/08/85 @   0:01  to   5/09/85 @   9:40
Unit:          18
Device Type:  2265V1

Total  Cyl  Surf   Block  Volume      Command        Status  I/O Status Word

    9    3     1       4  OS      SIO, Read          SOFT   6010000400000100
    6    0     0       7  OS      SIO, Read          SOFT   6010000400000100
    4  710     0       8  OS      SIO, Read          SOFT   6010000400000100
    2  369     3       2  OS      SIO, Read          SOFT   6010000400000100
    2  710     0       8  OS      SIO, Read          SOFT   6810000400000100
    1   21     2       0  OS      SIO, Read          SOFT   6010000400000200
    1    3     2       1  OS      SIO, Read          SOFT   6010000400000100
    1  705     3       1  OS      SIO, Read          SOFT   6010000400000100
    1    1     4       2  OS      SIO, Read          SOFT   6010000400000300
    1  705     0       3  OS      SIO, Read          SOFT   6010000400000100
    1    0     0       4  OS      SIO, Read          SOFT   6010000400000100
```

Figure 15-7. Sample IOELOG Disk Unit Summary Screen

The IOELOG Disk Unit Summary screen categorizes the total number of errors that occurred for a given disk unit by listing the number of errors unique to a particular block, cylinder, surface, command, status, and I/O Status Word (IOSW). The IOSW is a hexadecimal representation of the status that a device returns in response to an I/O instruction that the operating system issues. A description of the fields on the IOELOG Disk Unit Summary screen is as follows:

Column Head	Description
Total	The Total column displays the total number of errors detected for the device.
Cyl	The Cyl (Cylinder) column displays the cylinder number (in decimal form) within the disk where an error occurred.
Surf	The Surf (Surface) column displays the surface number (in decimal form) within the cylinder where an error occurred.
Block	The Block column displays the block number within the surface number (in decimal form) where an error occurred.
Volume	The Volume column displays the name (up to six characters) of the disk where an error occurred.
Command	The Command column refers to the disk command, translated from the I/O Control Word (IOCW), that the device was executing at the time that an error occurred.
Status	The Status column displays the type of error detected: hard if the error is nonrecoverable, soft if it is recoverable.
I/O Status Word (IOSW)	The I/O Status Word column displays the hexadecimal representation of the status of an I/O command word (IOCW) that the operating system issues to a device.

15.4.4 The IOELOG Translated Device Unit IOSW Summary

To translate an IOSW for a particular error type, position the cursor on the line that displays the error and press ENTER. The resulting screen provides you with an analysis of the error conditions and retry indicators given in the IOSW. A sample IOELOG Translated Disk IOSW screen is shown in Figure 15-8. The range of the error log, the disk unit number, and the device type are displayed at the top of the screen.

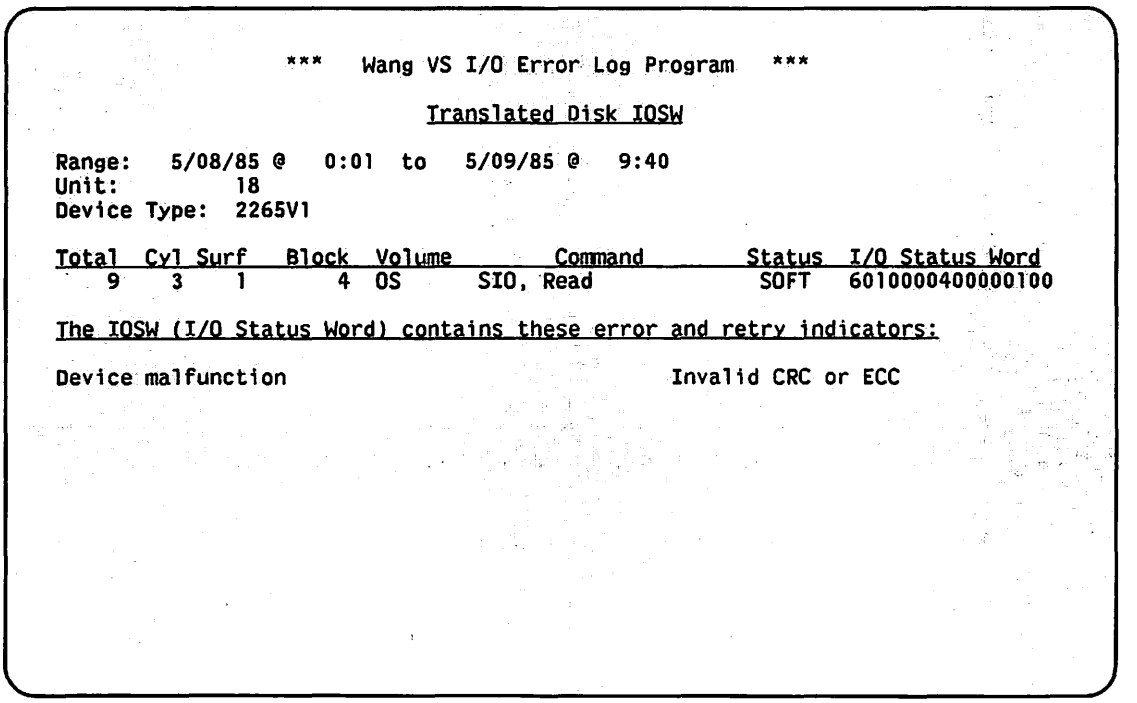


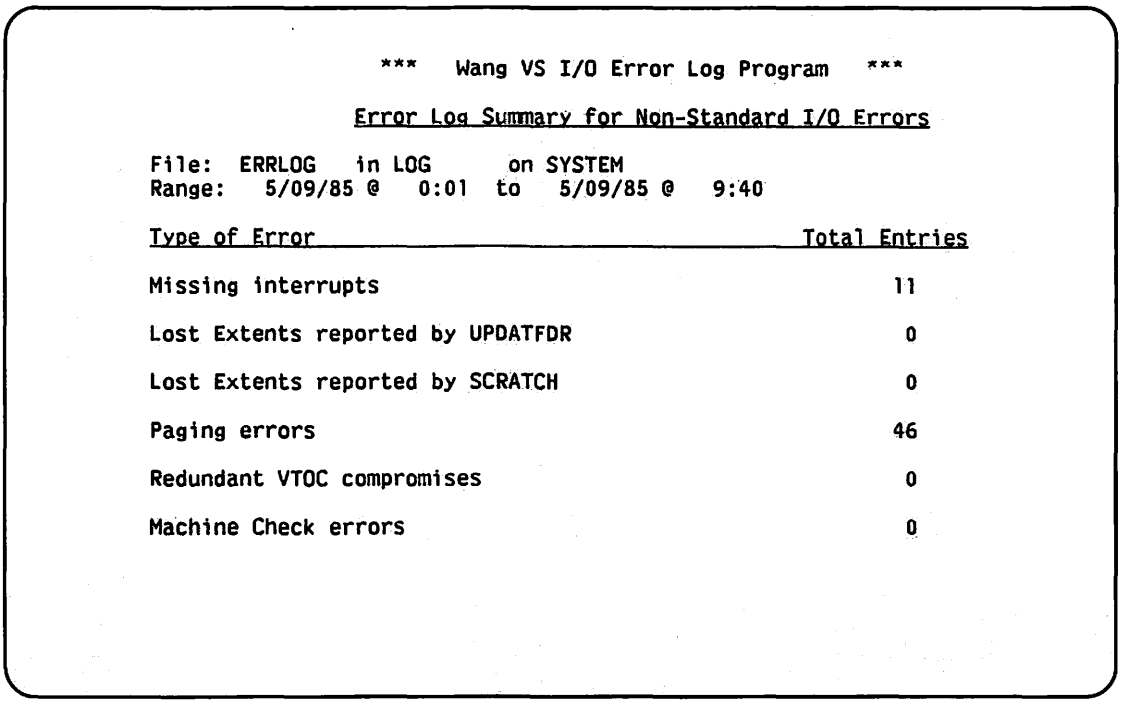
Figure 15-8. Sample IOELOG Translated Disk IOSW Screen

The total number of errors that occurred is given for the IOSW error type that you selected to examine. As in the disk unit summary, the block, cylinder, surface, and volume name on which an error occurred (if any) are listed, as well as the instruction type, error status, and IOSW in hexadecimal form. A breakdown of the error conditions and retry indicators is also given. The Sample Translated Disk IOSW screen provides you with the following functions:

PF Key	Function
1	Return to the Disk Unit Summary screen
13	Obtain on-line help information
15	Print the contents of the screen

15.5 THE NONSTANDARD I/O ERROR SUMMARY

You can analyze nonstandard I/O errors by pressing PF2 from the Standard I/O Error Summary screen (Figure 15-5). The Nonstandard I/O Error Summary screen (Figure 15-9) is then displayed containing any errors recorded in the log that are not logged against one of the five device classes. The log file name and location are displayed at the top of the screen, as well as the date and time range of the error log that you are analyzing.



```
                    ***   Wang VS I/O Error Log Program   ***

                  Error Log Summary for Non-Standard I/O Errors

File:  ERRLOG   in LOG      on SYSTEM
Range:   5/09/85 @   0:01  to   5/09/85 @   9:40

Type of Error                                          Total Entries

Missing interrupts                                            11

Lost Extents reported by UPDATFDR                              0

Lost Extents reported by SCRATCH                               0

Paging errors                                                 46

Redundant VTOC compromises                                     0

Machine Check errors                                           0
```

Figure 15-9. Sample Nonstandard I/O Error Summary Screen

A description of the types of errors included in the Nonstandard I/O Error Summary screen are as follows:

Error	Description
Missing Interrupts	A device took too long to respond to a command.
Lost Extents Reported by UPDATFDR	The UPDATFDR Supervisory Call (SVC) cannot return an extent to the free extent list in the VTOC. This message usually does not indicate a disk error, but rather a problem with space in the VTOC.
Lost Extents Reported by SCRATCH	The SCRATCH SVC cannot return an extent to the free extent list in the VTOC. This message usually does not indicate a disk error, but rather a problem with space in the VTOC.
Paging Errors	An I/O instruction to the paging file failed due to a hard disk error. A paging file is a disk image of a task's memory.
Redundant VTOC Compromises	An I/O instruction to the VTOC failed due to a hard disk error.
Machine Check Errors	A memory parity error or an IOP error occurred. A memory parity error occurs when data is transmitted incorrectly. If an error that involves one bit occurs, the microcode detects the error and makes corrections. If the error involves two or more bits, the microcode detects the error but cannot correct it. IOELOG reports memory parity errors that involve two or more bits. An IOP error occurs when an IOP cannot move data into or out of the devices needed to process the data.

You can perform the following functions from the Nonstandard I/O Error Summary screen by pressing the corresponding PF key:

PF Key	Function	Description
ENTER	Display Error Summary	Press ENTER after you position the cursor next to any nonstandard error to continue processing, and to display an error summary that lists the total number of standard I/O errors that occurred for a particular device class.
1	Return	Return to the Standard I/O Error Summary screen (refer to Figure 15-5).
13	Information	Display on-line help information.
15	Print	Print the contents of the screen.

You can analyze each nonstandard error further by positioning the cursor next to an error and pressing ENTER. An error summary of the selected nonstandard error is displayed. Figure 15-10 illustrates a sample Missing Interrupts Summary screen. The Missing Interrupts Summary screen provides you with the error type and the device on which the error occurred. The date and time range of the error log appear at the top of the screen.

```
                    ***  Wang VS I/O Error Log Program  ***

                                Missing Interrupts

Range:   from  5/08/85 @  0:01 am  to  5/09/85  @  9:40  am

          Type                    Unit    Class      Device      Type       Total

ICP now ready lost                17      Disk       2265V-2     Rem          2
                                  19      Disk       2265V-2     Rem          1
                                  23      Disk       2265V-2     Rem          2
                                  40      Printer    2281V-S     Daisy        2
Unsolicited pending lost          18      Disk       2265V-2     Rem          1
                                  19      Disk       2265V-2     Rem          1
                                  23      Printer    2265V-2     Rem          1
                                  41      Disk       2281V-S     Daisy        3
I/O completion lost               17      Disk       2265V-2     Rem          3
                                  45      Printer    2281V-S     Daisy        1


 Press the appropriate PF Key to perform the desired action:
 (1) Return (2) First (3) Last (4) Prev (5) Next    (13) Information (15) Print
```

Figure 15-10. Sample Missing Interrupts Summary Screen

A description of the Missing Interrupts Error Summary screen fields is as follows:

Column Head	Description
Type	The Type column (the first of two) displays the type of missing interrupt that occurred.
Unit	The Unit column displays the unit number of the device.
Device	The Device column displays the device name recognized by the system.
Type	The Type column (the second of two) displays the type of device on which the error occurred (i.e, Rem, Fixed, or Fix/Rem disk; Daisy or Chain printer, etc.)
Total	The Total column displays the total number of missing interrupts recorded by the log for a given device unit.

The Missing Interrupts Error Summary screen also provides the following functions:

PF Key	Function	Description
1	Return	Return to the Nonstandard I/O Error Summary screen (Figure 15-9).
2	First	Display the first screen of missing interrupt errors (Figure 15-10). This option is available only if more than one screen is required to display the error list.
3	Last	Display the last screen of missing interrupt errors (Figure 15-10). This option is available only if more than one screen is required to display the error list.
4	Prev	Display the missing interrupt errors on the previous screen (Figure 15-10). This option is available only if more than one screen is required to display the error list.
5	Next	Display the missing interrupt errors on the next screen (Figure 15-10). This option is available only if more than one screen is required to display the error list.
13	Information	Display on-line help information.
15	Print	Print the screen contents.

15.6 THE IPL SUMMARY

To display the IOELOG IPL Summary screen (Figure 15-11), press PF3 from the Standard I/O Error Summary screen (Figure 15-5). The IOELOG IPL Summary screen displays (in a left to right sequence) the date and time that each IPL occurred.The time and date range of the error log being analyzed appears at the beginning of the summary.

NOTE
If there were no IPLs within the time frame that you specified, the IPLs option (PF3) is not displayed.

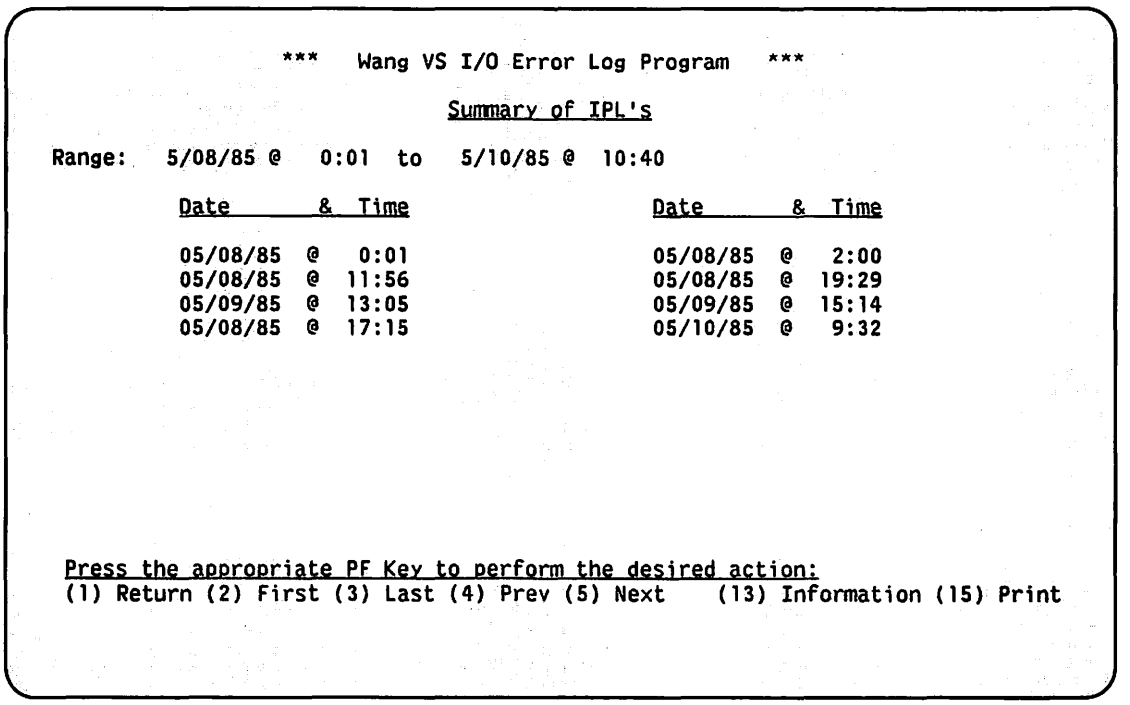


Figure 15-11. Sample IOELOG IPL Summary Screen

The functions on the IOELOG IPL Summary screen are as follows:

PF Key	Function	Description
1	Return	Return to the Standard I/O Error Summary screen (Figure 15-5).
2	First	Display the first screen of IPLs (Figure 15-11). This option is available only if more than one screen is required to display the IPL list.
3	Last	Display the last screen of IPLs (Figure 15-11). This option is available only if more than one screen is required to display the IPL list.
4	Prev	Display the IPLs on the previous screen. This option is available only if more than one screen is required to display the IPL list.
5	Next	Display the IPLs on the next screen (Figure 15-11). This option is available only if more than one screen is required to display the IPL list.
13	Information	Display on-line help information.
15	Print	Print the screen contents.

15.7 PRINTING AN I/O ERROR LOG FILE

To print the contents of an I/O error log file, press PF2 from the IOELOG Function Definition screen (Figure 15-3). When you press PF2, the IOELOG Range Definition screen (Figure 15-4) appears. You can obtain a portion of the error log by modifying the Startday, Starttime, Endday, and Endtime fields. Specify the time range for the printout. The printout lists the I/O errors in the specified time range. When you press ENTER, the error log is placed into a print file which can be sent to the print queue immediately or held until released for printing (depending on the value in your PRNTMODE field, which is set through the SET Usage Constants function (PF2) on the Command Processor menu).

An I/O error log printout contains the following information:

- The date and time that an error is recorded by the log
- The device(s) affected by the error
- The command issued to the device by the operating system
- The type of error that occurred as a result of that command

If a disk is affected by an I/O error, the location of the error on the disk is listed. The date and time range of the error log appears at the top of each page on the printout.

NOTE
An error occurred in the VTOC if a volume name is listed in the printout but file and library names are not.

When you print an error log, the log is formatted and put into a print file. The system assigns the name ELOG followed by four digits (in the form ELOGxxxx). The file is stored in your default SPOOLIB and SPOOLVOL which is set through the SET Usage Constants function (PF2) of the Command Processor menu or through a procedure. If you do not set any defaults, the print file is stored in the System Print Library (#PRT) on the System Volume.

When you press PF2 (Print) from the IOELOG Function Definition screen (Figure 15-3), the IOELOG Range Definition screen (Figure 15-4) is displayed. The log can be printed in its entirety or a certain portion can be specified. Refer to Section 15.4 to review the options from the IOELOG Range Definition screen. A sample partial I/O error log printout is shown in Figure 15-12. A description of the information that is included in an I/O error log printout follows Figure 15-12.

```
***  Wang VS I/O Error & IPL Log  ***     Covering the time period from    7/02/85 at 11:29 to  7/08/85 at  8:50

                    User WS Task Unit#  IOCW(I/O Command Word) IOSW(I/O Status Word)          Retry          (For Disks Only)
Date      Time      Device Class & Type  Command Description    Error Description             #  Volume Library  File      Cyl Sur Block

7/02/85 11:42         0  36    18   41 07E800 0800 026138 60100004 00000100                  1 OS     @SYSPAGE @S22C088 433   1       1
                           2265V1  CIO, Unknown IOCW      Pagein error
                                                          Soft error
                                                          Device malfunction
                                                          Invalid CRC or EEC

7/02/85 11:52         0  36    18   41 073000 0800 03AEC8 60100004 00000200                  2 OS     @SYSPAGE @S201D10 670   2       1
                           2265V1  CIO, Unknown IOCW      Pagein error
                                                          Soft error
                                                          Device malfunction
                                                          Invalid CRC or EEC

7/02/85 12:01         0  36    18   41 06C800 0800 024C50 60100004 00000100                  1 OS     @SYSPAGE @S22AE30 418   1       7
                           2265V1  CIO, Unknown IOCW      Pagein error
                                                          Soft error
                                                          Device malfunction
                                                          Invalid CRC or EEC

7/02/85 12:04       ---   1     8   A0 01A000 0800 000000 20190001 08000000
              WS           2256C   CIO, Load Microcode    Hard Error
                                                          Device malfunction
                                                          Device or memory damage
                                                          No device processor code
                                                          Device powered off
                                                          Device processor not running

7/02/85 12:18       ---   1    43   A0 063800 0800 000000 20190001 08000000
              WS           2266C   CIO, Load Microcode    Hard Error
                                                          Device malfunction
                                                          Device or memory damage
                                                          No device processor code
                                                          Device powered off
                                                          Device processor not running

7/02/85 12:59         0  18    18   41 028000 0800 03AE68 60100004 00000100                  1 OS     @SYSPAGE @S201D10 670   0       7
                           2265V1  SIO, Read              Pagein error
                                                          Soft error
                                                          Device malfunction
                                                          Invalid CRC or EEC
```

Figure 15-12. Sample Partial I/O Error Log Printout

Column Head	Description
Date	The Date column shows the date of an entry into the log.
Time	The Time column shows the time of an entry into the log.
User	The User column shows the user ID.
WS	The WS (Workstation) column shows the workstation number that is associated with the user ID.
Task	The Task column shows the task number that is associated with the running program.
Unit #	The Unit # column shows the unit number of the device on which an error occurred.
Device Class & Type	The Device Class & Type column shows the device classification (e.g., disk) and device type.
IOCW Command Description	The IOCW column shows the type of command that is issued by the operating system to the device. A hexadecimal representation of the IOCW is also given.
IOSW Error Description	The IOSW Error Description column shows the status of the command previously issued by the operating system to the device. A hexadecimal representation of the IOSW is also given.
Retry #	The Retry # column shows the number of times a command is issued to a device by the operating system before it is successful (a soft error).

The following additional information appears when a disk error occurs:

Column Head	Description
Volume	The Volume column shows the name of the volume that is affected by an error.
Library	The Library column shows the name of the library that is affected by an error.
File	The File column shows the name of the file that is affected by an error.
Cyl	The Cyl (Cylinder) column shows the cylinder on which an error occurred (if any).
Sur	The Sur (Surface) column shows the surface on which an error occurred (if any).
Block	The Block column shows the block on which an error occurred (if any).

CHAPTER 16
THE IOTRACE UTILITY

16.1 INTRODUCTION

The IOTRACE utility helps you to diagnose device input/output (I/O) problems. IOTRACE enables you to selectively capture a history of the I/Os issued to one or more devices for later analysis. It monitors the I/O trace table and records I/O information for a range of devices, based on the criteria that you define. You can retain captured data in a small disk file that can be collected and analyzed at your convenience. IOTRACE has the following features:

- The IOTRACE Function Selection screen — Enables you to run the utility as a foreground or a background task, or to print previously traced data.
- The Print option — Enables you to format IOTRACE data for a printed report or for display at the workstation.
- Trap setting — Used to specify which type(s) of I/O operation is recorded, for analysis, into the IOTRACE output file. When a trap is set, only those types of I/O operations specified are recorded, eliminating from the output file the I/O operations that are not needed for analysis. You can set multiple traps to include specific types of I/O. The I/O operation types for which traps are set are as follows:
  - IOSW (I/O Status Word)
  - SIO (Start I/O)
  - CIO (Control I/O)
  - HIO (Halt I/O)
- The Time option — Records all I/O operations during a specified time period and ends IOTRACE processing after a specified amount of time has elapsed.
- Multiple IOTRACE processing — More than one user can run IOTRACE without duplicating output file names, since each user's logon ID is overlaid onto the output file name for any specific IOTRACE run.
- IOTRACE data storage — After processing, IOTRACE stores the collected data in the library, @IOTRACE, for you to analyze at your convenience.

Figure 16-1 provides an overview of IOTRACE processing.

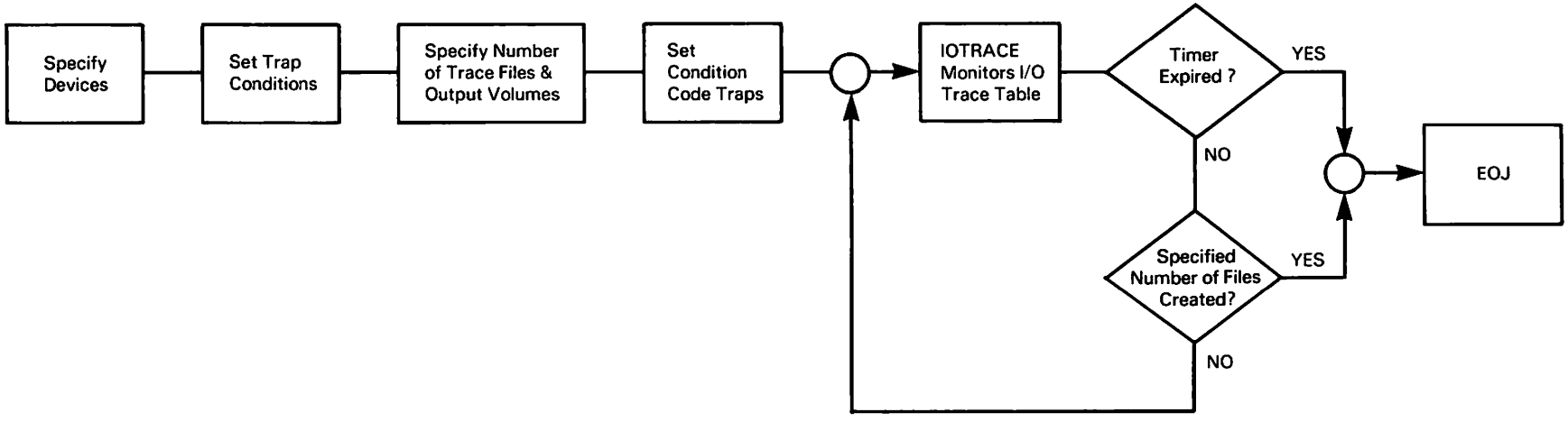


Figure 16-1. IOTRACE Processing

16.2 RUNNING IOTRACE

When you run the IOTRACE utility, the IOTRACE Function Selection screen (Figure 16-2) enables you to choose a trace or print operation. Indicate your choice by pressing one of the following PF keys:

PF Key	Function	Description
1	Trace	Trace I/O events with IOTRACE running as a foreground task from your workstation.
2	Trace from Background	Trace I/O events with IOTRACE running as a background (batch) task. Running IOTRACE as a background task minimizes the impact on system resources.
3	Print	Format and print previously traced I/O events or display formatted data at the workstation.
16	End Job	Cancel processing of IOTRACE.

```
*** MESSAGE 0001 BY TRACE

          INFORMATION REQUIRED BY PROGRAM IOTRACE
                    TO DEFINE SELECT
               ACTIVE SUBPROGRAM IS IOTRACE

        *** WANG VS I/O TRACE UTILITY - VERSION x.xx.xx ***

      PFK 1  = START TO TRACE I/O EVENTS

      PFK 2  = TRACE I/O EVENTS FROM BACKGROUND

      PFK 3  = PRINT PREVIOUSLY TRACED I/O EVENTS



      PFK 16 = END JOB
```

Figure 16-2. IOTRACE Function Selection Screen

16.2.1 Input Definition

If you select PF1 or PF2 from the IOTRACE Function Selection screen, the IOTRACE Traps Definition screen (Figure 16-3) prompts you to enter the range of devices that you want to trace. You can accept the default values shown on the screen or enter another range in the LOWDEV and HIGHDEV fields.

```
***  MESSAGE 0001 BY TRACE

                INFORMATION REQUIRED BY  PROGRAM  IOTRACE
                          TO DEFINE INPUT
                     ACTIVE SUBPROGRAM IS IOTRACE

             ***  WANG VS I/O TRACE UTILITY - VERSION  x.xx.xx  ***

Please specify the following information for device I/O tracing:

Range of devices to be traced LOWDEV   = 000   HIGHDEV  = 255

Set at least one trap condition to close the trace file.

   IOSWTRAP = XXXXXXXXXXXXXXXX   SIOTRAP  = XXXXXXXXXXXXXXXX X = don't care
   CIOTRAP  = XXXXXXXXXXXXXXXX   HIOTRAP  = XXXXXXXXXXXXXXXX

Output volume name for trace file VOLUME   = SYSTEM

Desired number of trace files     TRAPREQ  = 001

Desired time to run trace (HH:MM, 24 HR MAX)  TIME     = 0000

            Press (ENTER) to continue, (13) for HELP, (16) to EXIT
```

Figure 16-3. Sample IOTRACE Traps Definition Screen

NOTE
If you press PF13, an on-line, brief description of each of the input fields is shown on the IOTRACE Input Definition screen. The Help Information screen is then displayed. To return to the input screen from the Help Information screen, press ENTER.

The Time field, located at the bottom of the IOTRACE Traps Definition screen, enables you to specify the amount of time for a trace session. If you set the timer, you do not need to set any other traps. The timer traps (and records from the I/O trace table) all I/O operations over the specified amount of time. IOTRACE processing ends when the timer expires or when IOTRACE has created the specified number of trace files, whichever event occurs first.

The timer option is especially useful when you do not know which trap to use for device I/O analysis. The timer is checked only after each 100 trapped I/Os are written to disk; therefore, a session can last longer than expected if you are tracing only one device or if system activity is low.

NOTE
If you do not enter a value in the Time field on the IOTRACE Traps Definition screen (Figure 16-3), you must specify at least one trap condition on either the IOTRACE Traps Definition screen or the Condition Code Traps screen (Figure 16-4).

Defining Condition Code Traps

If traps are specified, the Condition Code Traps screen (Figure 16-4) appears. This screen gives you the option to specify additional trap conditions.

Each I/O Command Word (IOCW) entry in the trace table contains a condition code. The condition code is returned by the instruction that used the I/O. The condition, stored in the IOCW element of the trace table, is the condition code from the SIO, CIO, and HIO instructions. For more information about these instructions, refer to the *VS Principles of Operation*.

To specify additional trap(s) for one or more of the conditions, specify Y (yes) in the appropriate spaces on the Condition Code Traps screen. When you are satisfied with the entries to the Condition Code Traps screen, press ENTER to initiate IOTRACE processing.

For an explanation of condition code traps and status or control words, refer to the *VS Principles of Operation*.

16.3 IOTRACE PROCESSING

If you press ENTER from the Condition Code Traps screen (Figure 16-4), IOTRACE processing begins as either a foreground or a background task, depending on the function that you selected from the IOTRACE Function Selection screen (Figure 16-2). When processing is complete, IOTRACE assigns a file name to the IOTRACE output.

```
***  MESSAGE 0020 BY TRACE

                INFORMATION REQUIRED BY  PROGRAM  IOTRACE
                          TO DEFINE CCTRAPS
                     ACTIVE SUBPROGRAM IS IOTRACE


Each IOCW entry in the trace table contains the condition code returned
   by the instruction which issued the I/O.

Enter (Y) for any condition code to be trapped.

The trace file closes when the trap condition occurs.

  CONDITION TYPE        START I/O       CONTROL I/O       HALT I/O
  DEVICE BUSY           SIODB    = N    CIODB    = N      HIODB    = N
  IOP BUSY              SIOIB    = N    CIOIB    = N      HIOIB    = N
  IOP NOT OPERATIONAL   SIOIN    = N    CIOIN    = N      HIOIN    = N

                Press (ENTER) to continue, (1) to RETURN
```

Figure 16-4. Condition Code Traps Screen

The IOTRACE output file name is taken from the day and time that processing ended, using a day:hour:minutes:seconds format (DDHHMMSS). For example, if an IOTRACE process completed on the 14th of the month at 3:27:00 pm, the output file would be named 14152700 (15 is used for 3 pm on a 24-hour clock).

IOTRACE takes another precautionary measure to exclude the remote possibility that two IOTRACE runs completed at the same time: the user ID is overlaid onto the first three spaces of the IOTRACE output file. For example, JRR52700 would be the output file for user ID JRR. (The first three digits (141) of the output file (14152700) is overlaid by the user ID (JRR).) User IDs with characters not recognized by the IOTRACE procedure (i.e., the blank space(s) on user IDs with less than three characters) will be ignored and not carried over when the ID is overlaid onto the output file name.

When either foreground or background processing is complete, the IOTRACE output file is stored in the @IOTRACE library for later analysis. The workstation returns to the Command Processor menu upon completion of an IOTRACE run and the utility must be rerun to print the file report.

16.3.1 Foreground Task

If you selected PF1 (START TO TRACE I/O EVENTS) from the IOTRACE Function Selection screen (Figure 16-2), IOTRACE begins to monitor the I/O trace table, as a foreground (interactive) task. It also tests the entries in the table against the criteria that you specified. IOTRACE tests the trace table entries in the following order:

1. The device for any given I/O must be within the specified device range.
2. If you specified a trap value, IOTRACE compares the trap value to the IOSW or IOCW of the trace table. A match triggers the trap and IOTRACE copies the I/O operation into the output file.
3. If you specified a condition code, IOTRACE compares that code against the condition code of the trace table entry. A match triggers the trap and IOTRACE copies the I/O operation into the output file.

When the trap occurs, IOTRACE stops testing and copies the next 50 to 100 I/O operations from all devices in the system trace table. This ensures that any problem triggered by the trap is recorded.

During processing, IOTRACE displays the message "Program IOTRACE in progress" on your workstation screen. The workstation returns to the Command Processor menu upon completion of an IOTRACE run and the utility must be rerun to print the file report.

16.3.2 Background Task

If you selected PF2 (TRACE I/O EVENTS FROM BACKGROUND) from the IOTRACE Function Selection screen (Figure 16-2), the system uses the information from the IOTRACE Input Definition screen and the Condition Code Traps screen to generate a procedure named xxxTRACE, (where xxx is your logon ID). The system automatically submits xxxTRACE to the background task queue for processing.

During processing, IOTRACE returns to the IOTRACE Function Selection screen (Figure 16-2) and you are free to use your workstation. When processing is completed, a completion message is displayed on your terminal.

16.4 IOTRACE OUTPUT

When you press PF3 (PRINT PREVIOUSLY TRACED I/O EVENTS) from the IOTRACE Function Selection screen, the system displays the IOTRACE Print File Definition screen (Figure 16-5). This screen enables you to format and print the previously traced I/O events or to display formatted trace data at your workstation. Following Figure 16-5 is a description of the fields on the IOTRACE Print File Definition Screen.

```
***  MESSAGE 0000 BY IOPRNT

                INFORMATION REQUIRED BY  PROGRAM  IOTRACE
                          TO DEFINE INPUT
                     ACTIVE SUBPROGRAM IS IOTRACE

VS IO TRACE DATA FILE PRINT PROGRAM VERSION  x.xx.xx

                ENTER NAME OF IO TRACE FILE TO BE PRINTED

     FILE     = ********      LIBRARY  = @IOTRACE      VOLUME   = SYSTEM


 TITLE    = ************************************************************

     TYPE OF  OUTPUT   = PRINTER  PRINTER OR SCREEN
     LINES PER PAGE OF OUTPUT  LINES    = 55

     SPECIFY REPORT TYPE     = S  S = SUMMARY PAGE ONLY
                                  B = BRIEF, SUMMARY AND 50 ENTRIES
                                  D = DETAIL, SUMMARY AND ALL ENTRIES

                           PRESS PFK 16 TO END JOB
```

Figure 16-5. Sample IOTRACE Print File Definition Screen

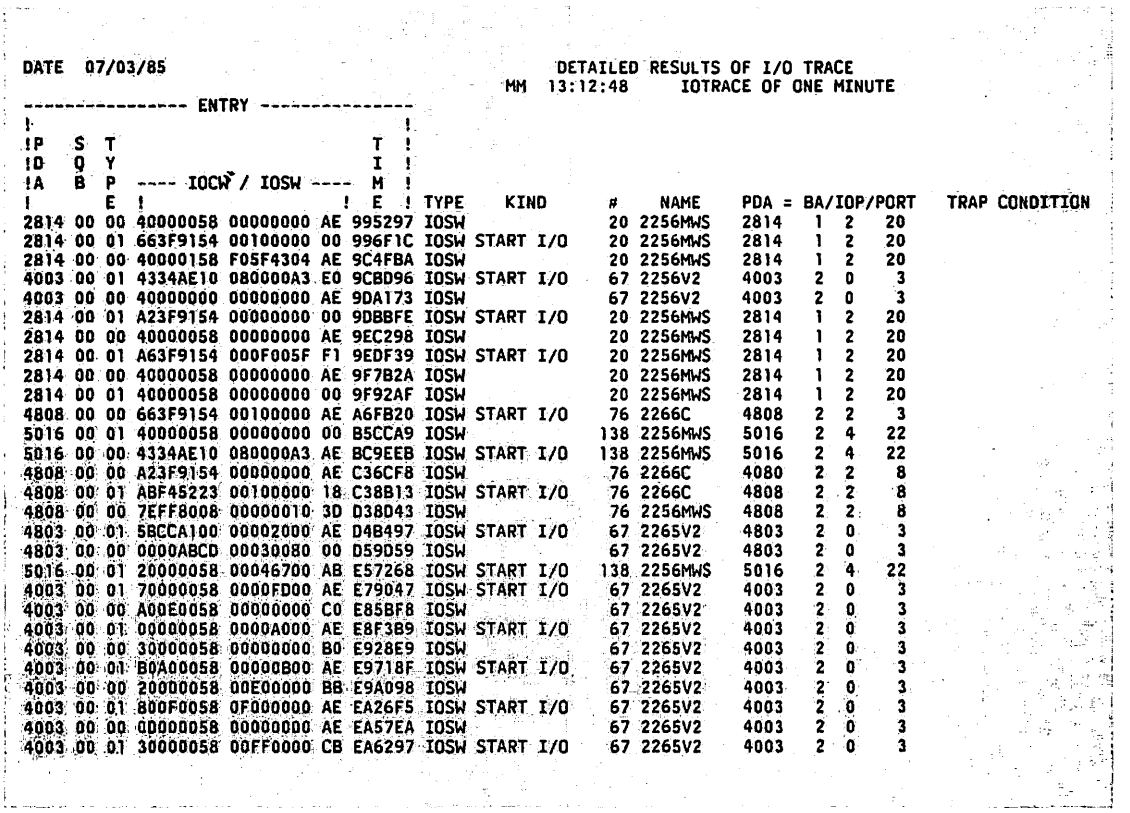
Field	Description
FILE	Specify the name of the trace file to be formatted. The default value is the name of the most recent trace file in the @IOTRACE library on the system volume. If there is more than one file in the @IOTRACE library, you can obtain a list of additional files in the library by pressing PF5. When you press the PF5 key, position the cursor under the block graphic that is displayed next to the file that you want and press ENTER. This places the specified file name in the File field on the Print Input Information screen.
LIBRARY	Specify the name of the library that contains the trace data files. The default value is @IOTRACE.
VOLUME	Specify the name of the volume on which the trace data files are located. The default value is the current system volume.
TITLE	Create a page title of up to 60 characters in length. The title is centered at the top of each page of the output report.
TYPE OF OUTPUT	Specify the output device (PRINTER or SCREEN) for the finished report. If you select PRINTER, the program formats and prints previously traced I/O data. If you select SCREEN, the program links to the DISPLAY utility to display the print file at the workstation. The print file is not saved. To view the print file at the workstation, the system library (or your program library) must contain the DISPLAY utility.

Field	Description
LINES	Specify the number of lines per page for the finished report.
SPECIFY REPORT TYPE	Specify the type of detail to be contained in the report. Enter one of the following values: S (Summary) — A 1-page summary of the trace session (refer to Figure 16-6). B (Brief) — The summary page and 50 detail lines from the trace file; the element that caused the trap (or on which the timer expired) is centered on the listing. (Refer to Figure 16-7.) D (Detail) — The summary page and all detail lines in the trace file. The data format is the same for both brief and detail listings. (Refer to Figure 16-7.)

After printing or displaying the formatted trace data, press PF16 and the program returns to the Print File Definition screen (Figure 16-5). At this point, you can format another file or press PF16 to return to the IOTRACE Function Selection screen (Figure 16-2).

```
DATE 07/03/85                  RESULTS OF IOTRACE                    PAGE    1

THIS IS A TRACE FROM A VS CPU TYPE 4  TAKEN ON 07/13/85 AT  10:51:21:54
THE TRACE FILE NAME = NLI05207 LIBRARY = @IOTRACE VOLUME = SYS620
SYSTEM OS VERSION  = xx.xx.xx    IOTRACE VERSION = xx.xx.xx

SYSTEM I/O TRACE RANGE   = 0000 THRU 7FFF   SYSTEM HAS 150 DEVICES TRACED
USER DEFINED TRACE RANGE = 0000 THRU 4C05

           TRAP MASKS AND CONDITION CODE TESTS DEFINED BY USER
                                DEVICE BUSY    IOP BUSY    IOP NOT OPER.
SIO IOCW XXXXXXXX XXXXXXXX          Y            N             N
CIO IOCW XXXXXXXX XXXXXXXX          N            N             N
HIO IOCW XXXXXXXX XXXXXXXX          N            N             N
    IOSW XXXXXXXX XXXXXXXX

TRAP OCCURED ON 02/13/85 AT  10:52:06:15
TRACE TABLE ENTRY WHICH CAUSED TRAP = 40000000 40000000 00000000 C4641B7C
DEVICE # =  120  PDA = 4000  BUS ADAPTOR = 2  IOP # = 0  PORT =   0
DEVICE NAME = 75  Meg Rem Disk
REASON FOR TRAP = TRACE TIMER EXPIRED
```

Figure 16-6. Sample Trace Summary Page



```
DATE  07/03/85                                    DETAILED RESULTS OF I/O TRACE
                                                MM  13:12:48     IOTRACE OF ONE MINUTE

---------------- ENTRY ----------------
!                                      !
!P   S  T                          T   !
!D   Q  Y                          I   !
!A   B  P  ---- IOCW / IOSW ----   M   !
!       E  !                    !  E   ! TYPE    KIND       #    NAME     PDA = BA/IOP/PORT   TRAP CONDITION
2814 00 00 40000058 00000000 AE 995297 IOSW                20 2256MWS  2814   1  2   20
2814 00 01 663F9154 00100000 00 996F1C IOSW START I/O      20 2256MWS  2814   1  2   20
2814 00 00 40000158 F05F4304 AE 9C4FBA IOSW                20 2256MWS  2814   1  2   20
4003 00 01 4334AE10 080000A3 E0 9CBD96 IOSW START I/O      67 2256V2   4003   2  0    3
4003 00 00 40000000 00000000 AE 9DA173 IOSW                67 2256V2   4003   2  0    3
2814 00 01 A23F9154 00000000 00 9DBBFE IOSW START I/O      20 2256MWS  2814   1  2   20
2814 00 00 40000058 00000000 AE 9EC298 IOSW                20 2256MWS  2814   1  2   20
2814 00 01 A63F9154 000F005F F1 9EDF39 IOSW START I/O      20 2256MWS  2814   1  2   20
2814 00 00 40000058 00000000 AE 9F7B2A IOSW                20 2256MWS  2814   1  2   20
2814 00 01 40000058 00000000 00 9F92AF IOSW                20 2256MWS  2814   1  2   20
4808 00 00 663F9154 00100000 AE A6FB20 IOSW START I/O      76 2266C    4808   2  2    3
5016 00 01 40000058 00000000 00 B5CCA9 IOSW               138 2256MWS  5016   2  4   22
5016 00 00 4334AE10 080000A3 AE BC9EEB IOSW START I/O     138 2256MWS  5016   2  4   22
4808 00 00 A23F9154 00000000 AE C36CF8 IOSW                76 2266C    4080   2  2    8
4808 00 01 ABF45223 00100000 18 C38B13 IOSW START I/O      76 2266C    4808   2  2    8
4808 00 00 7EFF8008 00000010 3D D38D43 IOSW                76 2256MWS  4808   2  2    8
4803 00 01 5BCCA100 00002000 AE D4B497 IOSW START I/O      67 2265V2   4803   2  0    3
4803 00 00 0000ABCD 00030080 00 D59D59 IOSW                67 2265V2   4803   2  0    3
5016 00 01 20000058 00046700 AB E57268 IOSW START I/O     138 2256MWS  5016   2  4   22
4003 00 01 70000058 0000FD00 AE E79047 IOSW START I/O      67 2265V2   4003   2  0    3
4003 00 00 A00E0058 00000000 C0 E85BF8 IOSW                67 2265V2   4003   2  0    3
4003 00 01 0000005B 0000A000 AE E8F3B9 IOSW START I/O      67 2265V2   4003   2  0    3
4003 00 00 30000058 00000000 B0 E928E9 IOSW                67 2265V2   4003   2  0    3
4003 00 01 B0A00058 00000B00 AE E9718F IOSW START I/O      67 2265V2   4003   2  0    3
4003 00 00 20000058 00E00000 BB E9A098 IOSW                67 2265V2   4003   2  0    3
4003 00 01 800F0058 0F000000 AE EA26F5 IOSW START I/O      67 2265V2   4003   2  0    3
4003 00 00 0D000058 00000000 AE EA57EA IOSW                67 2265V2   4003   2  0    3
4003 00 01 30000058 00FF0000 CB EA6297 IOSW START I/O      67 2265V2   4003   2  0    3
```

Figure 16-7. Sample Trace Listing Format

16.5 A SAMPLE IOTRACE PROCEDURE

You can control IOTRACE processing through the VS Procedure language. Appendix A provides a complete list of IOTRACE GETPARMs. For detailed information about the syntax of the VS Procedure language, refer to the *VS Procedure Language Reference*.

The following procedure monitors the I/O trace table for a single device (LOWDEV and HIGHDEV have the same value). The procedure sets a trap for the completion with error of an I/O operation to the device. The procedure requests the creation of a maximum of three trace files and designates ZENITH as the output volume.

```
PROCEDURE
    RUN IOTRACE
        ENTER INPUT LOWDEV = 020, HIGHDEV = 020,
            IOSWTRAP = 2XXXXXXXXXXXXXXX, VOLUME = ZENITH,
              TRAPREQ = 3
        ENTER CCTRAPS
RETURN
```

CHAPTER 17
THE LISTVTOC UTILITY

17.1 INTRODUCTION

The LISTVTOC utility performs a verification check on the Volume Table of Contents (VTOC) of a specified disk volume or member of a volume set to determine whether or not the VTOC contains any bad pointers or count fields. The resulting analysis consists of three listings:

- Space allocation for the volume
- File names and attributes, by library (excluding secondary volumes of a volume set)
- A VTOC map, which is a dump of the VTOC control blocks

This utility also provides an analysis of the contents of a VTOC. A VTOC is structured in the following way:

- The first block of the VTOC is an available space block. This block contains pointers to the first and last blocks of free extents on the volume. Additional File Directory Available space (FDAV) blocks may be allocated. When searching for space on the volume to store a new file, the operating system searches this block first for a sufficiently large area of space. If sufficient space is not available in the first block, the operating system subsequently follows the chain of free extent pointers in subsequent FDAV blocks until sufficient space is found.
- The second block of the VTOC is a first-level index block (FDX1). A first-level index block is an index of libraries on the volume and may be chained to additional first-level index blocks. It also contains a pointer to the second-level index block as well as pointers to free blocks within the VTOC and pointers to FDX2 and FDR blocks with free records. However, subsequent FDX1 blocks do not contain this additional information.
- The third block of the VTOC is a second-level index block. A second-level index block is an index of files within a library and may be chained to additional second-level index blocks.
- The fourth block of the VTOC contains File Descriptor Records (FDRs) which describe files.
- The remaining used blocks of the VTOC are used for chaining available space blocks, index blocks, and file descriptor blocks.

To obtain more information about VTOC structures and the READVTOC SVC (supervisory call), refer to the *VS Operating System Services Reference*. Figure 17-1 shows an overview of LISTVTOC processing.

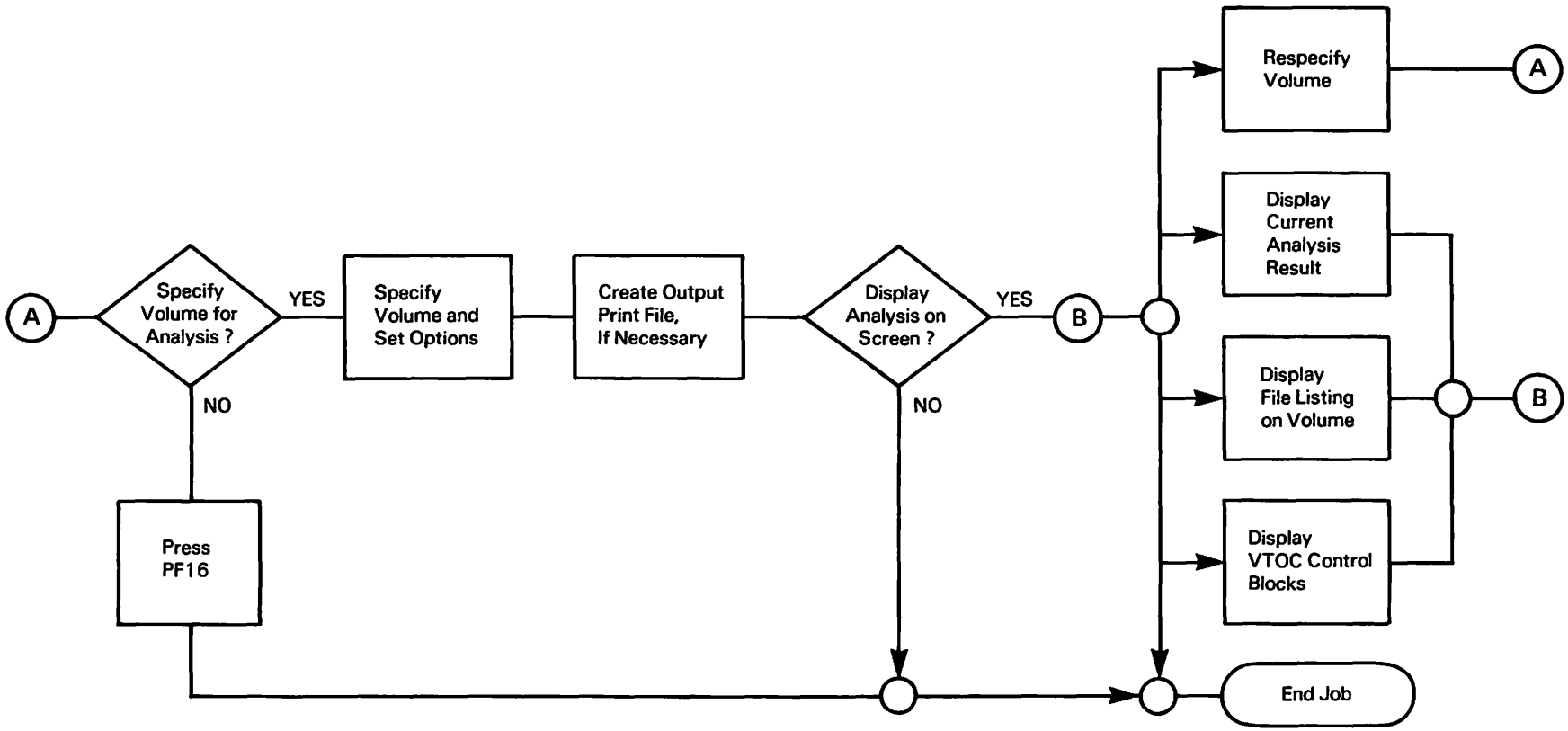


Figure 17-1. LISTVTOC Processing

17.2 DEFINING THE INPUT

When LISTVTOC processing begins, the LISTVTOC Input Definition screen (Figure 17-2) appears. To terminate LISTVTOC processing, press PF16.

```
***  MESSAGE 001  BY WT6V01

                INFORMATION REQUIRED BY  PROGRAM  LISTVTOC
                           TO DEFINE INPUT


                            VERSION x.xx.xx


THE LISTVTOC PROGRAM IS USED TO VERIFY A DISK VOLUME'S VTOC (VOLUME TABLE OF
CONTENTS), AND TO DISPLAY OR PRINT VTOC, FILE, EXTENT, AND VOLUME STATISTICS.

PLEASE SPECIFY THE FOLLOWING INFORMATION AND PRESS (ENTER):

THE VOLUME TO BE PROCESSED IS:        VOLUME   = ******
THE VOLUME ID IS:                     VSID     = 0**      (0 - 255)

PRINT  FILES    = NO*  (YES,NO)    ON LIBRARY  = ******** (BLANK TO PRINT ALL)
PRINT  VTOCMAP  = NO*  (YES,NO)
OUTPUT SCREEN   = YES  (YES,NO)
SELECT PRINT    = 55   LINES PER PAGE

                              Press (16) TO EXIT
```

Figure 17-2. LISTVTOC Input Definition Screen

CAUTION

When you run LISTVTOC, you must set the WORKVOL field in your usage constants to a single volume because work files cannot be processed on a volume set. If the WORKVOL field is set to a volume set member, the system does not allow the creation of a work file and, as a consequence, terminates LISTVTOC processing. (The work file that LISTVTOC creates is a copy of the VTOC. A copy is made so that the system is not tied up while you examine the VTOC.)

To set the WORKVOL field, press PF2 (SET Usage Constants) from the Command Processor menu. Specify a single volume in the WORKVOL field and press ENTER to accept the values that were specified.

The fields on the LISTVTOC Input Definition screen (Figure 17-2) are described as follows:

Field	Description
VOLUME	Specify the volume (or volume set) name on which you want to run LISTVTOC.
VSID	VSID only applies to systems with a volume set. Specify the volume identification number of the volume on the volume set on which you want to run LISTVTOC. The value range is 0 to 8. If the volume is a single volume, accept the default value (0).

Field	Description
PRINT FILES	Specify whether you want a sequentially-ordered file list printout from the specified volume or from a specified library on the volume. File attributes for each file are included in the listing. Values for this field are YES or NO. The default value is NO. If you specify NO, the listing is displayed on the workstation screen only. This option does not apply to secondary volumes of a volume set.
LIBRARY	Specify the name of the library from which you want a file listing. If you leave this entry blank, LISTVTOC generates a listing (by library) of all the volume files.
PRINT VTOCMAP	Specify whether or not you want a dump of the VTOC control blocks. If you specify YES, the VTOC map is placed in a print file. If you specify NO, the VTOC map can only be displayed on the workstation screen. You can review the dump by selecting Display VTOC Control Blocks (PF4) from the VTOC Analysis menu (Figure 17-3). The default value is NO.
OUTPUT SCREEN	Specify whether or not the results of the VTOC analysis are displayed on the workstation screen. If you specify NO, the program ends upon completion of the analysis with no menu displayed. If you specify YES, the VTOC Analysis menu is displayed allowing you to view the results of the analysis, the file listings, and the VTOC map. Refer to Section 17.3 for more information about the VTOC Analysis menu.
SELECT PRINT	Specify the number of lines to appear on each page of the print file. A value is needed only when a print file is requested. The default value is 55.

NOTE
If you enter NO next to all input fields, LISTVTOC creates a print file containing only the VTOC space and integrity analysis.

17.3 THE VTOC ANALYSIS MENU

The VTOC Analysis menu (Figure 17-3) is displayed upon completion of LISTVTOC processing if you specified YES for the Output Screen field on the LISTVTOC Input Definition screen.

```
WANG VS VOLUME TABLE OF CONTENTS ANALYSIS AND PRINT PROGRAM - VERSION  x.xx.xx

VTOC ANALYSIS ON VOLUME OS

PLEASE PRESS ONE OF THE PF KEYS FOR FURTHER ACTION:
       PF1  - TO RESPECIFY VOLUME FOR ANALYSIS
       PF2  - TO DISPLAY CURRENT ANALYSIS RESULT
       PF3  - TO DISPLAY FILE LISTING ON VOLUME
       PF4  - TO DISPLAY VTOC CONTROL BLOCKS
       PF16 - TO END JOB
```

Figure 17-3. VTOC Analysis Menu

The functions that are available from the VTOC Analysis menu are described as follows:

PF Key	Function	Description
PF1	Respecify Volume for Analysis	If you press PF1, LISTVTOC is restarted so that another volume can be analyzed.
PF2	Display Current Analysis Result	This function displays the LISTVTOC Space Allocation screen (Figure 17-4).

The LISTVTOC Space Allocation screen (Figure 17-4) displays a listing (totalled in blocks) which contains the following information:

- The number of free extents on the volume
- The number of libraries and files on the volume
- The number of blocks on the volume
- The number of available blocks on the volume
- The number of allocated blocks on the volume
- The number of blocks in the VTOC

- The number of free blocks in the VTOC
- The number of allocated FDAV control blocks on the volume
- The number of allocated FDX1 control blocks on the volume
- The number of allocated FDX2 control blocks on the volume
- The number of allocated FDR control blocks on the volume

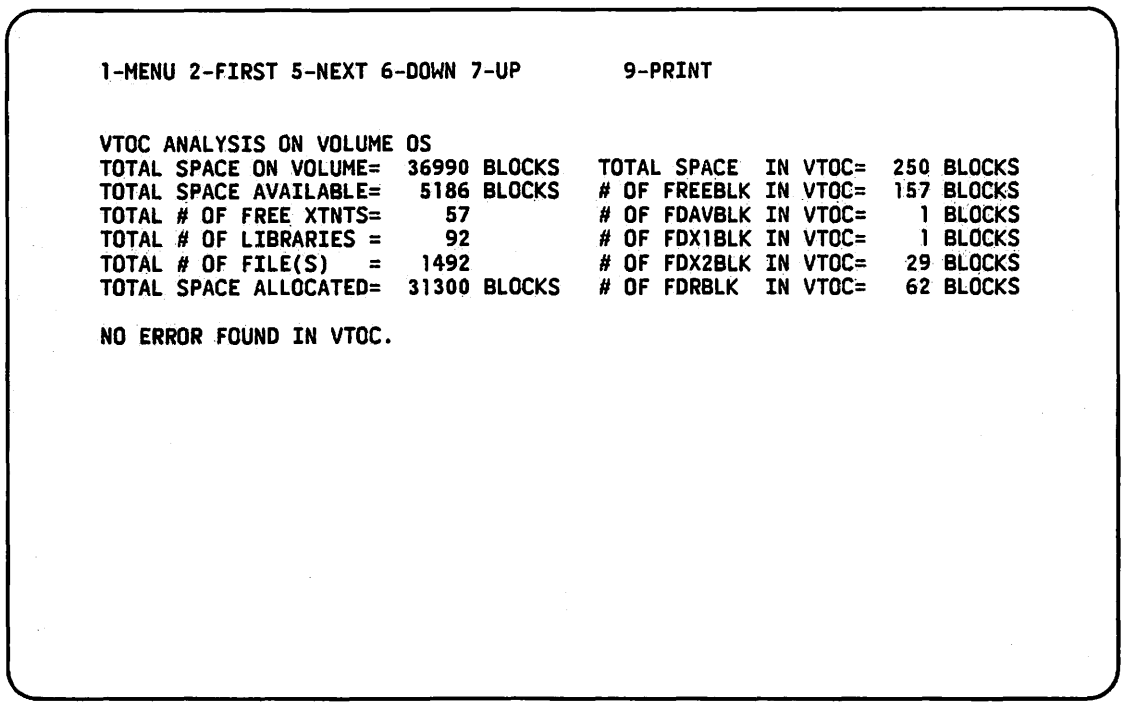
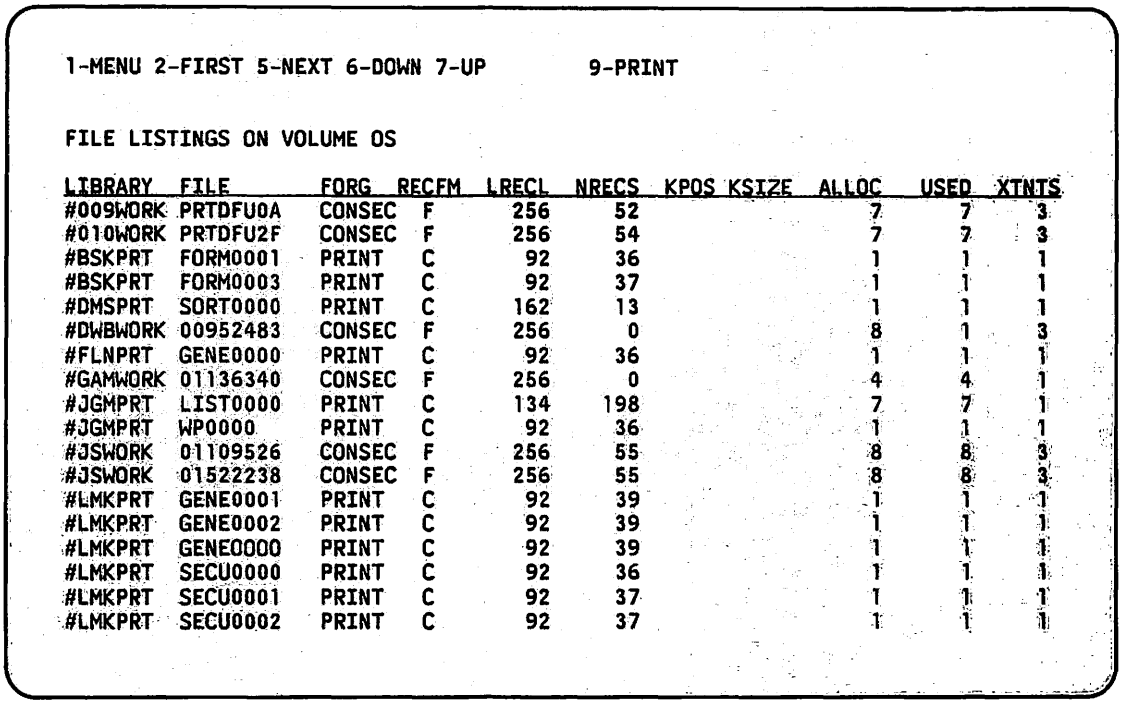


Figure 17-4. Sample LISTVTOC Space Allocation Screen

PF Key	Function	Description
PF3	Display File Listing on Volume	This function displays the LISTVTOC File Listing screen (Figure 17-5) and shows, in sequential order (by library), the file names and attributes of the library or volume that you specified on the LISTVTOC Input Definition screen.

NOTE

LISTVTOC calls the SORT utility to produce a sorted listing (by library) on the LISTVTOC File Listing screen. Therefore, the SORT utility must reside in @SYSTEM@ on the system volume. If SORT is not in @SYSTEM@ on the system volume, the LISTVTOC File Listing screen displays the files randomly. For information about the SORT utility, refer to Chapter 22.



```
1-MENU 2-FIRST 5-NEXT 6-DOWN 7-UP        9-PRINT

FILE LISTINGS ON VOLUME OS

LIBRARY   FILE       FORG   RECFM  LRECL  NRECS  KPOS KSIZE  ALLOC  USED  XTNTS
#009WORK  PRTDFU0A   CONSEC   F      256     52                  7     7     3
#010WORK  PRTDFU2F   CONSEC   F      256     54                  7     7     3
#BSKPRT   FORM0001   PRINT    C       92     36                  1     1     1
#BSKPRT   FORM0003   PRINT    C       92     37                  1     1     1
#DMSPRT   SORT0000   PRINT    C      162     13                  1     1     1
#DWBWORK  00952483   CONSEC   F      256      0                  8     1     3
#FLNPRT   GENE0000   PRINT    C       92     36                  1     1     1
#GAMWORK  01136340   CONSEC   F      256      0                  4     4     1
#JGMPRT   LIST0000   PRINT    C      134    198                  7     7     1
#JGMPRT   WP0000     PRINT    C       92     36                  1     1     1
#JSWORK   01109526   CONSEC   F      256     55                  8     8     3
#JSWORK   01522238   CONSEC   F      256     55                  8     8     3
#LMKPRT   GENE0001   PRINT    C       92     39                  1     1     1
#LMKPRT   GENE0002   PRINT    C       92     39                  1     1     1
#LMKPRT   GENE0000   PRINT    C       92     39                  1     1     1
#LMKPRT   SECU0000   PRINT    C       92     36                  1     1     1
#LMKPRT   SECU0001   PRINT    C       92     37                  1     1     1
#LMKPRT   SECU0002   PRINT    C       92     37                  1     1     1
```

Figure 17-5. Sample LISTVTOC File Listing Screen

NOTE

In volume sets, a listing of files (in a library or volume) is restricted to the root volume of the volume set. The VTOCs of secondary volumes contain only alias library names and alias file names.

PF Key	Function	Description
PF4	Display VTOC Control Blocks	This function displays the LISTVTOC Control Blocks screen (Figure 17-6) and shows the VTOC map on your workstation screen.
PF16	End Job	This function terminates LISTVTOC processing.

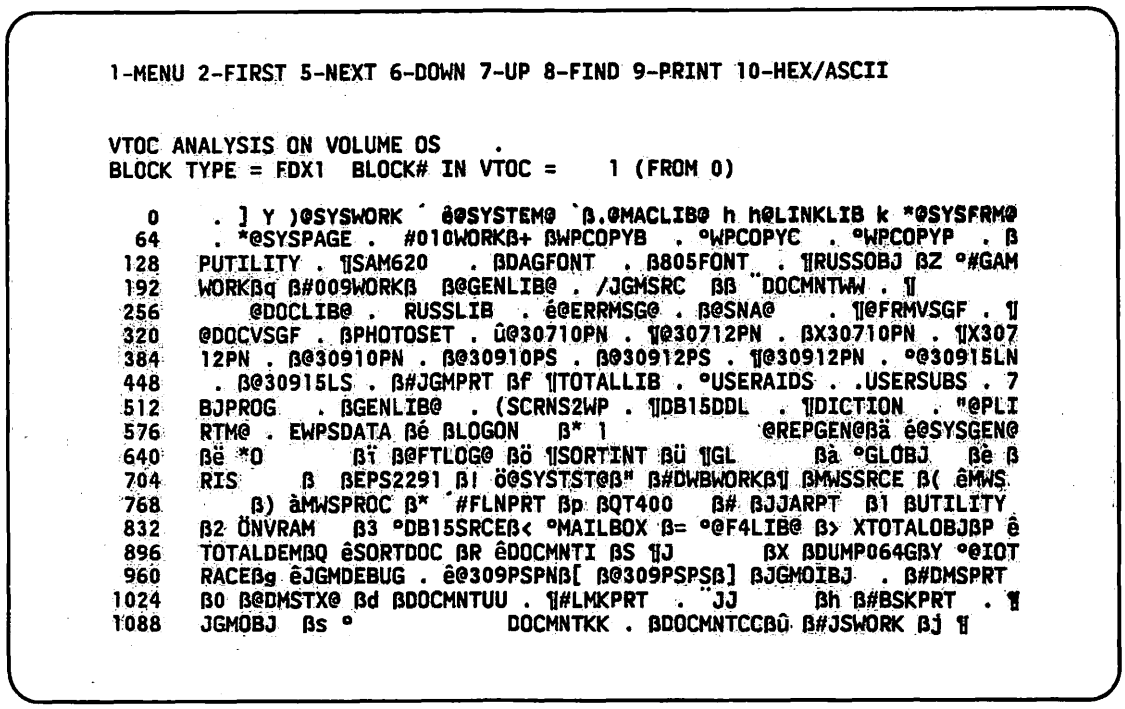


Figure 17-6. Sample LISTVTOC Control Blocks Screen

17.4 THE VTOC MAP

The VTOC map created by LISTVTOC displays in ASCII or hexadecimal notation the contents of the following control blocks:

Block Type	Description
FDAV	FDAV (File Directory Available Space) is the block that contains information for all unallocated areas on the volume.
FDX1	FDX1 (File Directory Index) is the first block that contains entries for each library on the volume and pointers to FDX2 records for each library.
FDX2	FDX2 (File Directory Index) is the second level that contains four FDX2 records in a 2048-byte block. The FDX2 records associate each file name with a library name, and contain pointers to the FDR block containing the file's FDR1 record.
FDR	FDR (File Descriptor Record) is the block that contains FDR1 records that describe the attributes of each file.

For more information about the VTOC, refer to the *VS Operating System Services Reference*.

NOTE
You should run LISTVTOC whenever you initiate Initial Program Load (IPL) procedures (unless you need to get the system running as fast as possible). Verifying your disk can help achieve optimum disk performance.

17.5 A SAMPLE LISTVTOC PROCEDURE

You can control LISTVTOC processing through the VS Procedure language. A list of LISTVTOC GETPARMs is available in Appendix A. Refer to the *VS Procedure Language Reference* for details concerning Procedure language syntax.

The following procedure creates a print file containing a VTOC space and integrity analysis, as well as a dump of the VTOC control blocks of the SYSTEM volume. The procedure eliminates workstation interaction by setting the Screen field to NO.

```
PROCEDURE
    RUN LISTVTOC
        ENTER INPUT VOLUME=SYSTEM, VTOCMAP=YES, SCREEN=NO
RETURN
```

CHAPTER 18
THE OISCART UTILITY

18.1 INTRODUCTION

The OISCART utility enables you to transfer word processing (WP) documents from the Wang Office Information System (OIS) to the Wang VS by using 1/4-inch, 4-track cartridge tapes. OISCART can retrieve WP documents that were created on an OIS and placed on 1/4-inch cartridge tape. This utility fully preserves the structure of the document throughout the transfer (even a document that is listed as "damaged" on the OIS remains in that state after it has been transferred to the VS).

Each OIS document that is transferred to the VS is converted to a VS WP document. For every document transferred by OISCART, a log record is created and placed into a log file named OISCARTL. If OISCARTL already exists, the log records are appended to that file. A control file (OISCARTC) enables you to query or print the log file.

NOTE
You must have a 4-track cartridge tape drive (2529V) and standard 1/4-inch, 4-track cartridge tapes to transfer files from OIS to the VS by means of OISCART.

CAUTION
The cartridge tape must be mounted through the OISCART utility.

The VS microcode for the 2529V cartridge tape drive in its default mode automatically adds a 2-byte sequence number to every block written to the tape. When the VS reads the tape, it consequently strips off those two bytes.

The OIS does not add two bytes to each block on the tape. If the VS reads an OIS tape in its default mode, the VS strips off the first two bytes in each block which is especially destructive to label blocks. However, the VS does offer an optional mode which does not add or strip the two bytes per block. This mode is used when you run OISCART.

While OISCART leaves the cartridge tape microcode in its default mode upon program completion, it is possible that an abnormal termination of OISCART could leave the 2529V cartridge tape drive in either mode. In case of an abnormal termination, turn off the tape drive and then turn it back on at the next mount.

An overview of OISCART processing is provided in Figure 18-1.

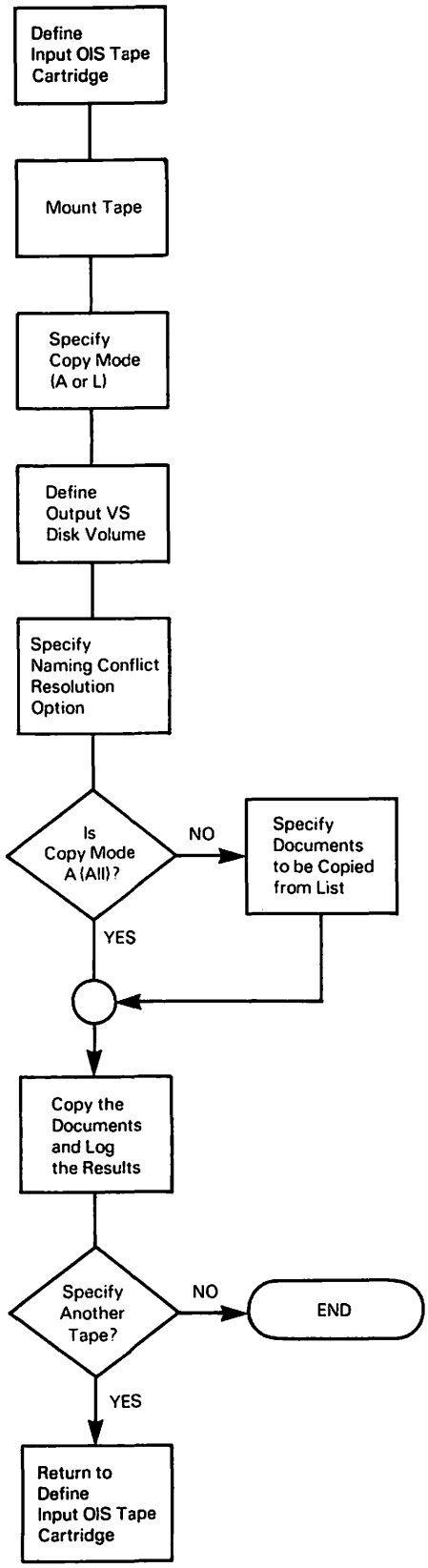


Figure 18-1. OISCART Processing

18.2 SPECIFYING THE OIS INPUT TAPE VOLUME

When OISCART processing begins, the OISCART Input Tape Volume Definition screen (Figure 18-2) appears. Specify the name of the OIS tape volume on which the OIS files reside and the device number on which the tape is to be mounted.

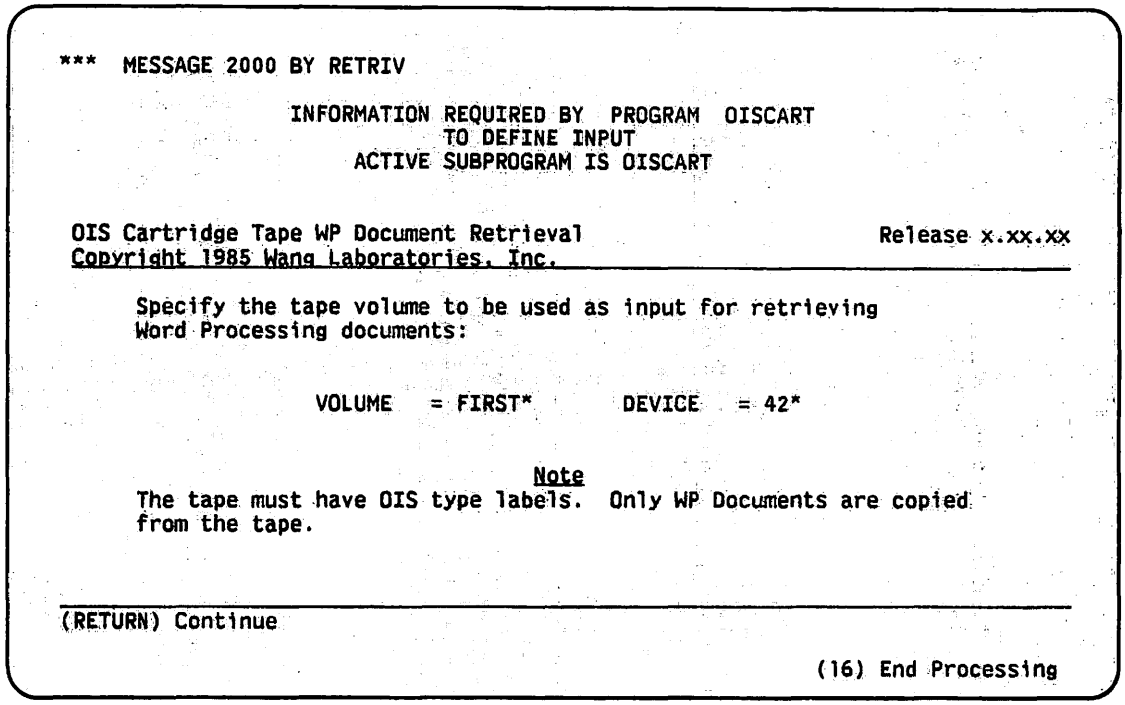


Figure 18-2. Sample OISCART Input Tape Volume Definition Screen

NOTES

The tape and the files on the tape must have been created from an OIS system and have OIS-type labels.

WP Plus documents must be converted to WP format before OISCART can transfer those documents. Only WP documents are copied from the tape. To convert WP Plus Documents into WP format, refer to the WP Plus Conversion Utility in the WP Plus Reference Manual.

18.2.1 The Volume Names of an OIS Tape

You must assign a second name to the OIS tape volume to be used on the VS. This name is used by the VS Operating System when it mounts and dismounts the tape. Because VS volume names accept only six characters, the VS cannot use the 8-character volume name already given to the tape during OIS tape initialization. The 6-character VS volume name that you assign to the tape does not have to have any relation to the 8-character OIS volume name that is already recorded on the tape.

18.2.2 The OISCART Copy Mode Screen

After you have specified the input tape, the OISCART Copy Mode screen (Figure 18-3) appears. The tape volume name on the OISCART Copy Mode screen is the OIS volume name (not the VS-assigned name). If you have not already mounted the tape volume on the tape drive, a Mount Tape Volume screen appears after you specify the input volume and device. To mount a tape volume, insert the cartridge into the drive and press the ONLINE button. For more information about the cartridge tape drive, refer to the *VS Model 2529V Cartridge Tape Drive Operating Procedures Guide*. Once the tape is mounted, OISCART proceeds with the OISCART Copy Mode screen (Figure 18-3).

```
***  MESSAGE 2000 BY RETRIV

                INFORMATION REQUIRED BY  PROGRAM  OISCART
                           TO DEFINE COPYMODE


 Input tape name is NEXTTAPE
_____________________________________________________________________

      Select the desired copy mode for the archive tape:

                                  MODE     = A

            Options:
                    A = Copy all the documents on the tape to the VS
                    L = Select from a list of documents to be copied


_____________________________________________________________________
(RETURN) Continue
(1) Respecify Input
```

Figure 18-3. Sample OISCART Copy Mode Screen

The OISCART Copy Mode screen enables you to select the method for copying the OIS files to the VS. If you specify A (All), all the WP documents (other file types are bypassed) are copied to corresponding VS document libraries. If you specify L (List), all the WP documents are listed on another screen and you specify only those files that you want copied from the tape. After you have specified the Copy mode, press RETURN to continue processing. You can press PF1 to return to the OISCART Input Tape Volume Definition screen to respecify the input.

18.3 SPECIFYING THE VS OUTPUT

The OISCART Output Disk Volume Definition screen appears after you specify the Copy mode for copying the OIS documents. Specify the VS disk volume on which the OIS documents are to be placed and press RETURN to continue processing. Youcan press PF1 to respecify the Copy mode.

When OISCART begins to copy the OIS documents to a VS library, it uses the OIS Document ID to determine the destination library of the document. For example, if OIS Document 0075R is copied to the VS, OISCART attempts to place that document into file 0075 in library DOCMNTR.

NOTE
DOCMNT is the prefix for all VS WP libraries; the suffix in the library name, a single R, indicates an uppercase library letter. If the library name contained a double-R suffix (e. g., DOCMNTRR), all document IDs in that library would have lowercase r for suffixes.

18.3.1 Naming Conflicts

If a document called 0075 already exists in VS Library DOCMNTR on the specified VS output volume, a naming conflict occurs when you attempt to copy 0075R from the OIS tape cartridge. You must specify, before initiating the transfer from tape to VS, the action that you want OISCART to take in case of a naming conflict. The OISCART Naming Conflict Options screen (Figure 18-4) appears after the OISCART Output Disk Volume Definition screen.

NOTE
If you specified L (List) as the Copy mode, OISCART requires you to specify the naming conflict options before selecting the documents for transfer.

Specify one of the following options to resolve any naming conflicts:

1. Specify P (Prompt) in the Action field to prompt you to resolve a naming conflict if one occurs. The OISCART copy operation is suspended while you resolve the naming conflict.
2. Specify D (Delete) in the Action field to automatically delete the VS document whose document ID conflicts with the document ID of the incoming OIS document.
3. Specify F (Find) in the Action field to have OISCART find and use a new VS document ID in the same library without interrupting OISCART processing. OISCART uses the same pointer that is used by the WP Menu option, Create New Document, to find the next available document ID.

To respecify the action to be taken before proceeding further, press PF1. To begin the copy operation, press RETURN.

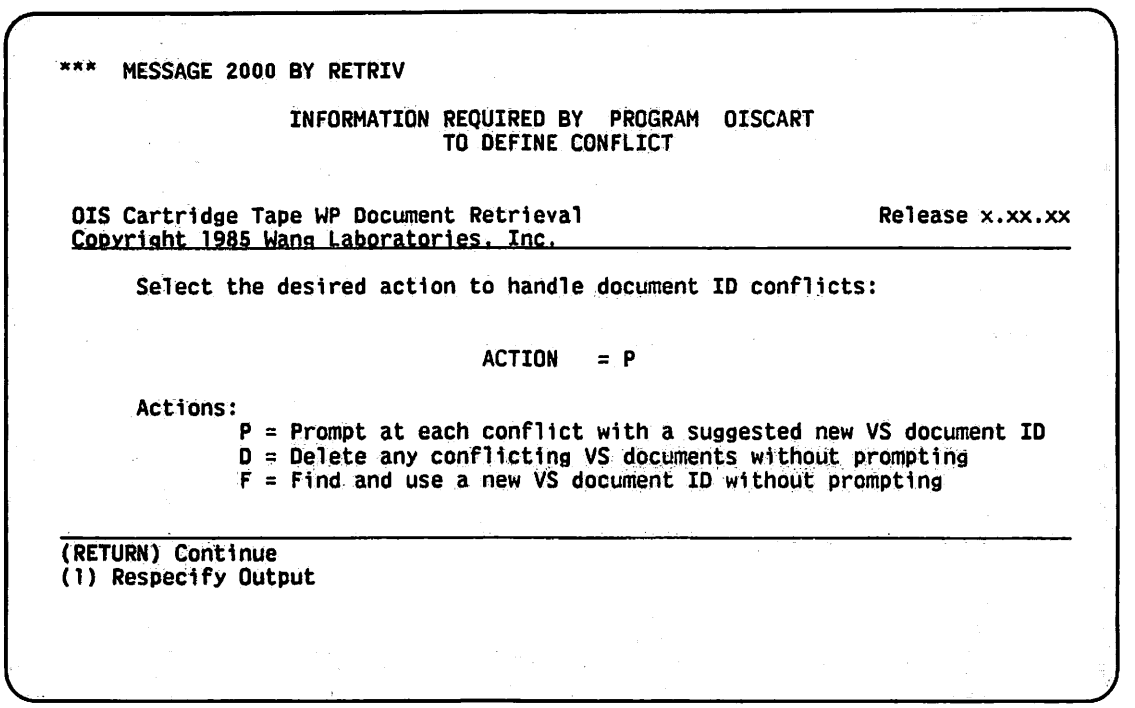


Figure 18-4. OISCART Naming Conflict Options Screen

18.3.2 Specifying Documents from a Document List

If you specified L (List) on the OISCART Copy Mode screen, the OISCART Document Specification screen (Figure 18-5) appears after you have specified the output options. The OISCART Document Specification screen displays a list of all the documents on the OIS tape. Enter any nonblank character in the pseudoblank to specify those documents that you want copied to their corresponding VS libraries.

NOTE

Since the entire tape volume is read, it may take some time before the list is displayed. Also, files that are split between tape volumes cannot be retrieved.

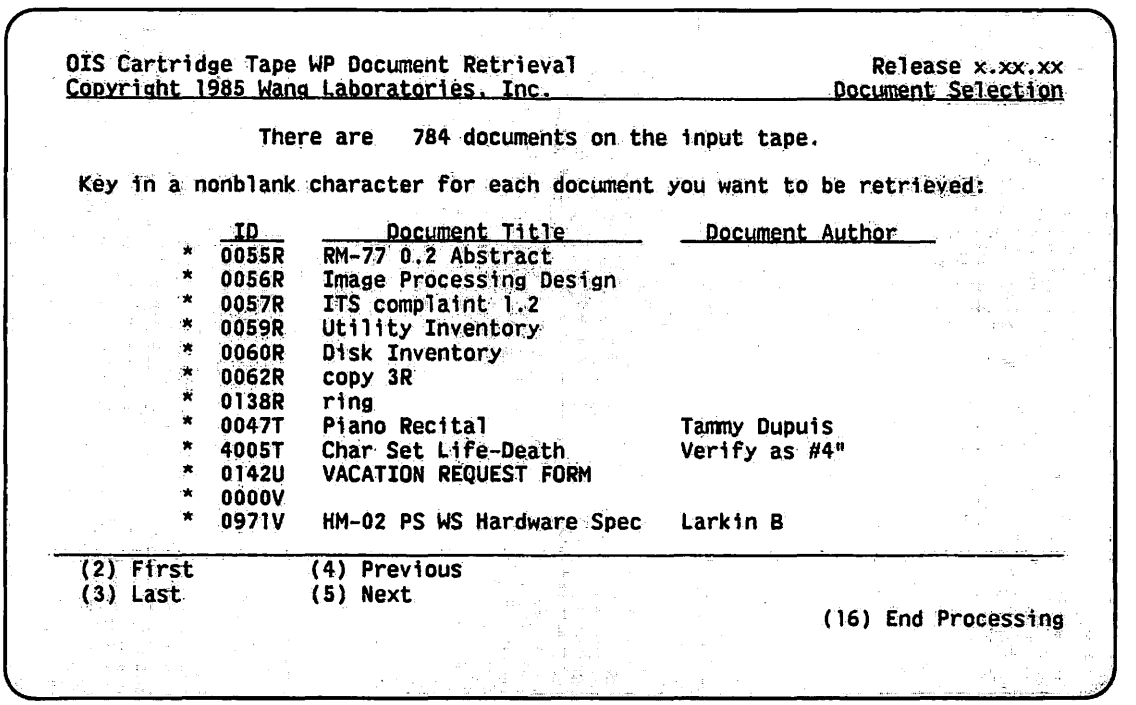


Figure 18-5. Sample OISCART Document Specification Screen

If all of the documents on the tape cannot be displayed on one screen for selection, another OISCART Document Specification screen appears after you make the selections on the first screen and press RETURN. OISCART provides as many screens as are necessary to list all of the documents.

If you want to respecify any of the screens before you press RETURN on the last OISCART Document Specification screen, press PF2 to display the first selection screen. All selections that you previously made are retained. If you want to modify your selections on the third screen, for example, you must press RETURN twice to bypass the first and second screens. When you have specified all the documents that you want copied from the OISCART Document Specification screens, press PF16 from the last OISCART Document Specification screen to begin the copy operation.

18.3.3 Resolving a Naming Conflict

If you specified P (Prompt) in the Action field on the OISCART Naming Conflicts Options screen and a naming conflict occurs, the OISCART Naming Conflict Resolution screen (Figure 18-6) appears. When this screen appears, OISCART prompts you with a suggested new document ID, which has been verified as being an available document ID, for the incoming OIS document. You can accept this document ID by pressing RETURN from the prompt screen, or you can specify another document ID. If you specify another document ID that already exists, a subsequent naming conflict occurs and the OISCART Naming Conflict Resolution screen reappears. To delete the existing VS document and copy the input document with its original document ID, press PF3

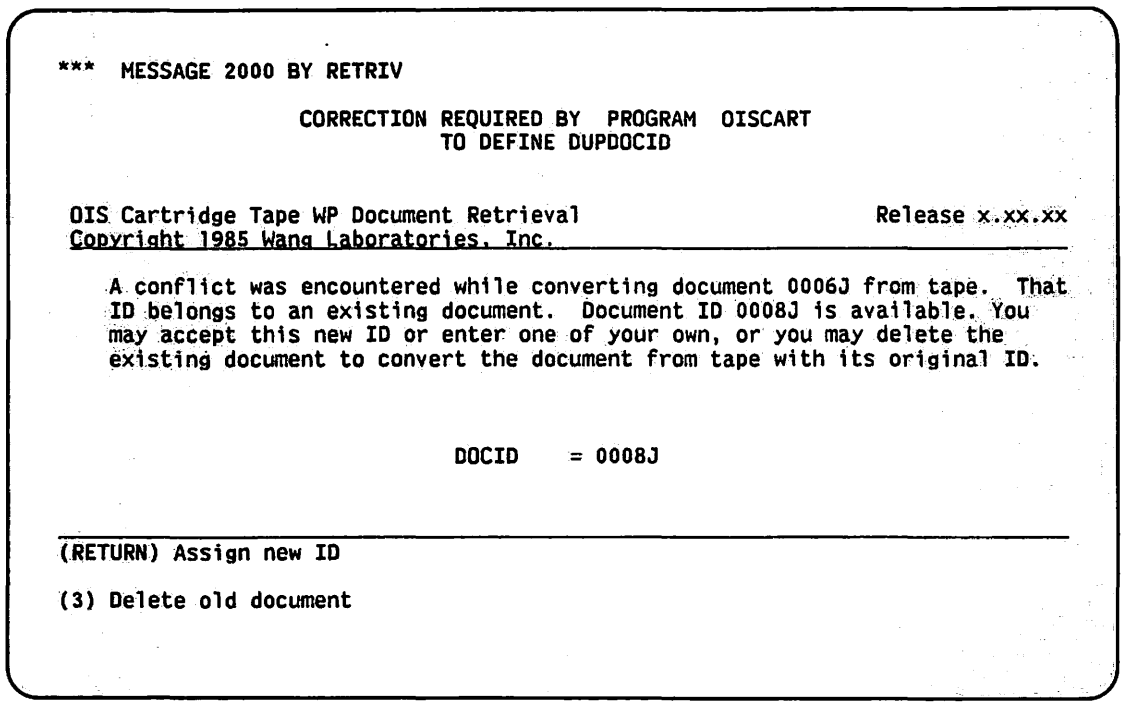


Figure 18-6. Sample OISCART Naming Conflict Resolution Screen

18.3.4 The Dismount and End-of-Job Screen

When the copy operation is completed on all the documents that you selected from the tape, the operating system displays the Dismount screen. Press RETURN to acknowledge the dismount.

After the dismount screen, an End-of-Job screen appears and shows the number of documents that have been copied. Press PF1 to rerun OISCART for another tape, press PF9 to view the log file, or press PF16 or RETURN to terminate OISCART processing.

18.4 THE OISCART LOG FILE

The OISCART log file enables you to keep track of each document transfer that occurs during OISCART processing. For every document transferred by OISCART, a log record is created, which contains the following information:

- The source (OIS) volume and document IDs
- The destination (VS) volume and document IDs
- Document name
- Date and time of conversion

Every time you run OISCART and transfer documents to the VS, the most recent log is appended to the existing log file. If you want to keep the log file of each set of transfers separate, either copy the existing log file to another file or delete it.

The OISCART utility uses the CONTROL, DISPLAY, and INQUIRY utilities to access and display the log file. For more information about the DISPLAY utility, refer to Chapter 8. For more information about the CONTROL and INQUIRY utilities, refer to the *VS File Management Utilities Reference*.

18.4.1 OISCART, OISCARTC, OISCARTI, and OISCARTL

The OISCARTC, OISCARTI, and OISCARTL files are linked either by the OISCART utility or are created by the OISCART utility.

- The OISCARTC file is a control file, created by the CONTROL utility, that defines the structure of the records that are placed into OISCARTL. OISCARTC enables you to query or print the log file.
- The OISCARTI file is an INQUIRY procedure created to display the OISCART log file. OISCARTI can be modified so that the output better suits your particular needs.

  NOTE
  If you want to make any modifications to OISCARTI, you should make a copy of OISCARTI and modify the **copy** *to prevent damage to the original OISCARTI file.*

- The OISCARTL file is the log file that is created by the OISCART utility. Each record of the log is placed into a log file during OISCART processing.

18.4.2 Accessing the Log File

To access the log file, press PF9 from the End-of-Job screen. (You can also access the log file at a later time by running OISCARTI through the Command Processor menu and press PF1 (RUN Program or Procedure). Specify OISCARTI in the Program field and the appropriate library and volume names, and press ENTER.)

The OISCART Log File Inquiry screen (Figure 18-7) appears. You can either view an individual document transfer record by specifying the document in the OIS Document ID field, or leave the OIS Document ID field blank to view all the records in the log. Press RETURN in either case to continue processing.

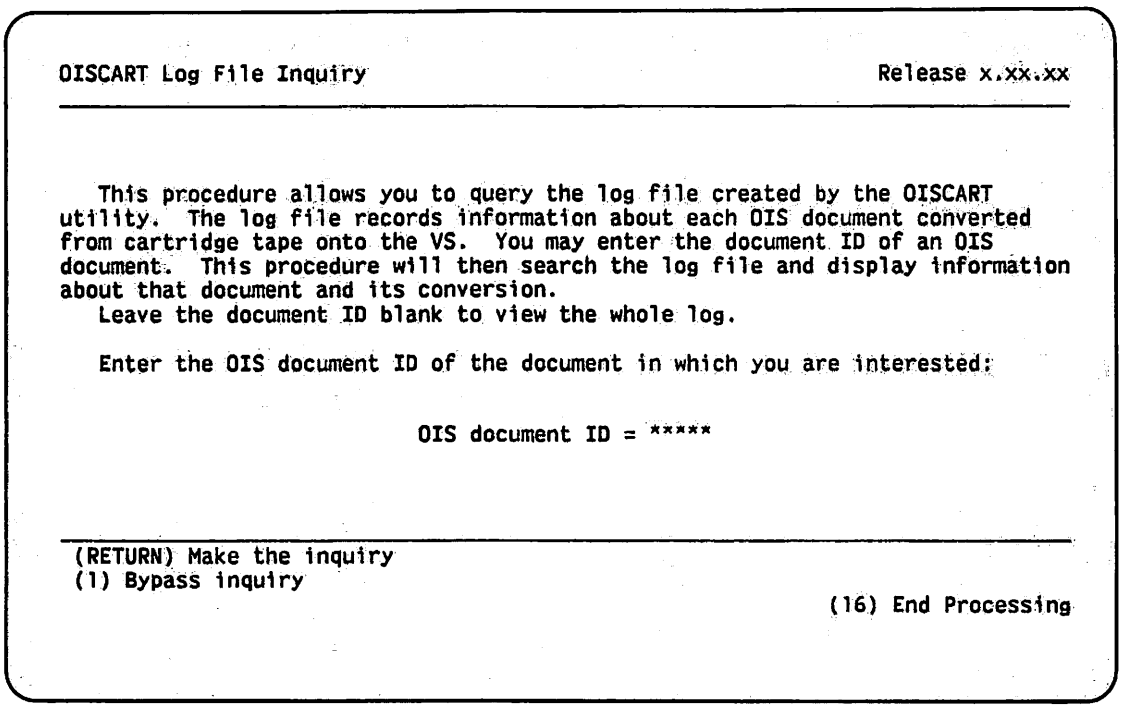


Figure 18-7. OISCART Log File Inquiry Screen

The information shown in Figure 18-8 is too wide to fit entirely on the workstation screen. OISCART uses the DISPLAY utility to display the log file. To view the list of DISPLAY utility options, press PF1 from the screen that displays the log file. To print the OISCART log file, press PF15 from the screen that displays the log file or from the DISPLAY Options screen (refer to Chapter 8 for more information about the DISPLAY utility). Press PF16 when you are finished viewing the log file.

OIS DOCUMENT ID	OIS SYSTEM NAME	VS DOCUMENT ID	VS VOLUME NAME	DOCUMENT NAME	CONVERSION DATE	CONVERSION TIME
0007	G	0807	SYSTEM	chemical structures	850512	094408
0010	G	0810	SYSTEM	Glossary for loading font	850512	094418
0006C	G	0006C	SYSTEM	******DOCUMENT INDEX******	850512	094428
0024C	G	0024C	SYSTEM	TELEPHONE LIST DEPT. 14	850512	094434
0004D	G	0054D	SYSTEM	price list	850512	094444
0005D	G	0055D	SYSTEM	cost list	850512	094456
0076D	G	0076D	SYSTEM	View Graphs	850512	094502

Figure 18-8. Sample OISCART Log File Printout

When you press PF16 after viewing the log file, the next screen enables you to perform another inquiry by pressing PF1, or you can continue (do not perform another inquiry) by pressing RETURN. When you press RETURN, the next screen prompts you to either delete the log file by pressing PF3 or end OISCART processing (and save the log file) by pressing PF16.

18.5 A SAMPLE OISCART PROCEDURE

You can control OISCART processing through the VS Procedure language. Appendix A contains a list of OISCART GETPARMs. Refer to the *VS Procedure Language Reference* for details about Procedure language syntax.

The following procedure copies all of the documents from the OIS tape volume CVOL (cartridge volume) using Device 42. The Copy mode is set to A (All) because an L value (List) requires user interaction. The output volume is the VS destination volume (SYSTEM, in this case). The Naming Conflict Resolution option is F (Find) to automatically find a document location in case of a naming conflict without deleting any of the existing VS documents. OISCART processing concludes by the EOJ 16 statement (end-of-job, PF16).

```
PROCEDURE
    RUN OISCART
        ENTER INPUT VOLUME=CVOL, DEVICE=42
        ENTER COPYMODE MODE=A
        ENTER OUTPUT VOLUME=SYSTEM
        ENTER CONFLICT ACTION=F
        ENTER EOJ 16
RETURN
```

CHAPTER 19
THE PATCH UTILITY

19.1 INTRODUCTION

The PATCH utility enables you to display and modify the hexadecimal value of any file in 16-byte fields at your workstation. You can also produce a dump of an entire file or a specified portion of a file.

Even though you can use PATCH on any file, the PATCH utility is especially useful for VS system dumps. For this reason, this chapter concentrates on using PATCH for enabling the system to automatically perform a system dump when the need for one arises.

19.1.1 VS System Dumps

The VS system dump routine creates a physical memory image on a dump device when the operator invokes it or when the system detects internal errors. There are four types of system dumps:

1. Operator-forced dump from Control mode — occurs as the result of a fatal system error and is initiated by the operator
2. System-forced dump caused by a fatal error — occurs as a result of a fatal error and is initiated by the system
3. Continuable dump — occurs as a result of a nonfatal error and is initiated by the system
4. Snapshot dump — initiated by the operator when the system is not frozen and a dump of main memory is desired. The system resumes normal operation after a snapshot dump.

For more information on system dumps, refer to the *VS System Operator's Guide* or to the individual processor handbook for your computer system.

19.2 USING PATCH FOR SYSTEM DUMPS

The first four bytes in the dump file, @CMDUMP@, contain a protection code that prevents the system from writing any data into the file unless specifically initiated by an operator. However, the PATCH utility enables you to dump directly into the @CMDUMP@ file by resetting the first four bytes to zero in the @CMDUMP@ file, thereby reducing the time that system resources are suspended. Through the PATCH utility, you can set the @CMDUMP@ file to automatically receive system dump data when a system dump condition arises, without operator interaction.

NOTE
You should reset the first four bytes of the @CMDUMP@ file back to zero ***after*** *you have examined the dump information (or copied it to another file). This is to allow system operations to continue normally in case of another system dump condition. Setting the first four bytes after each dump can help system operation proceed as normally as possible under any system dump condition.*

If operations are suspended as the result of a fatal system error and the @CMDUMP@ file is not set to receive the dump data, an operator must go to the operator console (Workstation 0) and initiate a dump of main memory into the @CMDUMP@ file. System resources are "frozen" until the dump is initiated. When the dump is complete, the system automatically re-IPLs using the SYSGEN data from the previous IPL and users must log on again to resume operations.

If a nonfatal system error occurs, an operator must also go to the operator console and initiate the dump if the dump file is not set to receive the data. System resources are "frozen" while the dump is being completed but re-IPLing is not necessary when nonfatal system errors occur. When the dump is complete, operations can resume.

When the system operator wants a snaphot dump, the operator must initiate the dump. Using the PATCH utility, the operator can obtain a snapshot dump with only a brief pause to system resources.

19.2.1 Continuable Dump Processing

If a continuable dump condition arises and the protection bytes on the @CMDUMP@ file have **not** been reset, the operator console displays a message that an error warranting a continuable dump has occurred and offers the following options:

PF Key	Function
1	Ignore the current dump and continue processing. (The contents of @CMDUMP@ remain unchanged.)
3	Allow the dump to overwrite the data in @CMDUMP@. Data currently in @CMDUMP@ is lost.
16	Enter Control mode

NOTE
You can also specify another device to receive the dump information ***but*** *the device that you specify must have another @CMDUMP@ file residing in another @SYSDUMP library.*

If the first four bytes are set to zero, system operations pause for approximately 15-seconds (workstation screens "freeze" for approximately 15 seconds) while a continuable dump takes place. Setting the first four bytes to zero leaves @CMDUMP@ unprotected from writing into it when a continuable dump condition occurs. By resetting the first four bytes in @CMDUMP@ beforehand, you can avoid waiting for system operations to return to normal while someone selects an option from the Operator's Console menu.

After the dump is finished, a message is displayed at the operator console and the first four bytes are reset to protect the new dump information. Normal operations can then resume. If another continuable dump condition occurs before the last dump file has been examined or copied to another file, the system halts and redisplays the continuable dump options on the Operator's Console menu.

19.2.2 Snapshot Dump Processing

Snapshot dumps are initiated through the system operator and not through a specific error. The first four bytes must still be set to zero in order to dump into @CMDUMP@. Through PATCH, you can obtain a snapshot dump with only a pause in system operation. To invoke a snapshot dump, perform the following steps:

1. Press PF11 to enter the Operator's Console menu from the Command Processor menu. (Only system operators and the operator console can access the Operator's Console menu.)
2. Press PF14 (SYSTEM Options) from the Operator's Console menu.
3. Press PF10 to initiate a snapshot dump from the System Options menu.

19.3 SETTING THE AUTOMATIC DUMP BYTES THROUGH PATCH PROCESSING

The first four bytes in the @CMDUMP@ file comprise a default value that protects the file from automatically having other data written over the information currently in the dump file. The PATCH utility enables you to reset these bytes, making the dump file reusable for system dumps.

To save the existing dump information, run the COPY utility and copy the existing dump file to a separate file. Refer to Chapter 3 for more information about the COPY utility. Send the copied file to Wang support for analysis. After copying the existing dump file, reset the first four bytes on the @CMDUMP@ file by running the PATCH utility. With the PATCH utility, you can enable automatic dumping so that the file for which you want a dump can write over the existing information in @CMDUMP@. If you do not want to keep the current dump information, just run the PATCH utility without running the COPY utility.

An overview of PATCH processing is provided in Figure 19-1.

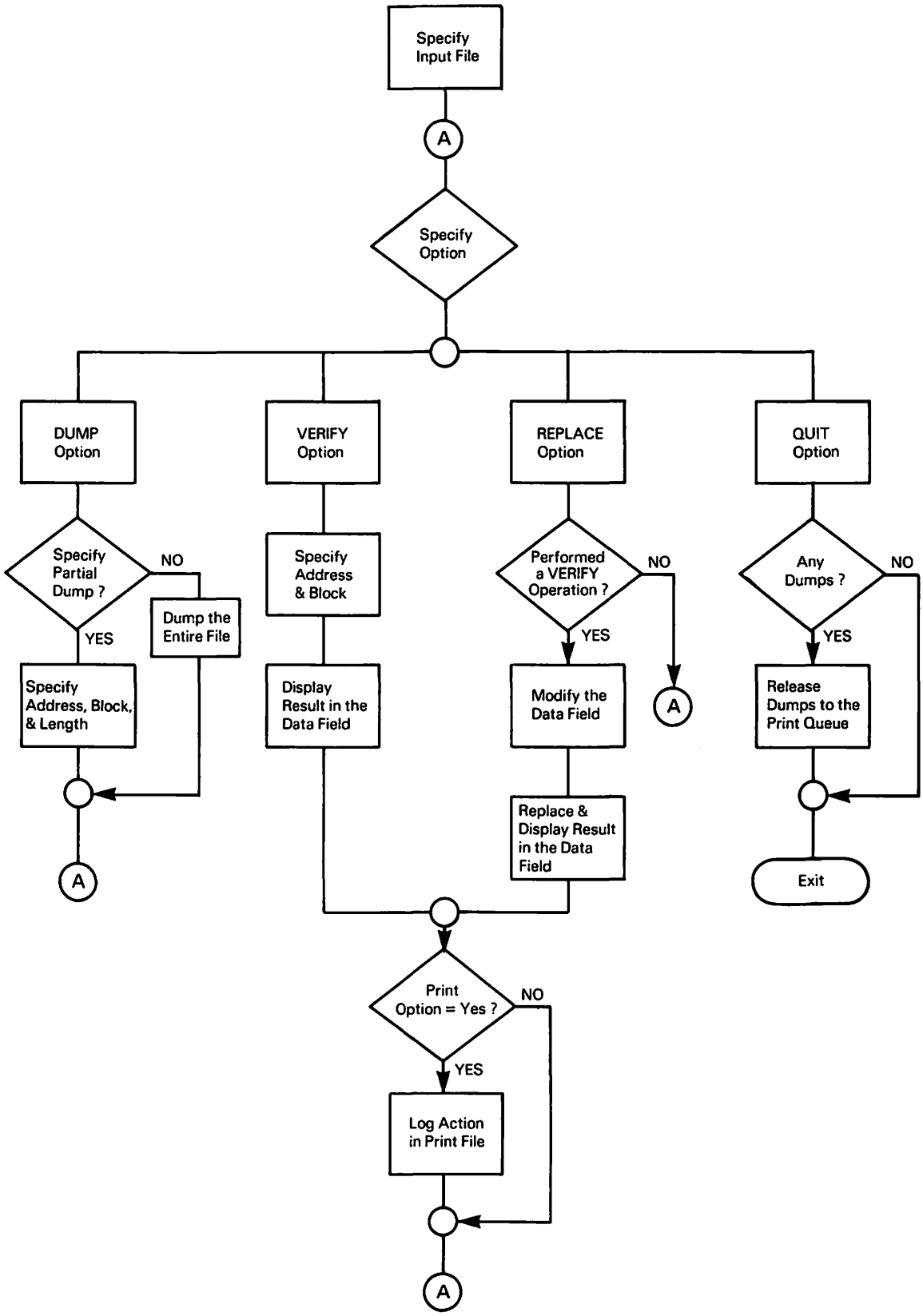


Figure 19-1. PATCH Processing

To run the PATCH utility and reset the @CMDUMP@ file, perform the following steps:

1. Press PF1 (RUN Program or Procedure) from the Command Processor menu.
2. Specify PATCH in the Program field and press ENTER.
3. When PATCH processing begins, the PATCH Input Definition screen (Figure 19-2) appears and prompts you to enter the file, library, and volume names, and the device on which the PATCH operation is to be performed.
   a. Specify @CMDUMP@ in the File field.
   b. Specify @SYSDUMP in the Library field.
   c. Specify the name of the volume on which @SYSDUMP resides in the Volume field.
   d. Specify DISK in the Device field.

```
***  MESSAGE 000  BY OPEN

                  INFORMATION REQUIRED BY  PROGRAM  PATCH
                              TO DEFINE IOFILE




PLEASE ASSIGN "IOFILE"          (TO BE UPDATED BY THE PROGRAM)

TO ASSIGN THIS FILE TO A DISK FILE, PLEASE SPECIFY:
    FILE     = @CMDUMP@ IN LIBRARY  = @SYSDUMP ON VOLUME   = SYSTEM


    DEVICE   = DISK*******
```

Figure 19-2. Sample PATCH Input Definition Screen

4. When you press ENTER from the PATCH Input Definition screen, the PATCH Operations screen (Figure 19-3) appears.
   a. Specify VERIFY in the Option field.
   b. Specify 0 in the Address field.

```
***  MESSAGE 0000 BY PATCH

                INFORMATION REQUIRED BY  PROGRAM  PATCH
                          TO DEFINE COMMAND


WANG VS FILE PATCH PROGRAM - VERSION  x.xx.xx

ACTIVE FILE IS @CMDUMP@ IN LIBRARY @SYSDUMP ON VOLUME SYSTEM

OPTION   = ******* (DUMP  VERIFY  REPLACE  QUIT)

ADDRESS  = ****** IN BLOCK    = ********       LENGTH   = ****** (FOR DUMP)

DATA     = ******** ******** ******** ********


PRINT    = NO*
LINES    = 55 PER PAGE
```

Figure 19-3. PATCH Operations Screen

5. Press ENTER. The Data field now shows the hexadecimal code for the first 16 bytes of the @CMDUMP@ file. (Every two digits represents one byte.)
6. Set the first four bytes (first eight digits) to 0
7. Specify REPLACE in the Option field and press ENTER.
8. Specify QUIT in the Option field to terminate PATCH processing.

The PATCH Operations screen (Figure 19-3) fields are described as follows:

Field	Description
OPTION	Specify DUMP, VERIFY, REPLACE, or QUIT. The first letter of each option is sufficient. DUMP — The Dump option places a hexadecimal dump of all (or the specified portion) of the file in a print file. If you leave the Address field, the Block field (or the Section field, which only appears for object files), and Length field blank, the PATCH utility dumps the entire file. The Print field is ignored because a printout is always created for the Dump option. The print file that is created is not released until PATCH processing ends (by specifying Quit in the Option field). For information about setting print mode and spool volume defaults, refer to the *VS System User's Introduction*.

Field	Description
OPTION (cont.)	VERIFY — The Verify option displays the 16 bytes (in hexadecimal numbers) that are located at the address specified in the Block (or Section) field. You must perform a Verify operation before you can perform a Replace operation. If you specify Block (or Section) and leave Address blank, the PATCH utility displays the first 16 bytes (in hex) of that block (or section). The first block or section of the file is assumed when the Block field (or Section field) is blank. After you specify Verify and the appropriate fields, press ENTER. When the Verify option has been executed, the bytes requested are displayed in the Data field. The message DATA VERIFIED appears in the upper left-hand corner of the screen. REPLACE — The Replace option enables you to modify the contents of a file by changing its hexadecimal displayed notation. Only after data to be patched is displayed by the Verify option can you execute the Replace option. To change any portion of the 16-byte hexadecimal display, specify Replace in the Option field, modify the Data field as desired, and press ENTER. (The Address and Block (or Section) field default values should not be altered.) The contents of the file at the specified location are replaced by the hexadecimal notation you specified in the Data field. No replacement occurs for Data field positions that are deleted by use of the space bar. If you specify YES in the Print field, a line is placed in the print file containing the word Replace, the specified address, the specified block (or section), and the specified hexadecimal notation that replaced the 16 bytes. QUIT — The Quit option terminates PATCH processing and releases a single print file that contains all PATCH- generated print output.
ADDRESS	Specify the hexadecimal address in the specified block (or section) of the data to be dumped, verified, or replaced. If you do not specify an address, a value of zero is assumed.
BLOCK or SECTION	The Block field appears when you run PATCH on anything except an object program. The Section field only appears for object programs. For all files (except object programs), you can specify the 2-KB block in which the data is located as a decimal digit. Block 0 (the first block of the file) is assumed if you do not specify a block. For object programs, specify the location of the object program (or portion) in which the data is to be patched or dumped. If you do not specify a block or section, the beginning of the program is assumed.
LENGTH	Used only in conjunction with the Dump option, the Length field indicates the number of bytes (in hexadecimal notation) to be dumped. If you do not specify a Length value, an entire block (or section) is dumped.
DATA	The Data field shows 16 bytes of hexadecimal data, beginning at the address specified in the Block (or Section) field, when the Verify option is processed. When the bytes are displayed (after Verify is processed) you can then modify the Data field and process the Replace option.

Field	Description
PRINT	The Print field determines whether a line is placed into the print file for each Verify and Replace option processed. The default value is NO. If you specify YES in the Print field, a line is placed in the print file that contains the word Verify or Replace (whichever option is being processed), the address, the block (or section), and the sixteen bytes of verified or replaced data.
LINES	Specify the number of lines to be printed on each page of the output file. The default value is 55.

NOTE
A message is displayed for each option, confirming the option that has just been processed.

19.4 A SAMPLE PATCH PROCEDURE

Although most PATCH functions are normally interactive, PATCH processing can be controlled through the VS Procedure language. This procedure resets the @CMDUMP@ file for automatic dump processing (as described in the procedure in Section 19.3). A complete list of PATCH GETPARMs can be found in Appendix A. Refer to the *VS Procedure Language Reference* for details on Procedure language syntax.

```
PROCEDURE      RUN PATCH
        ENTER IOFILE FILE=@CMDUMP@, LIBRARY=@SYSDUMP,
            VOLUME=SYSTEM
        ENTER COMMAND OPTION=VERIFY, ADDRESS=0
        ENTER COMMAND OPTION=REPLACE, ADDRESS=00000000,
            DATA=00000000
        ENTER COMMAND OPTION=QUIT
RETURN
```

CHAPTER 20
THE POOLSTAT UTILITY

The POOLSTAT utility enables you to monitor the use of page pools on any VS system (except VS80). The system monitors the use of the page pools and issues warnings when a page pool nears capacity. You can view page pool statistics at any time through the POOLSTAT utility. You allocate a page pool to a volume through the DISKINIT utility. For more information about DISKINIT and page pools, refer to Chapter 7.

When POOLSTAT processing begins, the System Pagepool Monitor screen (Figure 20-1) displays the utilization statistics for one page pool. If your system has more than one page pool, you can view the statistics for each pool by pressing either PF4 (Previous Page Pool) or PF5 (Next Page Pool) from the System Pagepool Monitor screen.

```
                System Pagepool Monitor - Version  x.xx.xx

                          Update every 2 seconds

Volume:                                  SYSTEM

Capacity:                               2.9 MB
Current usage:                          1.2 MB   41%
Peak usage:                             1.5 MB   52%

Memory commitment:                     11.8 MB  403%
User count:                               6

Ref rate (Immed.):                     0.0/sec
Ref rate (20% decay):                  0.0/sec


      PRESS:  ENTER  To Update Immediately
                (4)  Previous Page Pool
                (5)  Next Page Pool
               (16)  Terminate
```

Figure 20-1. Sample System Pagepool Monitor Screen

If a page pool has been allocated a capacity of greater than 32 MB, the System Pagepool Monitor screen displays an asterisk next to the page pool's capacity and advises you that the excess capacity is not used. For each page pool, the System Pagepool Monitor screen provides the following information:

Field	Description
Volume	The Volume field indicates the name of the volume on which the page pool resides.
Capacity	The Capacity field indicates the actual size of the page pool.
Current Usage	The Current Usage field indicates the amount of space being used by tasks assigned to the page pool, and the percentage of the pool capacity that this represents.
Peak Usage	The Peak Usage field indicates the maximum amount of space used by tasks assigned to the page pool since the pool was initialized, and the percentage of the pool capacity that this represents. Access is allowed to a page pool at Initial Program Load (IPL) if the volume is enabled for paging; otherwise, when you enable the volume for paging (through the Operator's Console menu).
Memory Commitment	The Memory Commitment field indicates the total modifiable data area size currently assigned to the page pool, and the percentage of the pool capacity that this represents.
User Count	The User Count field indicates the current number of tasks assigned to the page pool.
Ref Rate (Immed.)	The Ref Rate (immediate) field indicates the number of I/O operations that were made to the pool during the last second.
Ref Rate (20% Decay)	The Ref Rate (20% decay) field indicates the average pool I/O rate which is displayed for any given amount of time but is weighted in favor of the last several seconds.

NOTE

You should run POOLSTAT when you first establish a page pool on your system. POOLSTAT statistics can help you to determine if your system's current page pool capacity is adequate for the paging requirements of the tasks assigned to the pagepool. Refer to Chapter 7 for information about estimating the size of a page pool.

CHAPTER 21
THE SHRSTAT UTILITY

The SHRSTAT utility enables you to display statistics that the Sharer collects as it processes user requests. The Sharer is a dedicated system task that coordinates access to shared files. For more information on the Sharer, refer to the *VS Data Management System (DMS) Reference*.

When the Data Management System (DMS) receives requests by multiple users for access to a shared file, the DMS issues control instructions to the Sharer. The Sharer coordinates access to the shared files, and returns the requests to the DMS. The system records the Sharer's activity and the SHRSTAT utility creates a statistics report on that activity.

When you run the SHRSTAT utility, the SHRSTAT Sharer Statistics screen (Figure 21-1) displays statistics accumulated since the last Initial Program Load (IPL).

```
        ***  Wang VS Sharer Statistics  -  Version x.xx.xx  ***
                        [Sharer Version x.xx.xx]

     Buffer Pool Information              Sharer Memory Pool Information

     # of buffers: 32                    Current memory available: 970548 bytes
        Hit count: 28                     Least memory available: 965540 bytes
       Miss count: 30                   Peak memory load occurred at 21:29:30
  Hit/miss ratio: .93 to 1                                      on 05/07/85

      DMS Requests Processed              Miscellaneous Information

            Open: 10                      Total messages processed: 70
           Close: 10                           Current # of users: 0
            Read: 36                      Most simultaneous users: 1
           Write: 1                       Current # of open files: 0
         Rewrite: 0                   Most simultaneous open files: 2
          Delete: 0
           Start: 0                       Buffers are not fixed
     Adv. Sharing: 0                      Control blocks are not fixed

Please press <Enter> to update, PF 1 to establish a counter baseline,
        PF 17 to revert from the baseline, or PF 16 to exit
```

Figure 21-1. Sample SHRSTAT Sharer Statistics Screen

The following categories of information are shown on the SHRSTAT Sharer Statistics screen:

- **Buffer Pool Information** — This category displays the following fields:
  - Number of buffers
  - Hit count
  - Miss count
  - Hit/Miss ratio

  NOTE
  Hit count and Miss count refers to the number of times the Sharer did and did not find information when searching the buffers.

- **DMS Requests Processed** — This category displays the number of requests processed for the following operations:
  - Open
  - Close
  - Read
  - Write
  - Rewrite
  - Delete
  - Start
  - Advanced Sharing

  NOTE
  The Close field displays statistics for explicit Close operations only. More information on Advanced Sharing can be found in the VS DMS/TX Reference.

- **Sharer Memory Pool Information** — This category contains internal memory management statistics. The category displays the following fields:
  - Current memory available
  - Least memory available (since the last IPL)
  - Peak memory load (the time and date when the least memory available occurred)
- **Miscellaneous information** — This category is an indication of the Sharer workload. This category displays the following fields:
  - Total messages processed
  - Current number of users
  - Most simultaneous users (since the last IPL)
  - Current number of open files
  - Most simultaneous open files (since the last IPL)

The following functions are available to you from the SHRSTAT Sharer Statistics screen (Figure 21-1):

PF Key	Function	Description
ENTER	Update	Update all information fields.
1	Establish Counter Baseline	Resets all Buffer pool statistics, DMS request statistics, and miscellaneous information fields to zero. This establishes a counter statistic baseline to capture test statistics. Subsequent updates reflect changes to the counter statistics since the establishment of the baseline.
16	Exit	Terminate the SHRSTAT program.
17	Revert from the Baseline	Cancel the counter statistics baseline. (Fields revert to displaying statistics accumulated since the last IPL.)

CHAPTER 22
THE SORT UTILITY

22.1 INTRODUCTION

The SORT utility enables you to sort a file according to one or more key fields within the records. You can also merge two or more sorted files. With these two capabilities, you can perform the following functions:

- Sort a single data file into an order that you specify.
- Sort up to 10 data files into a single, ordered output file.
- Merge up to 20 data files into a single, ordered output file.
- Merge files that are already sorted into a single, ordered output file.
- Select specific records from one or more input files and sort them into a single, ordered output file.
- Produce an output file that contains only the primary index key field from each record in the input file. You specify the sort order for the input primary index key field.

You can retrieve input files from tape volumes (which have consecutive files) or disk volumes (which have consecutive, indexed, or relative files). The SORT utility leaves the input file(s) intact unless you assign the same name to the input and output files (which replaces the input file with the output file). SORT always creates the output file, containing the sorted or merged records, as a consecutive file.

The Sort function of the utility takes an unordered file (or files) and places it into a specified order, producing **one** ordered, consecutive, output file. You specify the order of the output file. When you sort two or more files, the utility automatically merges the files.

The Merge function takes two or more files that already have the same format and order, and combines them into one ordered, consecutive, output file. Both Sort and Merge allow you to extract specific input records or parts of records (fields) for processing. If any file has not been previously sorted when attempting to merge two or more files, the program is terminated.

To perform SORT processing, you must specify the following items in the order listed:

1. Specify the program options (SORT or MERGE).
2. Specify the input file(s) to be sorted or merged.
3. Specify the keys with which you are going to sort or merge the files.
4. Specify the output file (destination of the output).

Figure 22-1 shows an overview of SORT processing.

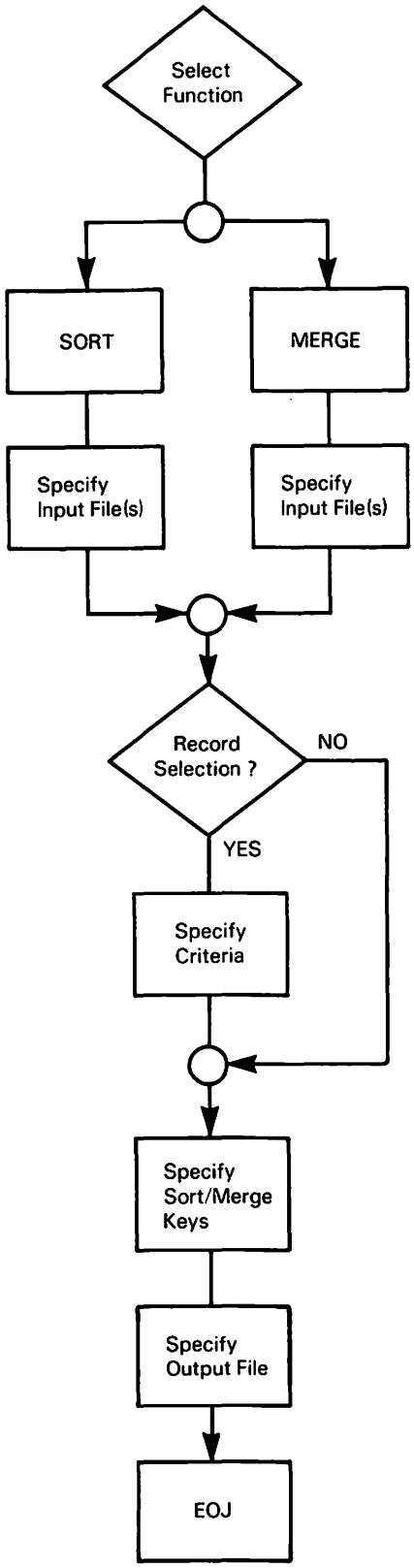


Figure 22-1. SORT Processing

22.2 RUNNING THE SORT UTILITY

To run the SORT utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify SORT in the Program field and press ENTER. When SORT processing begins, the SORT Utility Options screen appears (Figure 22-2). Specify the function (SORT or MERGE) to be performed, the amount of memory to be used for processing the Sort or Merge operation, and the output selection criteria that you need to process the files. To terminate SORT processing without defining the program options, press PF16.

```
***  MESSAGE 0000 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORT
                           TO DEFINE OPTIONS

                                        o


You are now running Wang VS Sort/Merge Program,. Version xx.xx.xx
Please specify the program options below:


          FUNCTION = SORT*   (SORT or MERGE)
          MEMORY   = 128 K   (Memory used, Maximum size = 174 K)


Do you want an ADDROUT output file ?       ADDROUT  = NO*  (YES or NO)
Do you want a KEYOUT output file ?         KEYOUT   = NO*  (YES or NO)
Do you require a STABLE sort ?             STABLE   = NO*  (YES or NO)
Do you want to REFORMAT the records ?      REFORMAT = NO*  (YES or NO)

                      Press PF 16 to terminate SORT.
```

Figure 22-2. SORT Utility Options Screen

The SORT Utility Options screen fields are described as follows:

Field	Description
FUNCTION	Specify the function (SORT or MERGE) that you want to perform. The Sort function sorts one or more files into a single output file. The Merge function merges two or more ordered files into a single, sorted, output file. The default value is SORT.
MEMORY	Specify the amount of main memory that you want allocated to perform the Sort or Merge function. The main memory used affects the efficiency of the operation being performed. For most operations, 128 KB (the default value) is sufficient. If main memory cannot accommodate 128 KB, then the system determines the amount of memory that is actually available and runs the utility in that amount of memory. Enlarging the work area beyond 128 KB to improve Sort or Merge processing time increases the competition with other users for main memory. The maximum amount of available main memory is displayed next to the Memory field. The maximum amount value is dependent upon the size of the system.

Field	Description
MEMORY (cont.)	The optimum amount of main memory for each Sort or Merge operation can only be determined by experimenting with the actual operating environment. If the Sort or Merge operation takes a lot of processing time or a large amount of main memory, you should run the operation in batch mode during the off-peak hours of your system's use so that other users are not affected.
ADDROUT	Specify YES in the Addrout field to produce a single-field, consecutive file of sorted, three-byte records used by RPG II programs. Each record is a positive binary number representing the relative number of a record in the input file. When you perform a sort with the Addrout option, the **only** output is the relative numbers of the records of the sorted file. For an example of the Addrout option, refer to Section 22.2.1. For more details on Addrout files, refer to the *VS RPG II Language Reference*. You cannot use the Addrout field with the Merge function, nor can you combine it with the Keyout field. Addrout can be combined with a Stable sort. The default value for Addrout is NO.
KEYOUT	Specify YES in the Keyout field to produce a single-field, consecutive file. The file is produced by using an indexed key of the input file as the sort criteria. The input file must be indexed. You cannot use a Keyout file with the Merge function, nor can you combine it with an Addrout file. Keyout can be combined with a Stable sort. The default value for Keyout is NO. For an example of the Keyout option, refer to Section 22.2.1.
STABLE	Specify YES in the Stable field to produce an output file in which records with identical sort keys remain in the same order as the input order sequence. The default value for Stable is NO. For an example of the Stable option, refer to Section 22.2.1.
REFORMAT	Specify (YES or NO) whether you want to reformat the record layout of the output file. You can use the Reformat option in conjunction with the Stable. You cannot use the Reformat option with the Merge function, nor can you combine it with a Addrout or Keyout file. The default value for Reformat is NO.

The combinations of program functions and output options for a sorted file are summarized in Table 22-1:

Table 22-1. SORT Utility Function/Option Matrix Table

	Program Function		Output Option			
	SORT	**MERGE**	**ADDROUT**	**KEYOUT**	**STABLE**	**REFORMAT**
ADDROUT	YES	NO	–	NO	YES	NO
KEYOUT	YES	NO	NO	–	YES	NO
STABLE	YES	YES	YES	YES	–	YES
REFORMAT	YES	NO	NO	NO	YES	–

22.2.1 Output Options for a Sorted File

The output options (Addrout, Keyout, and Stable) for a sorted file are illustrated in the following examples. Each record in a sort file is a positive binary number that is intended to be read by an RPG II program; ASCII Display Characters appear if you have the output file displayed at your workstation.

Sample Addrout Output File

The Addrout output file is a consecutive file (consisting of 3-byte records) containing the sequence numbers of the records in the input file. In the following sample, the input is a consecutive file to be sorted on the Code field in ascending order. For example, the name Varkonyi is in Record Number 008, and is listed first in the output file because it has the lowest Code value (1110) in the Sort field. The name Kaplan is in Record Number 003, and is listed second because it has the second lowest Code value (1120) in the Sort field.

Sorted Input File				**Addrout Output File**
Record Number	**Name**	**Code**	**Address**	**Record Sequence After Sort by Code**
001	Busby	2221	Greenwich St	008
002	Isaac	8677	Pawtucket Rd	003
003	Krieger	1120	Roman Rd	005
004	Mayer	3001	E. 67th St	001
005	Metcalfe	2220	Sachem St	007
006	Pascucci	6676	W Haven	004
007	Sterling	2231	Kiel Ave	006
008	Varkonyi	1110	Beacon St	002

NOTE

The SORT utility cannot produce Addrout files from a Merge function or from multiple input files.

Sample KEYOUT Output File

In the following sample, the input is an indexed file with three index keys (Name, Code, and Address). The primary index key is Name, the first alternate key is Code, and the second alternate key is Address. When the input file is sorted, SORT uses the Code field (in ascending order), finds the lowest value in Code, and places the corresponding primary index key (Name) in the Keyout Output File. For example, Varkonyi has the lowest value (1110) in the Code field so that record is accessed and the contents of the primary index key field is displayed first.

Sorted Input File			**Keyout Output File**
Name	**Code**	**Address**	**Name**
Busby	2221	Greenwich St	Varkonyi
Isaac	8677	Pawtucket Rd	Krieger
Krieger	1120	Roman Rd	Metcalfe
Mayer	3001	E. 67th St	Busby
Metcalfe	2220	Sachem St	Sterling
Pascucci	6676	W Haven	Mayer
Sterling	2231	Kiel Ave	Pascucci
Varkonyi	1110	Beacon St	Isaac

Sample STABLE Output File

Input Files 1 and 2 are to be sorted in **descending order** using the **Code field** as the primary sort key. Note the order of the entries that have identical sort keys in the input files. Then note the order in which they are placed in the output file (depending on the Stable option selection).

If you specify YES in the Stable field, the output records with the same sort key values remain in the same input order sequence; whichever record is sorted first is listed first. For example, Spencer in Input File 1 has Code 3001. Mayer in Input File 2 also has Code 3001. When the SORT utility comes across two 3001s, it places the record that it found first (in Input File 1) before the record that it found second (in Input File 2) when writing to the output file.

Input File 1

Name	Code	Address
Crawford	1110	Foster Ave
Johnson	2220	Jericho St
Moyer	2220	Elm St
Purchis	2221	Ritter St
Spencer	3001	Speakman Dr

Input File 2

Name	Code	Address
Busby	2221	Greenwich St
Isaac	8677	Pawtucket Rd
Krieger	1120	Roman Rd
Mayer	3001	E. 67th St
Metcalfe	2220	Sachem St
Pascucci	6676	W Haven
Sterling	2231	Kiel Ave
Varkonyi	1110	Beacon St

Input File 1 sorted with Input File 2 creates this output file when Stable = YES.

Name	Code	Address
Isaac	8677	Pawtucket Rd
Pascucci	6676	W Haven
Spencer	3001	Speakman Dr
Mayer	3001	E. 67th St
Sterling	2231	Kiel Ave
Purchis	2221	Ritter St
Busby	2221	Greenwich St
Johnson	2220	Jericho St
Moyer	2220	Elm St
Metcalfe	2220	Sachem St
Krieger	1120	Roman Rd
Crawford	1110	Foster Ave
Varkonyi	1110	Beacon St

Input File 1 sorted with Input File 2 creates this output file when Stable = NO.

Name	Code	Address
Isaac	8677	Pawtucket Rd
Pascucci	6676	W Haven
Spencer	3001	Speakman Dr
Mayer	3001	E. 67th St
Sterling	2231	Kiel Ave
Busby	2221	Greenwich St
Purchis	2221	Ritter St
Moyer	2220	Elm St
Metcalfe	2220	Sachem St
Johnson	2220	Jericho St
Krieger	1120	Roman Rd
Varkonyi	1110	Beacon St
Crawford	1110	Foster Ave

NOTE

When you specify NO in the Stable field, records are sorted only by the fields that you specify. No specific order is imposed for the sort of records with duplicate sort keys. As a result, your sorted output for the previous STABLE = NO example may differ but only for those fields that were not specified as sort key fields.

22.3 SPECIFYING THE INPUT FILE TO BE SORTED OR MERGED

After you select the program options, the SORT Utility Input File Definition screen (Figure 22-3) appears.

```
***  MESSAGE 0000 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORT
                           TO DEFINE INPUT



Please enter name of the file to be sorted.


INPUT FILE     = ******** in LIBRARY  = ******** on VOLUME   = ******

Is this a SHARED file ?                      SHARED   = NO*  (YES or NO)
Do you want to select input records ?        SELECT   = NO*  (YES or NO)
Do you have more input files ?               MOREFILE = NO*  (YES or NO)


FILE INPUT DEVICE   = DISK    (DISK or TAPE)

If input device is tape, then enter file sequence number     FILESEQ  = 1***
and the maximum number of input records.                     RECORDS  = 1000**
```

Figure 22-3. SORT Utility Input File Definition Screen

NOTE

If you selected the Merge function on the SORT Utility Options screen (Figure 22-2), you can specify up to 20 input files. The SORT Utility Input File Definition screen automatically reappears until you press ENTER on an unaltered screen.

Specify the input file, library, and volume names of the file that you want to process. Also, specify whether you want to open indexed and consecutive input files in Shared mode. To select specific records from a file for use in a Sort or Merge operation, specify YES in the Select field. (Refer to Section 22.3.2 for more information on selecting records.) The Morefile field enables you to specify more than one input file. If you previously selected the Sort function, you can specify only one input file **unless** you specify YES in the Morefile field. The SORT Utility Input File Definition screen fields are described as follows:

Field	Description
INPUT FILE	Specify the file name of the first input file (or next input file, in case of the Merge function) that you want to sort or merge. If the Merge function has been selected, you can specify up to 20 input files. You can specify up to 10 input files with the Sort function. The Input Definition screen automatically appears again until you enter an unaltered screen. There is no default value for Input File.

Field	Description
LIBRARY	Specify the name of the library in which the input file resides. The default value for Library is your INLIB, which is set through the SET Usage Constants function (PF2) on the Command Processor menu or through a Procedure language SET statement.
VOLUME	Specify the name of the volume on which the input file resides. The default value for Volume is your INVOL, which is set through the SET Usage Constants function (PF2) on the Command Processor menu or through a Procedure language SET statement.
SHARED	Specify (YES or NO) whether you want to open consecutive and indexed input files in Shared mode. If you specify YES, the SORT Utility Lock screen appears next and you must specify the fields on that screen. (Refer to Section 22.3.1.) If you specify NO, the SORT utility opens the input file in Input mode. The default value for Shared is NO.
SELECT	Specify (YES or NO) whether you want to select specific input records for use in a Sort or Merge operation. If you specify YES, the Select screen appears next and you must also specify the criteria that you want on that screen. (For additional information on defining Select criteria, refer to Section 22.3.2.) When you specify YES, only records that match the selection criteria that you specify are used to produce the output file. You exit the Select screen by pressing PF16. If you specify NO in the Select field, then all of the records in the input file are processed, and the Select screen is not shown. The default value for Select is NO.
MOREFILE	Specify (YES or NO) whether you want to sort more than one input file (and automatically merge the files) into a single output file. If you specify YES, then you must also specify all input file names, before continuing SORT processing, through additional SORT Input File Definition screens that automatically appear. However, the additional screens do not include the Select and Morefile options, since you need to specify these options only once. When you finish entering the names of files to be merged, do not alter the next SORT Input File Definition screen that appears; just press ENTER. This terminates the file selection process. The default value for Morefile is NO.
FILE INPUT DEVICE	Specify the type of device (DISK or TAPE) on which the input file resides. If you specify DISK, then no further information is required to complete the input definition. If you specify TAPE, then you must also specify the file sequence number in the Fileseq field and the maximum number of input records in the Records field.
FILESEQ	If you specify TAPE in the File Input Device field, then also specify the sequence number of the input file on the tape. The default value for Fileseq is 1.
RECORDS	If you specify TAPE in the File Input Device field, then also specify the approximate number of records that are to be processed in the file. The SORT utility compares the tape file label to the number of Records specified. If you underestimate the number of records in a file, you may have to select Restart. (For further details on Restart, refer to Section 22.5.1.) The smallest possible overestimation is the most efficient. The Records field defaults to 1000.

Field	Description
RECORDS (cont.)	You can find the number of records of any disk file by displaying the file attributes. To display the attributes of a disk file during SORT processing, perform the following procedure:

1. Press the HELP key to suspend SORT processing.
2. Press PF5 (Manage FILES/LIBRARIES)
3. Specify the file, library, and volume names of the file whose attributes you want to display and press ENTER. (You should see a list of files with the cursor positioned at the file that you specified. If you do not, check the names of the file, library, and volume to ensure that they are spelled correctly.)
4. Press ENTER. This displays the attributes of the specified file. The number of records for the specified file is displayed in the lower center portion of the screen.
5. To exit the File Attributes screen, either press PF1 to select another file and display its attributes, or press the HELP key if you want to continue SORT processing. The HELP key returns you to the Command Processor menu (modified, since you have suspended SORT processing).
6. Press PF1 (CONTINUE Processing) to resume SORT processing.

22.3.1 Sorting Files in Shared Mode

If you specify Shared mode for an input file, the SORT Utility Lock screen (Figure 22-4) prompts you to specify (YES or NO) whether a lock should be placed on the file before copying it so that no changes can be made to the file during the sort operation.

```
***  MESSAGE 0000 BY SORT,

                INFORMATION REQUIRED BY  PROGRAM  SORT
                           TO DEFINE LOCK



Please enter the following options to process a file in shared mode.

Should a lock be placed on the file before copying ?   LOCK     = YES (YES, NO)
  If YES, no changes to the file can be made during
  the sort.  If NO, changes to the file can be made.

If LOCK=YES, specify TIMEOUT and BYPASS:

How many seconds should the timeout be ?               TIMEOUT  = 10*
  (If TIMEOUT=NO, sort will wait
   until the file is available).

If the timeout expires, should the file be bypassed ?  BYPASS   = NO* (YES, NO)
```

Figure 22-4. SORT Utility Lock Screen

The SORT Utility Lock screen fields are described as follows:

Field	Description
LOCK	Specify (YES or NO) whether you want to suspend updates to a file that you are displaying. If you lock a file (YES), no changes to the file can occur while you are displaying it. If you specify NO, no lock is placed on the file, and there is no need to specify the Timeout and Bypass options. The default value is YES.
TIMEOUT	Specify a Timeout value (0 to 255 seconds) for a file if Lock = YES. If another user currently holds the file for update, the Timeout field specifies the length of time that the SORT utility will wait to open the file in Shared mode with a lock. The default value is 10 seconds.
BYPASS	Specify (YES or NO) whether you want to skip a file if the Timeout value expires. If BYPASS = YES and the timeout expires, the SORT utility skips the file. If BYPASS = NO and the timeout expires, the Lock screen reappears with the message "FILE filename IN libname ON volname IS HELD BY USER xxx." The default value is NO.

When timeout expiration occurs, you can continue with the Sort operation; redefine the Lock, Timeout, and Bypass options, and press ENTER. The SORT utility also displays a message that informs you of the following options:

- You can skip the particular file on which the timeout occurred by pressing PF1.
- You can terminate the copy operation if you are copying a library or volume, by pressing PF16.

22.3.2 Defining SELECT Criteria

If you specify YES in the Select field on the SORT Utility Input File Definition Screen (Figure 22-3), the SORT utility selects the records to be sorted or merged on the basis of the relationships defined on the SORT Utility Select screen (Figure 22-5). Individual input records are selected by testing them against comparison values that you specify through the SORT Utility Select screen.

```
***  MESSAGE SEL  BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORT
                          TO DEFINE SELECT


Enter record selection criteria below, supplying field position, length, and
  format type (Binary,Char,Decimal,...), test relation (EQ,NE,GT,GE,LT,LE),
  and test value (in quotes) or comparison field position (without quotes).
Set connector to "AND" to continue current criterion; use "OR" to begin
  alternative criterion.      (Use Chars or Decimal numeric for test value.)

   FLDPOS1  = 0001     LENGTH1  = 001     FLDTYP1  = C
   TSTREL1  = EQ       VALUE1   = "P"***************       CONECT1  = AND

   FLDPOS2  = 0003     LENGTH2  = 001     FLDTYP2  = C
   TSTREL2  = EQ       VALUE2   = "A"***************       CONECT2  = ***

   FLDPOS3  = ****     LENGTH3  = ***     FLDTYP3  = C
   TSTREL3  = **       VALUE3   = ******************      CONECT3  = ***

   FLDPOS4  = ****     LENGTH4  = ***     FLDTYP4  = C
   TSTREL4  = **       VALUE4   = ******************      CONECT4  = ***
```

Figure 22-5. Sample SORT Utility Select Screen

For example, if you are processing a payroll file, you may only want to process the records of part-time employees. The records of part-time employees all have the letter "P" in the first byte of each record. In this example, the test criteria is that all input files must equal "P" in Position 1 in order to be processed. If the test criteria matches the record, the record is selected for sorting. Otherwise, a record is not sorted or merged, and is not transferred to the output file.

Besides being tested for a literal data value, a field can also be tested against another field in the same record. For example, you can specify a condition where, if Position 1 equals Position 43, the record can be processed. When processing a file, you can test additional criteria by using the optional CONECTx field (where x is a number). You can specify the complex relations by combining simple relations with the AND connector. Alternative relations are specified with the OR connector.

In the Sample SORT Utility Select screen (Figure 22-5), the user has specified that only those payroll records that indicate **active** part-time employees should be sorted. The test criteria specifies that all input files must equal "P" in Position 1 AND must equal "A" in Position 3 (indicating Active status).

The SORT utility processes test statements in the following ways:

- If Statement 1 is true OR Statement 2 is true, then sort the record.
- If Statement 1 is true AND Statement 2 is true (both must be true), then sort the record.

You can select from the following options to sort or merge records. (The 'x' appended to each field is a numeric value that SORT automatically assigns.) All fields with the same numeric value suffixes are combined to specify a test relationship for data comparisons. For example, FLDPOS1, LENGTH1, FLDTYP1, TSTREL1, VALUE1, and CONECT1 are combined to identify the first selection criteria.

Field	Description
FLDPOSx	Specify the starting field position in the record that is to be used in your test criteria. The first byte of a record occupies Position 1. When building a complex relation, the field positions do not have to be in ascending numeric order. For example FLDPOS1 = 5, FLDPOS2 = 3 is just as valid as FLDPOS1 = 3, FLDPOS2 = 5. There is no default value for FLDPOSx.
LENGTHx	Specify the number of bytes in the field to be tested, beginning with the field position (FLDPOSx) corresponding to LENGTHx. There is no default value for LENGTHx.
FLDTYPx	Specify the data format of the test criterion (set in VALUEx). The default value for FLDTYPx is C. (Sign refers to the plus (+) or minus (−) sign.) The field type options are: C Character data (alphanumeric) B Binary D External decimal, sign trailing in separate byte L Zoned decimal, sign leading in separate byte P Packed decimal, sign trailing included in last byte Z Zoned decimal, sign trailing included in last byte
TSTRELx	Specify the type of comparison to be made between the input data (starting at FLDPOSx) and VALUEx. There is no default value for TSTRELx. The test relationship options are: EQ Equal NE Not equal GT Greater than LT Less than GE Greater than or equal to LE Less than or equal to
VALUEx	Specify the value to be compared with the field in each input record beginning at FLDPOSx and extending for LENGTHx bytes. VALUEx can be a literal data value enclosed in single or double quotes. VALUEx can also be the starting position of another field in the record.

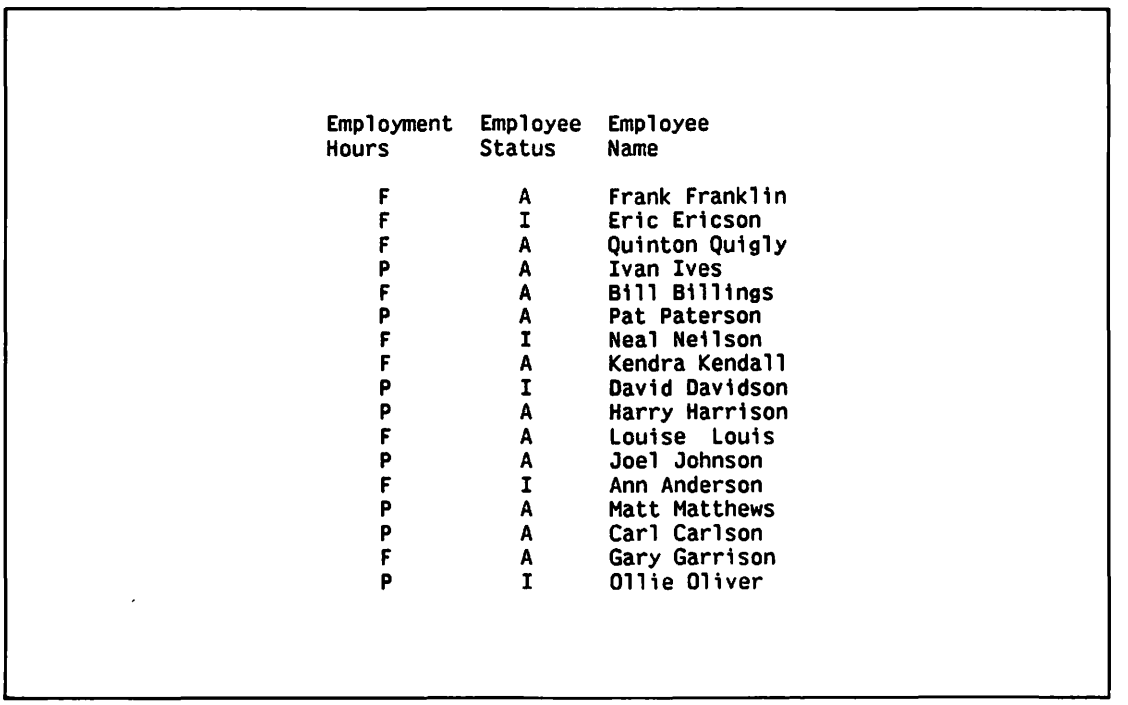
NOTE

You must enclose ***literal*** *data values in quotes, even if you use numbers within your criterion. If you do not type the quotes, VALUEx is treated as a field position. Also be aware of the following restrictions:*

- Eliminating the quotation marks can result in an erroneous output file or an aborted Sort or Merge.
- If you specify the starting position of another field in the record, it must be of the same type and length as the original field. Specifying the incorrect type or length can result in an erroneous output file or an aborted sort or merge.
- VALUEx can be a maximum of 16 bytes for a literal data value or for external, zoned or packed decimal data. VALUEx can be a maximum of 256 bytes for character data in a field-to-field comparison. VALUEx must be two or four bytes for binary data. No default value exists for VALUEx.

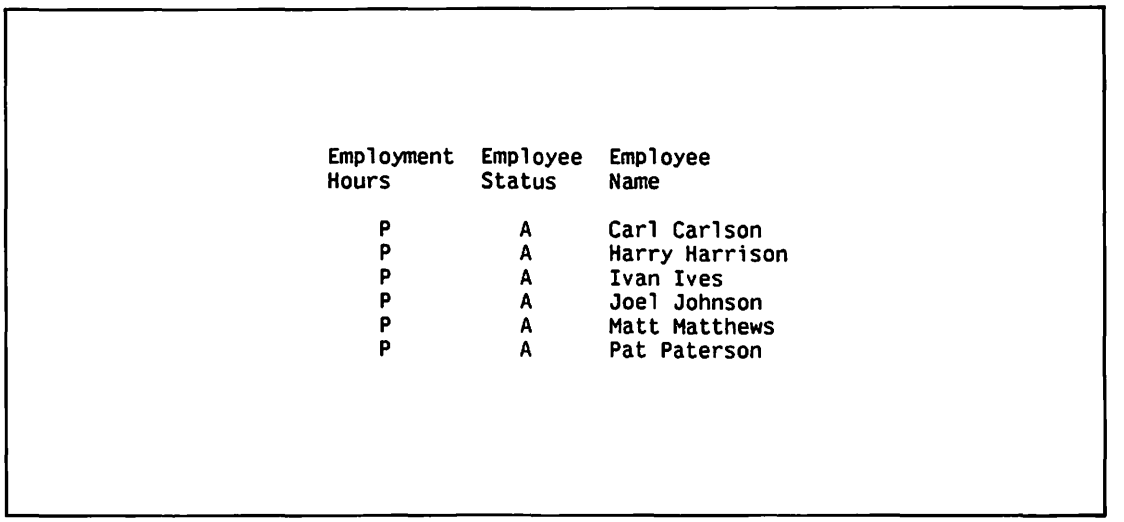
Field	Description
CONECTx	Specify the logical connector (AND or OR) that you want to use between selection criteria. You can specify up to 32 comparisons for each Sort or Merge operation; that is, the maximum number of ANDs and ORs that you can use for each Sort or Merge operation is 31. CONECTx is an optional field. A blank CONECTx field signifies the end of the comparisons in a single criterion. The default value for CONECTx is blank.

Figure 22-6 shows a partial payroll file with part-time and full-time employees who are either active or inactive. Figure 22-7 shows the partial payroll file sorted on the following test criteria: Sort all active part-time employees (FLDPOS1 = 1, LENGTH1 = 1, FLDTYP1 = C, TESTREL1 = EQ, VALUE1 = "P", CONECT1 = AND, FLDPOS2 = 3, LENGTH2 = 1, FLDTYP2 = C, TESTREL2 = EQ, VALUE2 = "A").



Employment Hours	Employee Status	Employee Name
F	A	Frank Franklin
F	I	Eric Ericson
F	A	Quinton Quigly
P	A	Ivan Ives
F	A	Bill Billings
P	A	Pat Paterson
F	I	Neal Neilson
F	A	Kendra Kendall
P	I	David Davidson
P	A	Harry Harrison
F	A	Louise Louis
P	A	Joel Johnson
F	I	Ann Anderson
P	A	Matt Matthews
P	A	Carl Carlson
F	A	Gary Garrison
P	I	Ollie Oliver

Figure 22-6. Partial Payroll File Before Selective Sort



Employment Hours	Employee Status	Employee Name
P	A	Carl Carlson
P	A	Harry Harrison
P	A	Ivan Ives
P	A	Joel Johnson
P	A	Matt Matthews
P	A	Pat Paterson

Figure 22-7. Partial Payroll File After Selective Sort

22.4 DEFINING THE SORT/MERGE KEYS

After you specify the input file and the selection criteria (if any), the Sort/Merge Keys screen (Figure 22-8) appears. From the Sort/Merge Keys screen, you select the data fields by which the file is to be sorted or merged. The fields that you specify are the sort keys, which should be distinguished from the index keys.

```
***  MESSAGE 0001 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORT
                          TO DEFINE KEYS


Please specify Sort/Merge keys:

NUMBER OF KEYS      = 1  (Less than or equal to 8)
Key attributes:
         Position(>0)      Length in Bytes   Sort Key Type    Sort Order(A,D)
Key 1:   POST1    = ****   LENGTH1  = ***    TYPE1     = C    ORDER1    = A
Key 2:   POST2    = ****   LENGTH2  = ***    TYPE2     = C    ORDER2    = A
Key 3:   POST3    = ****   LENGTH3  = ***    TYPE3     = C    ORDER3    = A
Key 4:   POST4    = ****   LENGTH4  = ***    TYPE4     = C    ORDER4    = A
Key 5:   POST5    = ****   LENGTH5  = ***    TYPE5     = C    ORDER5    = A
Key 6:   POST6    = ****   LENGTH6  = ***    TYPE6     = C    ORDER6    = A
Key 7:   POST7    = ****   LENGTH7  = ***    TYPE7     = C    ORDER7    = A
Key 8:   POST8    = ****   LENGTH8  = ***    TYPE8     = C    ORDER8    = A

* Key Type:    B=Binary, C=Character, D=Decimal, F=Floating, P=Packed
               Z=Zoned decimal, L=Zoned decimal sign leading
* Sort Order:  A=Ascending, D=Descending
```

Figure 22-8. Sort/Merge Keys Screen

All fields with the same number are combined to specify an individual Sort/Merge key. For example, POST1, LENGTH1, TYPE1, and ORDER1 are combined to identify Sort/Merge Key 1. To define the Sort/Merge keys, you must specify the following fields (the "x" suffix on some of the fields is a number that the SORT utility automatically assigns):

Field	Description
NUMBER OF KEYS	Specify the number of keys that you want to use for each sort or merge. A maximum of eight keys are allowed.
POSTx	Specify the starting position of the key field within the record. The first byte of a record is designated as Position 1. There is no default value for POSTx.
LENGTHx	Specify the length of the key field in bytes. There is no default value for LENGTHx.
TYPEx	The data type of the key field. The default value for TYPEx is C. (Sign refers to the plus (+) or minus (−) sign.) The options are: C Character data (alphanumeric) B Binary D External decimal, sign trailing in separate byte F Floating point L Zoned decimal, sign leading in separate byte P Packed decimal, sign trailing included in last byte Z Zoned decimal, sign trailing included in last byte
ORDERx	Specify the order in which the output file should be ordered: A (Ascending order — low to high) or D (Descending order — high to low). The default value for ORDERx is A.

If you specify more than one key in the Sort/Merge Keys screen, the SORT utility uses the keys in the order in which they are listed, that is, the sort is performed first on Key 1, then on Key 2, then on Key 3, etc. Using multiple Sort/Merge keys, you can sort using more than one field as your criteria. Similarly, you can use multiple Sort/Merge keys with one field; to sort on the first key specified, with the additional keys being used to sort duplicate values within the same (first) key. After you have specified the Sort/Merge keys that you want, press enter to process the files.

22.5 DEFINING THE OUTPUT FILE

After SORT processes the files, the SORT Utility Output Definition screen (Figure 22-9) appears. From this screen, specify where you would like the sorted output to be placed.

```
***  MESSAGE 0000 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORT
                             TO DEFINE OUTPUT



Please specify the output file below:

OUTPUT FILE     = ******** in LIBRARY  = ******** on VOLUME   = ******

Replace input file with same name?    REPLACE  = NO*    (YES or NO)

Do you want the file compressed?      COMPRESS = NO*    (YES or NO)

FILE OUTPUT DEVICE   = DISK    (DISK or TAPE)

If output device is tape, then enter file sequence number.   FILESEQ  = 1***
```

Figure 22-9. SORT Utility Output Definition Screen

A description of the SORT Utility Output Definition screen fields are as follows:

Field	Description
OUTPUT FILE	Specify the name of the output file to be created. There is no default value for the Output File field. If you specify the same name for the output file as the name for the input file, then a message is displayed, warning you that this action replaces the input file with the sorted output file. This is provided to protect your input file from accidental loss. If you are sure that you want the input file replaced, press PF3 to delete the previous input file. The sorted output file then takes its place.
LIBRARY	Specify the name of the library in which you want to place the output file. The Library value defaults to the OUTLIB value, which is set through the SET Usage Constants function (PF2) on the Command Processor menu.

Field	Description
VOLUME	Specify the name of the volume on which to place the output file. The Volume value defaults to the OUTVOL value, which is set through the SET Usage Constants function (PF2) of the Command Processor menu.
REPLACE	Specify (YES or NO) whether you want the input file to be replaced by the sorted output without first prompting you for confirmation of the replacement. If you specify YES in the Replace field, then the input file is deleted and the sorted output assumes the name of the input file. If you specify NO in the Replace field, then you must specify a name in the Output File field that is different from the name of the input file. The default value for Replace is NO. The Replace option does not appear under the following conditions: • If you selected a Merge operation • If YES is specified in the Morefile field • If the input file is indexed • If the input file is opened in Shared mode • If you are creating an ADDROUT or KEYOUT file
COMPRESS	Specify (YES or NO) whether you want to compress data on the output file. Compression can save disk space. Compress defaults to YES for variable-length input, and to NO for fixed-length input.
FILE OUTPUT DEVICE	Specify the type of device on which the output file is to reside. If the output device is DISK, no further information is required to complete the output definition. If the output device is TAPE, you must specify the file sequence number. The default value for the File Output Device field is DISK.
FILESEQ	Specify the sequence number of the output file on the tape (if any). The default value for Fileseq is 1.

After you define the output, the SORT utility creates the output file and the Command Processor menu appears with a SORT completion message. If the SORT utility was unable to proceed correctly, an error message and a return code appears. (Refer to Appendix B for more information on return codes.)

22.5.1 Restarting the SORT Utility

In some cases, an unsuccessful Sort or Merge occurs. When this happens, SORT utility processing is halted to allow you to restart or terminate the program. For example, the SORT utility will not proceed if the actual number of records in the input file is more than you specified on the SORT Utility Input Definition screen. In this case, an error message is displayed, the SORT utility provides you with an actual record count, and you are given the option either to restart SORT processing, using the actual record count, by pressing PF1, or to terminate SORT processing by pressing PF16 (in which case, an output file is not produced).

If you choose to restart SORT processing, you can do this in two different ways (depending upon whether the input file resides on tape or disk.) If the input file is on tape, you only have to rerun the SORT utility by using the actual record count. No further data entry operations are necessary because SORT saves the previously entered GETPARMs when it issues the Restart message. If the input file is on disk, the problem that caused the unsuccessful sort or merge may have resulted froma prior system failure. You should first reorganize the file through the COPY utility, before retrying the SORT utility. (Refer to Chapter 3 for more information on file reorganization through the COPY utility.)

22.6 A SAMPLE SORT PROCEDURE

You can control SORT processing through the VS Procedure language. You can also specify SORT input and output options, as well as selection criteria and key options. Appendix A contains a complete list of SORT GETPARMs. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

In the sample SORT procedure that follows, all defaults of the SORT Utility Options screen (Figure 22-2) have been accepted. The output file is sorted in descending order. Selection criteria specify that the value of the first byte of the output records must be greater than or equal to the value at Byte 10 of the input records, or greater than a literal value of A.

NOTE

The value in VALUE2 is enclosed in single quotes within double quotes. 'A' is a literal value specified in the Select screen. Because VS Procedure language uses quotation marks as delimiters, both sets of quotes are necessary. If you eliminate one set of quotation marks when writing the SORT procedure, or if both sets of quotes are the same, the procedure returns an error message indicating incorrect punctuation and a return code of 4. (Appendix B contains return code explanations.)

```
PROCEDURE
    RUN SORT
        ENTER OPTIONS
        ENTER INPUT FILE=SAVE3, LIBRARY=DJDCOPY,
            VOLUME=NEWSYS, SELECT=YES
        ENTER SELECT FLDPOS1=1, LENGTH1=1, TSTREL1=GE,
            VALUE1=10, CONECT1=OR, FLDPOS2=1, LENGTH2=1,
            TSTREL2=GT, VALUE2="'A'"
        ENTER KEYS POST1=1, LENGTH1=1, ORDER1=D
        ENTER OUTPUT FILE=FILE1, LIBRARY=PROC1
RETURN
```

CHAPTER 23
THE SORTINT UTILITY

23.1 INTRODUCTION

The SORTINT utility is an international version of the SORT utility. (Refer to Chapter 22 for information about the SORT utility.) You can use SORTINT to sort up to 20 files into a single, ordered output file. The sort can occur using the standard ASCII (American Standard Code for Information Interchange) sequence or an external collating sequence that has been previously defined. You can define an external collating sequence through the TABLEDIT utility (refer to Chapter 24 for more information).

You can also use SORTINT to reformat the output record. SORTINT supports all of the standard features of the SORT utility except for tape input and output, the use of shared consecutive files, and the Merge function. You can perform the following functions with SORTINT:

- Accommodate international character sets or sort orders other than standard ASCII by sorting with an external collating sequence. SORTINT prompts you to specify the name of the file that defines the external collating sequence.
- Sort a single file according to standard ASCII or an external collating sequence.
- Sort up to 20 files into a single, ordered output file according to standard ASCII or an external collating sequence.
- Select specified records from one or more input files and sort them into a single output file according to standard ASCII or an external collating sequence.
- Produce an output file that contains only the primary index key field from each record in the input file. (You specify the sort order for the primary index key field.)
- Produce an ordered output file of 3-byte records to be used by RPG II programs.
- Reformat the record layout of the output file. For example, by reformatting the output record, you can modify the length and sequence of fields in the record or include only selected fields in the record for display, print, or storage.

Figure 23-1 shows an overview of SORTINT processing.

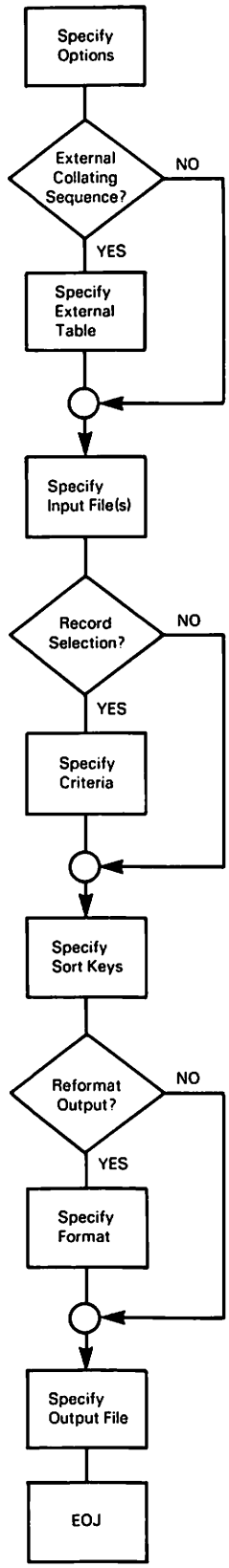


Figure 23-1. SORTINT Processing

NOTES

- *You can access input files (consecutive or indexed) from disk. The SORTINT utility leaves the input file(s) intact unless you assign the same name to the input and output files. The output file is always a consecutive file.*
- *SORTINT requires a 200% overhead in disk space.*
- *You cannot use an output file, sorted according to an external collating sequence, as input to other processes because DMS does not support external collating sequences.*

23.2 SORTINT PROCESSING

To run the SORTINT utility, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify SORTINT in the Program field and press ENTER. When SORTINT processing begins, the SORTINT Options screen appears (Figure 23-2). Specify the function (Sort or Merge) to be performed, the amount of memory to be used for processing the Sort or Merge operation, and the output selection criteria that you need to process the files.

```
***  MESSAGE 0001 BY SORTI

                  INFORMATION REQUIRED BY  PROGRAM  SORTINT
                             TO DEFINE OPTIONS



VS Sort/Merge Program, International Version x.xx.xx (c) Wang 1985
Select Options and Press ENTER to Continue, or Press PF-16 to Exit


            FUNCTION = SORT*              (SORT or MERGE)
            MEMORY   = 128 K   (Memory used, Maximum size = 172 K)


Do you want an ADDROUT output file ?        ADDROUT  = NO*  (YES or NO)
Do rou want a KEYOUT output file ?          KEYOUT   = NO*  (YES or NO)
Do you require a STABLE sort ?              STABLE   = NO*  (YES or NO)
EXTERNAL collating secuence ?               EXTERNAL = NO*  (YES or NO)
REFORMAT output record ?                    REFORMAT = NO*  (YES or NO)
Do you want to SHARE input files ?          SHARED   = NO*  (YES or NO)
```

Figure 23-2. SORTINT Options Screen

The SORTINT Options screen fields are described as follows:

Field	Description
FUNCTION	Specify the function (SORT or MERGE) that you want to perform. The Sort function sorts one or more files into a single output file. The Merge function (not actually supported by the SORTINT utility) is called from the SORT utility and merges two or more ordered files into a single, sorted, output file. The default value is SORT.

Field	Description
MEMORY	Specify the amount of main memory that you want allocated to perform the Sort or Merge function. The main memory used affects the efficiency of the operation being performed. For most operations, 128 KB (the default value) is sufficient. If main memory cannot accommodate 128 KB, then the system determines the amount of memory that is actually available and runs the utility in that amount of memory. Enlarging the work area beyond 128 KB to improve Sort or Merge processing time increases the competition with other users for main memory. The maximum amount of available main memory is displayed next to the Memory field. The maximum amount value is dependent upon the size of your modifiable data area. (Refer to Section 7.13.7 for more information about the modifiable data area.) The optimum amount of main memory for each Sort or Merge operation can only be determined by experimenting with the actual operating environment. If the Sort or Merge operation takes a lot of processing time or a large amount of main memory, you should run the operation in batch mode during the off-peak hours of your system's use so that other users are not affected.
ADDROUT	Specify YES in the Addrout field to produce an ordered output file of 3-byte records that are used by RPG II programs. Each record is a positive binary number that represents the relative number of a record in the input file. When you perform a sort with the Addrout option, the **only** output is the relative number of the records of the sorted file. For an example of the Addrout option, refer to Section 22.2.1. For more details on Addrout files, refer to the *VS RPG II Language Reference*. You cannot use the Addrout field with the Merge function, nor can you combine it with the Keyout function. Addrout can be combined with a Stable sort. The default value for Addrout is NO.
KEYOUT	Specify YES in the Keyout field to produce an ordered output file that consists of the primary index key field in the input file. The input file must be indexed. The file is produced by using an indexed key from the input file as the sort criteria. Keyout can be combined with the Stable or External options. You cannot use a Keyout file with the Merge function, nor can you combine it with an Addrout file. The default value for Keyout is NO. For an example of the Keyout option, refer to Section 22.2.1 in the SORT utility chapter.
STABLE	Specify YES in the Stable field to produce an ordered output file in which records with identical sort keys remain in the same order as the input order sequence. The default value for Stable is NO. For an example of the Stable option, refer to Section 22.2.1 in the SORT utility chapter.
EXTERNAL	Specify (YES or NO) whether you want to use a collating sequence other than ASCII. You cannot use the External option with the Merge function. The default value for External is NO.
REFORMAT	Specify (YES or NO) whether you want to reformat the record layout of the output file. You can use the Reformat option in conjunction with the Stable or External option. You cannot use the Reformat option with the Merge function, nor can you combine it with an Addrout or Keyout file. The default value for Reformat is NO.

Field	Description
SHARED	Specify (YES or NO) whether you want input files to be shared. (The input files must be indexed.) The default value for Shared is NO.

NOTE
If you specify NO to both the External and Reformat options, SORTINT calls the SORT utility to perform the sort operation. Refer to Chapter 22 for a description of SORT processing.

Table 23-1 summarizes the relationship between the program options and the types of output files available.

Table 23-1. SORTINT Function/Option Matrix Table

	Program Function		Output Option				
	SORT	MERGE	ADDROUT	KEYOUT	STABLE	REFORMAT	EXTERNAL
ADDROUT	YES	NO	–	NO	YES	NO	YES
KEYOUT	YES	NO	NO	–	YES	NO	YES
STABLE	YES	YES	YES	YES	–	YES	YES
REFORMAT	YES	NO	NO	NO	YES	–	YES
EXTERNAL	YES	NO	YES	YES	YES	YES	–

23.3 SPECIFYING THE EXTERNAL COLLATING SEQUENCE

If you specify YES in the External field, a screen appears and prompts you for file, library, and volume information. Specify the names and press ENTER. The default value for Library is TRNTABLE, and the default value for Volume is the value defined for INVOL through the SET Usage Constants function (PF2) of the Command Processor menu (or through a Procedure language SET statement). Use the TABLEDIT utility to create or modify files that define external collating sequences (refer to Chapter 24 for more information).

23.4 SPECIFYING THE INPUT FILE

After you select the program options and, if necessary, specify the file that defines the external collating sequence, the SORTINT Input File Definition screen (Figure 23-3) appears. Specify the input file, library, and volume names of the file that you want to process. To select specific records from a file for use in a Sort or Merge operation, specify YES in the the Select field. (Refer to Section 23.4.1 for information about the criteria for selecting records.) The Morefile field enables you to specify more than one input file.

If you previously selected the Sort function, you can specify only one input file **unless** you specify YES in the Morefile field (in which case, you can specify up to 20 input files. The SORTINT Input File Definition screen continues to appear until you press ENTER from an unaltered screen.) After specifying the input file, press ENTER.

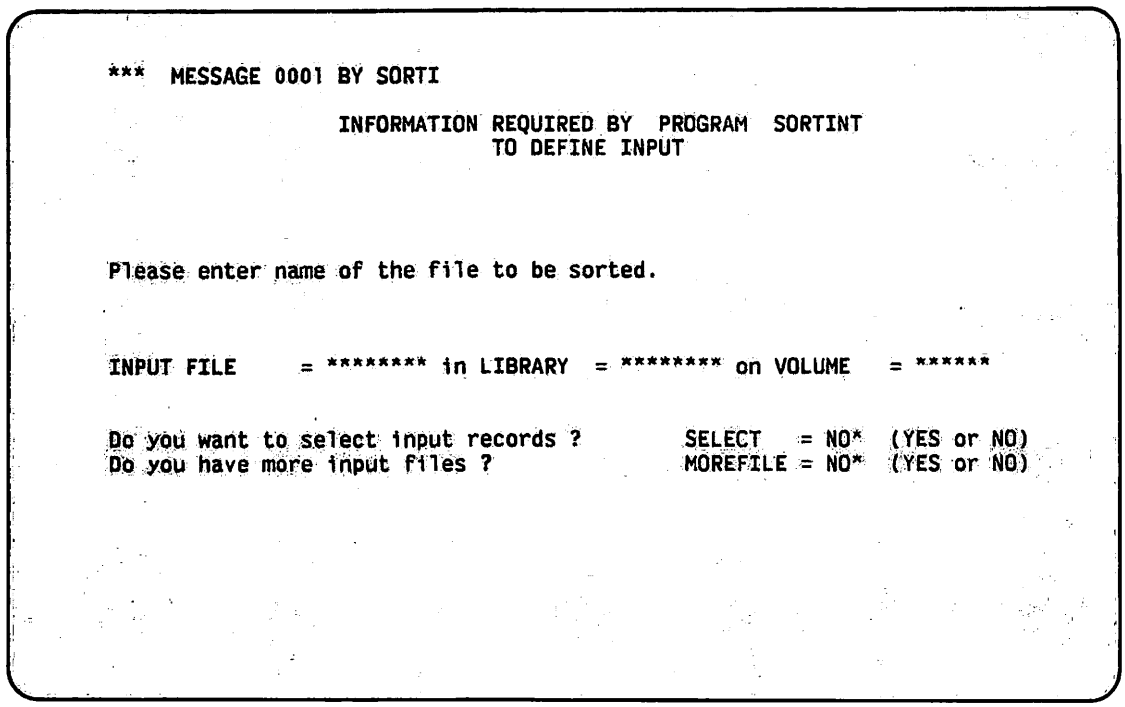


Figure 23-3. SORTINT Input File Definition Screen

The SORTINT input file must be a disk file. You can use the SORT utility to sort tape files (refer to Chapter 22 for more information). Specify the following fields to identify the input file:

Field	Description
INPUT FILE	Specify the file name of the first input file (or next input file, in case of the Merge function) that you want to sort or merge. If the Merge function has been selected, you can specify up to 20 input files. The Input Definition screen automatically appears again until you enter an unaltered screen. There is no default value for Input File.
LIBRARY	Specify the name of the library in which the input file resides. The default value for Library is your INLIB, which is set through the SET Usage Constants function (PF2) on the Command Processor menu or through a Procedure language SET statement.
VOLUME	Specify the name of the volume on which the input file resides. The default value for Volume is your INVOL, which is set through the SET Usage Constants function (PF2) on the Command Processor menu or through a Procedure language SET statement.
SELECT	Specify (YES or NO) whether you want to select specific input records for use in a Sort or Merge operation. If you specify YES, then you must also specify the criteria that you want through the Select screen, which appears next. (For additional information on defining Select criteria, refer to Section 23.4.1.) When you specify YES, only records that match the selection criteria that you specify are used to produce the output file. If you specify NO in the Select field, then all of the records in the input file are processed, and the Select screen is not shown. The default value for Select is NO.

Field	Description
MOREFILE	Specify (YES or NO) whether you want to sort more than one input file (and automatically merge the files) into a single output file. If you specify YES, then you must also specify all input file names, before you continue SORT processing, through additional SORTINT Input File Definition screens that automatically appear. However, the additional screens do not include the Select and More file options, since you need to specify these options only once. When you finish entering the names of files to be merged, do not alter the next SORTINT Input File Definition screen; just press ENTER. This terminates the file selection process. The default value for Morefile is NO.

23.4.1 Defining Selection Conditions

If you specify YES in the Select field on the SORTINT Input File Definition screen (Figure 23-3), the SORTINT utility selects the records to be sorted or merged on the basis of the relationships defined on the SORTINT Utility Select screen (Figure 23-4). Individual input records are selected by testing them against comparison values that you specify through the SORTINT Utility Select screen.

For example, if you are processing a payroll file, you may only want to process the records of part-time employees. The records of part-time employees all have the letter "P" in the first byte of each record. In this example, the test criteria is that all input files must equal "P" in Position 1 in order to be processed. If the test criteria matches the record, the record is selected for sorting. Otherwise, a record is not sorted or merged, and is not transferred to the output file.

Besides being tested for a literal data value, a field can also be tested against another field in the same record. For example, you can specify a condition where, if Position 1 equals Position 43, the record can be processed.

When processing a file, you can test additional criteria by using the optional CONECTx field (where x is a number). You can specify the complex relations by combining simple relations with the AND connector. Alternative relations are specified with the OR connector.

```
***  MESSAGE SEL  BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORTINT
                          TO DEFINE SELECT
                     ACTIVE SUBPROGRAM IS SORT


Enter record selection criteria below, supplying field position, length, and
  format type (Binary,Char,Decimal,...), test relation (EQ,NE,GT,GE,LT,LE),
  and test value (in quotes) or comparison field position (without quotes).
Set connector to "AND" to continue current criterion; use "OR" to begin
  alternative criterion.      (Use Chars or Decimal numeric for test value.)

   FLDPOS1  = ****     LENGTH1  = ***     FLDTYP1  = C
   TSTREL1  = **       VALUE1   = ******************       CONECT1  = ***

   FLDPOS2  = ****     LENGTH2  = ***     FLDTYP2  = C
   TSTREL2  = **       VALUE2   = ******************       CONECT2  = ***

   FLDPOS3  = ****     LENGTH3  = ***     FLDTYP3  = C
   TSTREL3  = **       VALUE3   = ******************       CONECT3  = ***

   FLDPOS4  = ****     LENGTH4  = ***     FLDTYP4  = C
   TSTREL4  = **       VALUE4   = ******************       CONECT4  = ***
```

Figure 23-4. Sample SORTINT Utility Select Screen

In the Sample SORTINT Utility Select screen, the user has specified that only those payroll records that indicate active part-time employees should be sorted. The test criteria specifies that all input files must equal "P" in Position 1 and must equal 'A' in Position 3 (indicating Active status). Note that the test statements in Figure 23-4 are processed by the SORTINT utility in the following ways:

- If Statement 1 is true OR Statement 2 is true, then sort the record.
- If Statement 1 is true AND Statement 2 is true (both must be true), then sort the record.

Refer to Figures 22-6 and 22-7 in Section 22.3.2 of the SORT utility chapter for an illustration of using selection criteria.

You can select from the following options to sort or merge records. (The 'x' appended to each field is a numeric value that SORT automatically assigns.) All fields with the same numeric value suffixes are combined to specify a test relationship for data comparisons. For example, FLDPOS1, LENGTH1, FLDTYP1, TSTREL1, VALUE1, and CONECT1 are combined to identify the first selection criteria.

The fields on the SORTINT Utility Select screen are described as follows:

Field	Description
FLDPOSx	Specify the starting field position in the record that is to be used in your test criteria. The first byte of a record occupies Position 1. When building a complex relation, the field positions do not have to be in ascending numeric order. For example FLDPOS1 = 5, FLDPOS2 = 3 is just as valid as FLDPOS1 = 3, FLDPOS2 = 5. There is no default value for FLDPOSx
LENGTHx	Specify the number of bytes in the field to be tested, beginning with the field position (FLDPOSx) corresponding to LENGTHx. There is no default value for LENGTHx.

Field	Description
FLDTYPx	Specify the data format of the test criterion (set in VALUEx). The default value for FLDTYPx is C. (Sign refers to the plus (+) or minus (−) sign.) The field type options are: C Character data (alphanumeric) B Binary D External decimal, sign trailing in separate byte L Zoned decimal, sign leading in separate byte P Packed decimal, sign trailing included in last byte Z Zoned decimal, sign trailing included in last byte
TSTRELx	Specify the type of comparison to be made between the input data (starting at FLDPOSx) and VALUEx. There is no default value for TSTRELx. The test relationship options are: EQ Equal NE Not equal GT Greater than LT Less than GE Greater than or equal to LE Less than or equal to
VALUEx	Specify the value to be compared with the field in each input record beginning at FLDPOSx and extending for LENGTHx bytes. VALUEx can be a literal data value enclosed in single or double quotes. VALUEn can also be the starting position of another field in the record. **NOTE** *You must enclose* ***literal*** *data values in quotes, even if you use numbers within your criterion. If you do not type the quotes, VALUEx is treated as a field position. Also be aware of the following restrictions:* • *Eliminating the quotation marks can result in an erroneous output file or an aborted Sort or Merge.* • *If you specify the starting position of another field in the record, it must be of the same type and length as the original field. Specifying the incorrect type or length can result in an erroneous output file or an aborted Sort or Merge.* • *VALUEx can be a maximum of 16 bytes for a literal data value or for external, zoned or packed decimal data. VALUEx can be a maximum of 256 bytes for character data in a field-to-field comparison. VALUEx must be two or four bytes for binary data. No default value exists for VALUEn.*
CONECTx	Specify the logical connector (AND or OR) that you want to use between selection criteria. You can specify up to 32 comparisons for each Sort or Merge operation; that is, the maximum number of ANDs and ORs that you can use for each Sort or Merge is 31. CONECTx is an optional field. A blank CONECTx field signifies the end of the comparisons in a single criterion. The default value for CONECTx is blank.

23.5 SPECIFYING THE SORT/MERGE KEYS

After you specify the input file(s) and the selection conditions (if any) and press ENTER, the Sort/Merge Keys screen (Figure 23-5) appears. From the Sort/Merge Keys screen, you select the data fields by which the file is to be sorted or merged. These selected data fields are the Sort/Merge keys. (The fields that you specify are the sort keys which are to be distinguished from the index keys).

```
***  MESSAGE 0001 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORTINT
                          TO DEFINE KEYS
                    ACTIVE SUBPROGRAM IS SORT

Please specify Sort/Merge keys:

NUMBER OF KEYS      = 1  (Less than or equal to 8)
Key attributes:
          Position(>0)      Length in Bytes  Sort Key Type   Sort Order(A,D)
Key 1:    POST1     = ****  LENGTH1  = ***   TYPE1    = C   ORDER1   = A
Key 2:    POST2     = ****  LENGTH2  = ***   TYPE2    = C   ORDER2   = A
Key 3:    POST3     = ****  LENGTH3  = ***   TYPE3    = C   ORDER3   = A
Key 4:    POST4     = ****  LENGTH4  = ***   TYPE4    = C   ORDER4   = A
Key 5:    POST5     = ****  LENGTH5  = ***   TYPE5    = C   ORDER5   = A
Key 6:    POST6     = ****  LENGTH6  = ***   TYPE6    = C   ORDER6   = A
Key 7:    POST7     = ****  LENGTH7  = ***   TYPE7    = C   ORDER7   = A
Key 8:    POST8     = ****  LENGTH8  = ***   TYPE8    = C   ORDER8   = A

* Key Type:     B=Binary, C=Character, D=Decimal, F=Floating, P=Packed
                Z=Zoned decimal, L=Zoned decimal sign leading
* Sort Order:   A=Ascending, D=Descending
```

Figure 23-5. Sort/Merge Keys Screen

You can specify from one to eight sort keys. Specify the number of sort keys in the Number of Keys field (the default value is 1.)

All fields with the same number are combined to specify an individual Sort/Merge key. For example, POST1, LENGTH1, TYPE1, and ORDER1 are combined to identify Sort/Merge Key 1. To define the Sort/Merge keys, you must specify the following field values (the "x" suffix on some of the fields is a number that the SORTINT utility automatically assigns):

Field	Description
NUMBER OF KEYS	Specify the number of keys that you want to use for each sort. A maximum of eight keys are allowed.
POSTx	Specify the starting position of the key field within the record. The first byte of a record is designated as Position 1. There is no default value for POSTx.
LENGTHx	Specify the length of the key field in bytes. There is no default value for LENGTHx.

Field	Description
TYPEx	The data type of the key field. The default value for TYPEx is C. (Sign refers to the plus (+) or minus (−) sign.) The options are: C Character data (alphanumeric) B Binary D External decimal, sign trailing in separate byte F Floating point L Zoned decimal, sign leading in separate byte P Packed decimal, sign trailing included in last byte Z Zoned decimal, sign trailing included in last byte
ORDERx	Specify the order in which the output file should be ordered: A (Ascending order — low to high) or D (Descending order — high to low). The default value for ORDERx is A.

If you specify more than one key on the Sort/Merge Keys screen, the SORTINT utility uses the keys in the order in which they are listed, that is, the sort is performed first on Key 1, then on Key 2, then on Key 3, etc. Using multiple Sort/Merge keys, you can sort using more than one field as your criteria. Similarly, you can use multiple Sort/Merge keys with one field; to sort on the first key specified, with the additional keys being used to sort duplicate values within the same (first) key. After you have specified the Sort/Merge keys that you want, press ENTER to process the files.

23.6 DEFINING THE OUTPUT FILE FORMAT

If you specify YES in the Reformat field on the SORTINT Options screen (Figure 23-2), the SORTINT Output Format Definition screen (Figure 23-6) appears and prompts you to define the format of records for the output file. By reformatting the output record, you can modify the length and sequence of fields in the record, or you can include only selected fields in the record.

```
***  MESSAGE 0001 BY SORTI

                  INFORMATION REQUIRED BY  PROGRAM  SORTINT
                              TO DEFINE FORMAT


Please specify Output file format:


OUTPUT RECORD  LENGTH    = ****       PAD       = 20
        Position(Input)    Lenght(Bytes)      Position(Output)
FLD1:   INPOS1    = ****   LENGTH1  = ***   OUTPOS1  = ****
FLD2:   INPOS2    = ****   LENGTH2  = ***   OUTPOS2  = ****
FLD3:   INPOS3    = ****   LENGTH3  = ***   OUTPOS3  = ****
FLD4:   INPOS4    = ****   LENGTH4  = ***   OUTPOS4  = ****
FLD5:   INPOS5    = ****   LENGTH5  = ***   OUTPOS5  = ****
FLD6:   INPOS6    = ****   LENGTH6  = ***   OUTPOS6  = ****
FLD7:   INPOS7    = ****   LENGTH7  = ***   OUTPOS7  = ****
FLD8:   INPOS8    = ****   LENGTH8  = ***   OUTPOS8  = ****
FLD9:   INPOS9    = ****   LENGTH9  = ***   OUTPOS9  = ****
FLD10:  INPOS10   = ****   LENGTH10 = ***   OUTPOS10 = ****
FLD11:  INPOS11   = ****   LENGTH11 = ***   OUTPOS11 = ****
FLD12:  INPOS12   = ****   LENGTH12 = ***   OUTPOS12 = ****
```

Figure 23-6. Sample SORTINT Output Format Definition Screen

Specify the length of the output record and a padding character (if needed). The output record has a minimum length of 1 byte and a maximum length of 2048 bytes. The padding character must be a valid hexadecimal code. The default padding character is 20 (i.e., blank).

A description of the SORTINT Output Format Definition screen fields are as follows. (The 'x' appended to each field is a numeric value that SORTINT automatically assigns):

Field	Description
INPOSx	Specify the starting position of the field in the input record.
LENGTHx	Specify the length of the field in the output record.
OUTPOSx	Specify the starting position of the field in the output record. (It cannot overlap or exceed the output record length that you defined.)

All fields with the same numeric value are combined to define a single field. For example, INPOS1, LENGTH1, and OUTPUT1 are used to define one field in the output record. You can define up to 12 fields in the output record. After defining the output record length, the padding character, and the fields, press ENTER.

23.6.1 Specifying the Output File

After SORTINT processes the files, the SORTINT Utility Output Definition screen (Figure 23-7) appears. From this screen, specify where you would like the sorted output to be placed.

```
***  MESSAGE 0000 BY SORT

                INFORMATION REQUIRED BY  PROGRAM  SORTINT
                          TO DEFINE OUTPUT
                     ACTIVE SUBPROGRAM IS SORT


Please specify the output file below:

OUTPUT FILE     = ******** in LIBRARY  = ******** on VOLUME   = ******
Replace input file with same name?    REPLACE  = NO*    (YES or NO)
Do you want the file compressed?      COMPRESS = NO*    (YES or NO)

FILE OUTPUT DEVICE   = DISK    (DISK or TAPE)
If output device is tape, then enter file sequence number.   FILESEQ  = 1***
```

Figure 23-7. SORTINT Utility Output Definition Screen

A description of the SORTINT Utility Output Definition screen fields are as follows:

Field	Description
OUTPUT FILE	Specify the name of the output file to be created. There is no default value for Output File. If you specify the same name for the output file as the name for the input file, then a message is displayed, warning you that this action replaces the input file with the sorted output file. This is provided to protect your input file from accidental loss. If you are sure that you want the input file replaced, press PF3 to delete the previous input file. The sorted output file then takes its place.
LIBRARY	Specify the name of the library in which you want to place the output file. The Library value defaults to the OUTLIB value, which is set through the SET Usage Constants function (PF2) on the Command Processor menu.
VOLUME	Specify the name of the volume on which to place the output file. The Volume value defaults to the OUTVOL value, which is set through the SET Usage Constants function (PF2) of the Command Processor menu.

Field	Description
REPLACE	Specify (YES or NO) whether you want the input file to be replaced by the sorted output without first prompting you for confirmation of the replacement. You can replace the contents of the input file with the sorted output (if the output file size is less than or equal to the input file size). If the output file is larger than the input file, then you must specify a new file. If you specify YES in the Replace field, then the input file is deleted and the sorted output assumes the name of the input file. If you specify NO in the Replace field, then you must specify a name in the Output File field that is different from the name of the input file. The default value for Replace is NO. The Replace option does not appear under the following conditions: • If you selected a Merge operation • If there is more than one input file • If the input file is indexed • If the input file is opened in Shared mode • If you are creating an ADDROUT or KEYOUT file. **NOTE** *The Replace option is especially useful for running the SORTINT utility as a background task through a procedure since you do not need to specify any output file fields. Refer to the* VS Procedure Language Reference *for more information on writing and running procedures.*
COMPRESS	Specify (YES or NO) whether you want to compress data on the output file. Compression can save disk space. The Compress field defaults to YES for variable-length input, and to NO for fixed-length input.

After you specify the output file and press ENTER, SORTINT creates the output file and then displays the Command Processor menu. A SORTINT completion message appears on the Command Processor menu screen. If SORTINT cannot proceed correctly, an error message and a return code appear on the screen. (Refer to Appendix B for a list of the SORTINT return codes.)

23.7 RESTARTING THE SORTINT UTILITY

In some cases, an unsuccessful sort occurs. When this happens, SORTINT utility processing is halted to allow you to restart or terminate the program. For example, the SORTINT utility will not proceed if the actual number of records in the input file is more than you specified on the SORTINT Utility Input Definition screen. In this case, an error message is displayed, the SORTINT utility provides you with an actual record count, and you are given the option either to restart SORTINT processing, using the actual record count, by pressing PF1, or to terminate SORTINT processing by pressing PF16 (in which case, an output file is not produced).

If you choose to restart SORTINT processing, you can do this in two different ways (depending upon whether the input file resides on tape or disk.) If the input file is on tape, you only have to rerun the SORT utility by using the actual record count. No further data entry operations are necessary because SORTINT saves the previously entered GETPARMs when it issues the Restart message. If the input file is on disk, the problem that caused the unsuccessful sort or merge may have resulted from a prior system failure. You should first reorganize the file through the COPY utility, before retrying the SORT utility. (Refer to Chapter 3 for more information on file reorganization through COPY.)

23.8 A SAMPLE SORTINT PROCEDURE

You can control SORTINT processing through the VS Procedure language. You can also specify SORTINT input and output options, as well as selection criteria and key options. Appendix A contains a complete list of SORTINT GETPARMs. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

The following procedure illustrates the use of the External and Reformat options. The output file is sorted in descending order. The selection condition is that an input record is greater than or equal to 'A' in Position 1.

NOTE
The value in VALUE1 is enclosed in single quotes within double quotes. 'A' is a literal value specified in the Select screen. Because VS Procedure language uses quotation marks as delimiters, both sets of quotes are necessary. If you eliminate one set of quotation marks when writing the SORT procedure, or if both sets of quotes are the same, the procedure returns an error message indicating incorrect punctuation and a return code of 4. (Appendix B contains return code explanations.)

```
PROCEDURE
    RUN SORTINT
        ENTER OPTIONS EXTERNAL = YES, REFORMAT = YES
        ENTER INTABLE FILE = SAVEX, LIBRARY = MAMCOPY,
            VOLUME = SYSTEM, SELECT = YES
        ENTER INPUT FILE = BROKER, LIBRARY = MAMDATA,
            VOLUME = SYSTEM, SELECT = YES
        ENTER SELECT FLDPOS1 = 1, LENGTH1 = 1, TSTREL1 = GE,
            VALUE1 = "'A'"
        ENTER KEYS POST1 = 1, LENGTH = 1, ORDER1 = D
        ENTER FORMAT LENGTH = 100, PAD = 20, INPOS1 = 1,
            LENGTH1 = 10, OUTPOS1 = 1, INPOS2 = 7,
            LENGTH2 = 10, OUTPOST2 = 11, INPOS3 = 20,
            LENGTH3 = 1, OUTPOST3 = 30
        ENTER OUTPUT FILE = SAVEZ, LIBRARY = MAMSORT,
            VOLUME = SYSTEM
RETURN
```

CHAPTER 24
THE TABLEDIT UTILITY

24.1 INTRODUCTION

The TABLEDIT (Table Edit) utility enables you to create or modify collating sequence table files and case flip table files. A collating sequence table enables you to define a specific sort order or international character set other than the standard American Standard Code for Information Interchange (ASCII) character set. After creating a collating sequence table, you can direct the SORTINT utility (refer to Chapter 23 for more information) to sort files according to that table.

A case flip table (similar to a character translation table) is used by a software application to translate all lowercase letters to all uppercase letters. A character translation table enables you to define characters that you want translated into other characters that you specify (i.e., it is not just an alphabetic translation). Although this chapter deals only with the alphabetic case flip function, the TABLEDIT case flip table option enables you to edit any character translation table.

Figure 24-1 shows an overview of TABLEDIT processing.

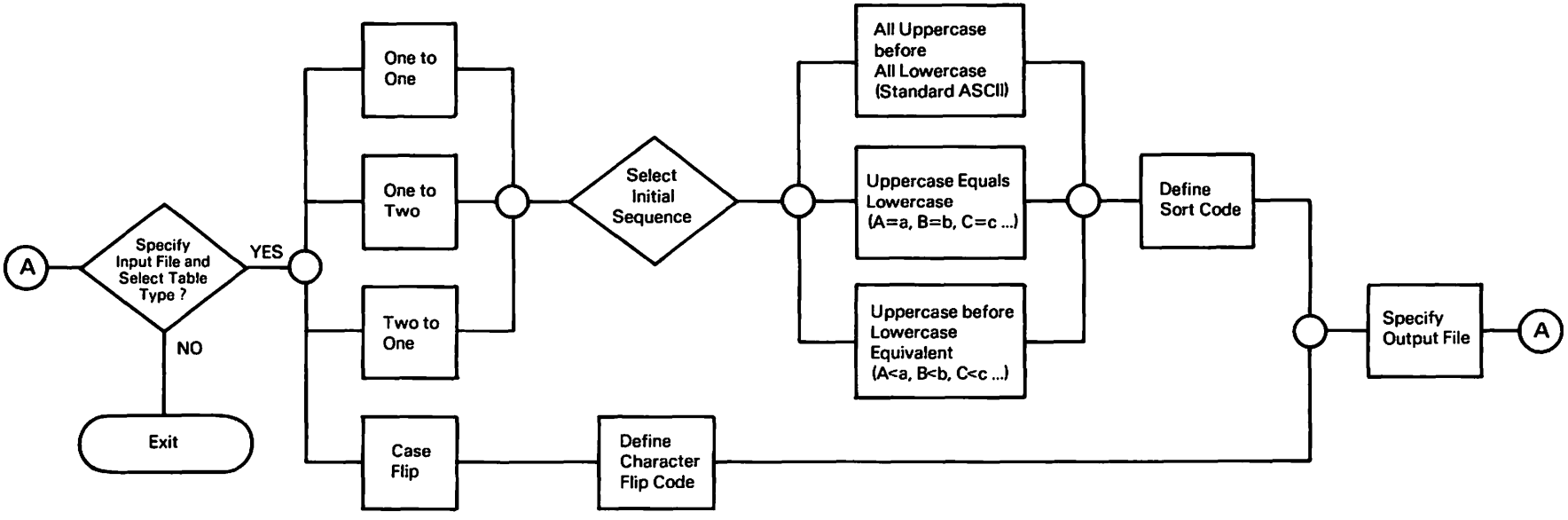


Figure 24-1. TABLEDIT Processing

24.2 SPECIFYING THE INPUT FILE

When TABLEDIT processing begins, the TABLEDIT Input Definition screen (Figure 24-2) appears. From the TABLEDIT Input Definition screen you can create a new collating sequence table or case flip table, or you can edit an existing table.To exit from the TABLEDIT utility, press PF16.

```
TABLEDIT (c) Copyright, 1985 Wang Laboratories, Inc          Select Table Type
Wang VS Table Modification Program, Version x.x                 PrName = TYPE

_____________________________________________________________________________

To edit an existing table, specify the file, library and volume.
To create a new table, leave the fields blank.

The existing table is in      File = ********
                           Library = ********
                            Volume = ******

Select the type of table by pressing the appropriate PF Key:

          (1)  Collating Sequence table - 1 to 1 only
          (2)  Collating Sequence table - 1 to 2 equivalences
          (3)  Collating Sequence table - 2 to 1 equivalences
          (4)  Case-Flip table

_____________________________________________________________________________


                                                              (16) Exit
```

Figure 24-2. TABLEDIT Input Definition Screen

To create a new table, leave the fields blank and press the program function (PF) key that corresponds to the table that you want to create.

To edit an existing table, specify the following fields:

Field	Description
File	Specify the name of the the file that contains the collating sequence or case flip table that you want to edit.
Library	Specify the name of the library in which the table file exists.
Volume	Specify the name of the volume on which the table file exists.

24.2.1 Initial Sequence Selection

If you are creating a new collating sequence table, the TABLEDIT Initial Sequence Selection screen (Figure 24-3) appears. However, if you are editing an existing table or creating a case flip table, this screen is bypassed.

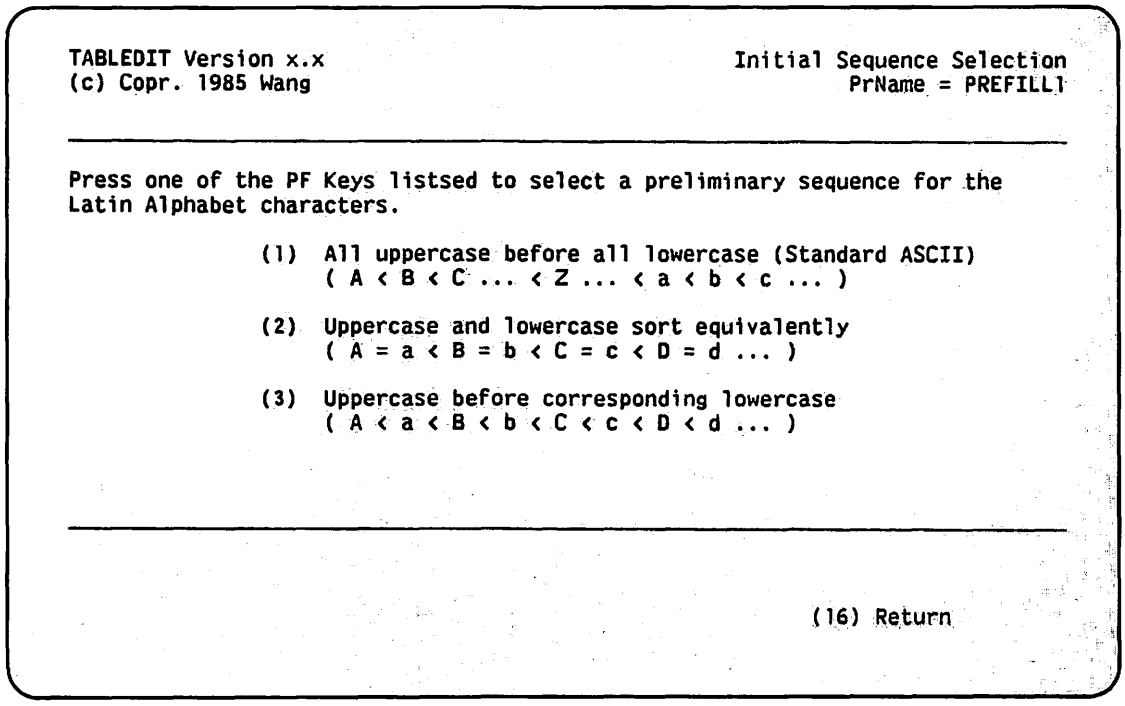


Figure 24-3. TABLEDIT Initial Sequence Selection Screen

The TABLEDIT Initial Sequence Selection screen enables you to set up a preliminary collating sequence from which you can modify the new table. Press the PF key that corresponds to the initial sequence that you want for the table that you specified. To return to the TABLEDIT Input Definition screen, press PF16 (Return).

24.3 EDITING THE COLLATING SEQUENCE TABLE

Whether you are creating or editing a collating sequence or case flip table, the TABLEDIT Edit Table screen (Figure 24-4) appears and enables you to modify the table to your specifications. This screen varies slightly depending on the type of table that you are editing. The differences are discussed in this section.

```
TABLEDIT Version x.x                                          Edit the Table
(c) Copr. 1985 Wang                                            PrName = EDIT

**FILE** in **LIB*** on **VOL* ( One to Two table with 04 equivalences )

   CRT      SORT  !  CRT      SORT  !  CRT      SORT  !  CRT      SORT
   CHAR HEX CODE  !  CHAR HEX CODE  !  CHAR HEX CODE  !  CHAR HEX CODE
                  !                 !                 !
    H   48  4F    !   P   50  5F    !   X   51  **    !   ◄   60  7A
    I   49  51    !   Q   51  61    !   Y   52  **    !   a   61  42
    J   4A  53    !   R   52  63    !   Z   53  **    !   b   62  44
    K   4B  55    !   S   53  65    !   [   54  **    !   c   63  46
    L   4C  57    !   T   54  67    !   \   55  **    !   d   64  48
    M   4D  59    !   U   55  69    !   ]   56  **    !   e   65  4A
    N   4E  5B    !   V   56  6B    !   ^   57  **    !   f   66  4C
    O   4F  5D    !   W   57  6D    !   _   58  **    !   g   67  4E
                  !                 !                 !

ENTER Summary     (5) Right         (9) Save
  (2) First       (6) Increment --                    (14) Show One to Two
  (3) Last        (7) Define
  (4) Left        (8) Delete                          (16) Cancel
```

Figure 24-4. Sample TABLEDIT Edit Table Screen

24.3.1 Introduction to Editing a Table

The characters of a table are displayed in columns of three attributes each in the initial sequence that you selected. The first attribute is the CRT Char (cathode ray tube character) column which displays the character as it appears on the workstation. The second attribute is the Hex (hexadecimal value) column which displays the hexadecimal equivalent to the CRT character. The third attribute is the Sort Code column (or Case Flip column for case flip tables) which displays a default of the hexadecimal equivalent of the CRT character.

The Sort Code column enables you to specify the sort order of a corresponding character. The Case Flip column enables you to specify the character to which you want to flip from the corresponding character that is displayed in the CRT Char field. Case flip character definitions are treated differently than collating sequence characters and are discussed later in this section.

You have the following options from the TABLEDIT Edit Table screen:

PF Key	Function	Description
ENTER	Summary	View an on-line illustration of the collating sequence table that you are defining or editing. (This key does not apply to case flip tables.)
2	First	Display the first set of characters in the table (where applicable).
3	Last	Display the last set of characters in the table (where applicable).
4	Left	Move the table display two panels to the left (where applicable).
5	Right	Move the table display two panels to the right (where applicable).
6	Increment	Add 1 to the values in the Sort Code column (from the sort code that you specify to the last character in the table).

PF Key	Function	Description
7	Define	Define a specific two-to-one equivalence (ch > c) or one-to-two equivalence (ä = ae), depending on the sequence table that you are editing. For more information, refer to Sections 24.3.5 and 24.3.6.
8	Delete	Delete a specific two-to-one equivalence (ch > c) or one-to-two equivalence (ä = ae), depending on the sequence table that you are editing.
9	Save	Save the edited table. Another screen (refer to Section 24.4) enables you to assign this table to a file.
14	Show One-to-Two	Display the one-to-two (or two-to-one) equivalences that have been defined for this collating sequence table.
16	Cancel	Cancel the editing session with this table. No changes are saved and the TABLEDIT Input Definition screen reappears.

The Increment Key

Use PF6 (Increment) when you want to define a unique sort code for a character and the place where you want that character sorted does not have an open sort code value. By specifying a hexadecimal number and pressing PF6, you create a gap in the hexadecimal codes into which you can insert the character that you are defining.

For example, if you wanted to give *ä* (hex 15) a unique value between lowercase a (hex 61) and lowercase b (hex 62) (hexadecimal values are in standard ASCII), specify 62 in the Increment field and press PF6. Hexdecimal values of 62 or greater are incremented by one, leaving the 62 Sort Code value open. After the Sort Code values are incremented, define *ä* to have a Sort Code value of 62. If you assigned a 62 to *ä* before incrementing, the *ä* would be equivalent to b.

24.3.2 The TABLEDIT Display Table Screen

When you press ENTER from the TABLEDIT Edit Table screen, the TABLEDIT Display Table screen (Figure 24-5) appears. (This function does not apply to case flip tables.) The TABLEDIT Display Table screen is an illustration of the collating sequence table that you are defining.

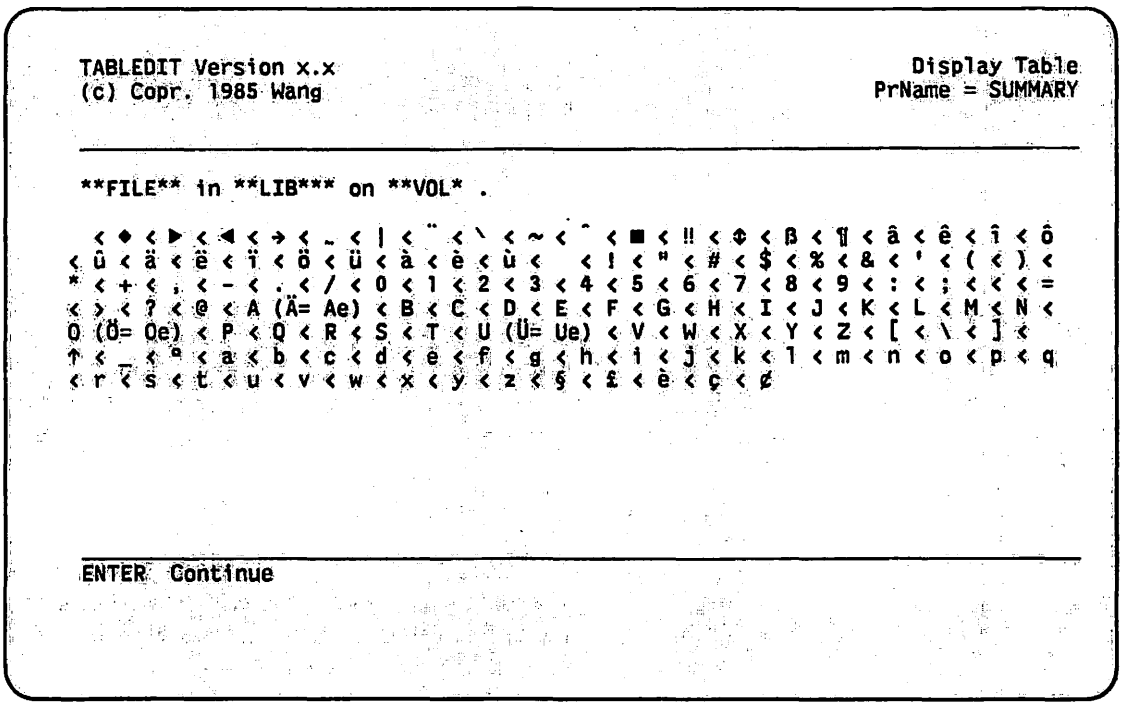


Figure 24-5. Sample TABLEDIT Display Table Screen

Any characters in a collating sequence table that are assigned the same Sort Code are considered equal by the SORTINT utility (refer to Chapter 23 for more information about SORTINT). For example, an asterisk (*) with a Sort Code of 4D is treated the same as an M (if M's Sort Code remained as 4D). The TABLEDIT Display Table screen displays the characters in a relational string with an equal sign (=) or a less than sign (<) between each character. Each character in the table is highlighted for clarity while the relational symbols are displayed at normal intensity.

24.3.3 The TABLEDIT Display Table Sort Precedence

The TABLEDIT Display Table is organized in a left to right sort precedence. Characters on either side of an equal sign have the same Sort Code value. Characters to the left of a less than sign are sorted before characters to the right of that sign. For example, the string "A = a < B = b" means that A and a have no sort precedences over each other but both A and a are sorted before B and b. B and b also have no sort precedence over each other. When there is no sort precedence among character cases (for example, richard = RICHARD), any preceding occurrences (richard) are placed before any following occurences of equal character value (RICHARD) when the collating table is used to sort data.

One-to-two collating sequence tables have some characters displayed in parentheses on the TABLEDIT Display Table screen. The characters that are displayed within the parentheses are sorted somewhere between the values to the left and the right of that character. For example, if you define "a-umlaut" (*ä*) to be sorted as ae on a one-to-two collating sequence table (which is to be sorted after ad and before af), the following result is displayed: "a (*ä* = ae) < b". If you want to see exactly where a character within parentheses lies in the collating sequence, press PF14 (Show One to Two) from the TABLEDIT Edit Table screen (refer to Figure 24-4).

24.3.4 Defining Sort Codes

When you select an initial sequence, the TABLEDIT utility sets the Sort Code column in the TABLEDIT Edit Table screen according to the initial sequence that you selected. The initial sequence that you select affects the Sort Code values. To define ä = ae, look for the Sort Code value in the TABLEDIT Edit Table screen for the letters a and e.

It is important that you use the Sort Code value of a character when you are defining a character equivalence and not the character's Hex column value. For example, if you selected the All Uppercase before All Lowercase (standard ASCII) initial sequence, the Sort Codes for these characters are as follows: a = 61, e = 65. If you selected the Uppercase and Lowercase Equivalence initial sequence, the Sort Codes for these characters are as follows: a = 41, e = 45. If you selected the Uppercase before Corresponding Lowercase initial sequence, the Sort Codes for these characters are as follows: a = 42, e = 4A.

24.3.5 The One-to-One Collating Sequence Table

The one-to-one collating sequence table is the simplest table. To modify the one-to-one collating sequence table, define the Sort Code for each character in the table according to your specifications; simply specify the sort order number (in hexadecimal) in the Sort Code field. For example, if you wanted the numbers 0 – 9 sorted after the letters A – Z and a – z (standard ASCII), specify a value for the number 0 in the Sort Code field that is greater than 7A (the hexadecimal value for z).

24.3.6 The One-to-Two Collating Sequence Table

To modify the one-to-two collating sequence table, define the Sort Code for each character in the table according to your specifications. To specify a one-to-two relationship, press PF7 (Define One-to-Two).

When you press PF7, the bottom region of the screen changes. Specify the character that you want sorted as two characters in the CRT Char or Hex fields; you can specify the character either way. If you specify both fields and the CRT character does not agree with the hexadecimal value that you provided, an error message appears, in which case you must respecify one of the fields. Next, specify the hexadecimal Sort Code value of the two characters that you want to be equal to the one character just specified. After you have specified the one-to-two translation, press ENTER.

For example, ä has a hexadecimal value of 15. Specify 15 in the Hex field. If you want ä to be sorted as ae (standard ASCII), specify 6165 in the Sort Code field, and then press ENTER.

24.3.7 The Two-to-One Collating Sequence Table

To modify the two-to-one collating sequence table, define the Sort Code for each character in the table according to your specifications as you would in a one-to-one table. To specify a two-to-one relationship, press PF7 (Define Two-to-One).

When you press PF7, the bottom region of the screen changes. Specify the two characters that you want sorted as one character in the CRT Char or Hex fields; you can specify the two characters either way. If you specify both fields and the CRT characters do not agree with the hexadecimal values that you provided, an error message appears, in which case you must respecify one of the fields. Next, specify the hexadecimal Sort Code value of the characters that you want to be equal to the two characters just specified. After you have specified the two-to-one translation, press ENTER.

For example, ch has a hexadecimal value of 6368. Specify either ch in the CRT Char field or 6368 (standard ASCII) in the Hex field (or both). If you want ch to be sorted as c, specify 63 (standard ASCII) in the Sort Code field, then press ENTER. After you press ENTER, the TABLEDIT Display Table screen appears.

However, if you want ch to be sorted between c and d, you must first provide a Sort Code space between c and d; that is, increment all the characters by 1, whose Sort Codes are 64 or greater (since 64 is the Sort Code for d using standard ASCII), using PF6 (Increment) from the TABLEDIT Edit Table screen (refer to Section 24.3.1). In this case d would now have a Sort Code of 65. Press PF7 to define a two-to-one relationship. Specify ch in the CRT Char field (or 6368 in the Hex field), 64 in the Sort Code field, and press ENTER. If you press ENTER from the TABLEDIT Edit Table screen after this operation, "a < b < c < ch < d" appears on the screen. The corresponding Sort Codes would be 61 (a), 62 (b), 63 (c), 64 (ch), and 65 (d).

24.3.8 The Case Flip Table

Unlike any of the collating sequence tables, the case flip table does not have a sort order. However, it is easier to define. The three columns in the case flip table are the CRT Char, the Hex, and the Case Flip columns. The CRT Char and Hex columns display a character on the workstation screen along with its corresponding hexadecimal value. The Case Flip column displays a hexadecimal blank (20) value for each character. A case flip table is used by a software application to translate all lowercase letters to all uppercase letters or to translate one character into another character.

To modify the case flip table, specify the hexadecimal value of the character to which you want the corresponding character to be changed. For example, to translate an *ä* to an a, find *ä* in the table (hexadecimal 15) and specify 61 in the Case Flip column entry that corresponds to *ä*.

24.4 SPECIFYING THE OUTPUT FILE

After you have defined the table, press PF9 to save the modifications. When you press PF9, the TABLEDIT Output Definition screen (Figure 24-6) appears. To return to the TABLEDIT Edit Table screen before saving the table, press PF16.

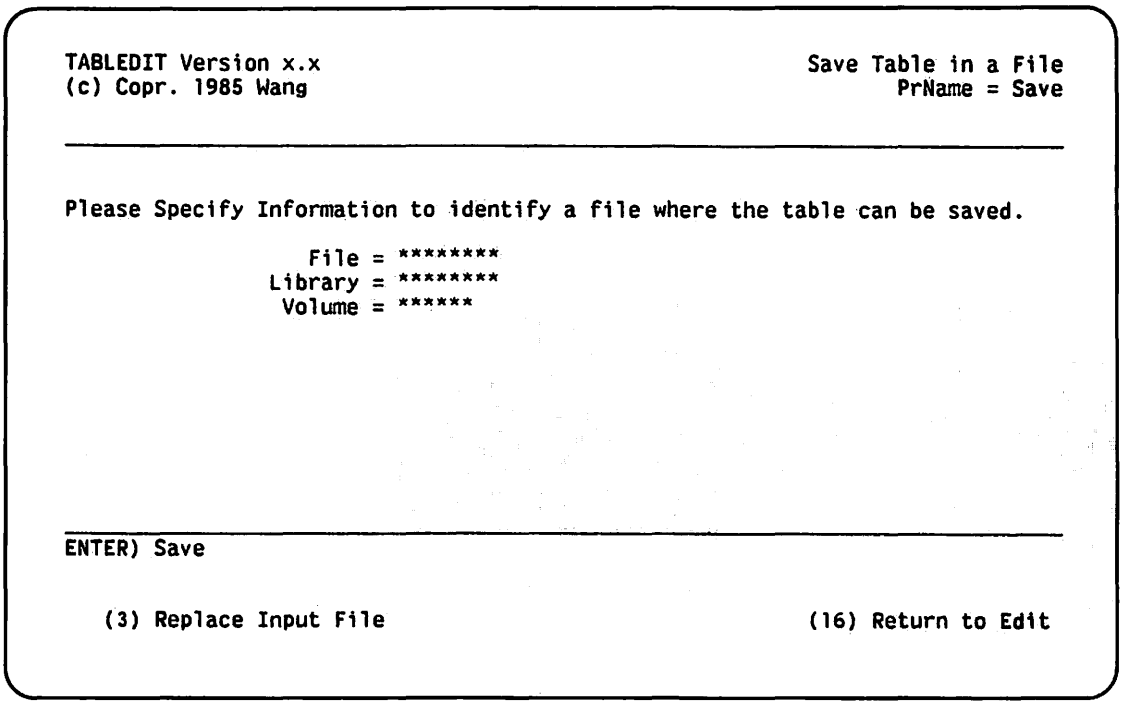


Figure 24-6. TABLEDIT Output Definition Screen

If you specified an existing table on the TABLEDIT Input Definition screen, you can replace the existing table with the modified table by pressing PF3. Otherwise, specify a new file in which to store the modified table and press ENTER.The TABLEDIT Output Definition screen fields are described as follows:

Field	Description
File	Specify the name of the file in which you want to save the table that you just defined.
Library	Specify the name of the library in which the table file is to exist.
Volume	Specify the name of the volume on which the table file is to exist.

24.5 A SAMPLE TABLEDIT PROCEDURE

You can control TABLEDIT processing through the VS Procedure language. You can also specify TABLEDIT input and output options, as well as selection criteria and key options. Appendix A contains a complete list of TABLEDIT GETPARMs. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

This procedure copies an existing table file (in TABLEFILE) into another file (SORTTAB) but states that the SORTTAB table is to have uppercase and lowercase equivalences (A = a < B = b ...). TABLEFILE may or may not have had this collating sequence.

```
PROCEDURE
    RUN TABLEDIT
        ENTER TYPE FILE = TABLEFILE, LIBRARY = TABLELIB,
            VOLUME = TABVOL
        ENTER PREFILL 2
        ENTER EDIT 9
        ENTER SAVE FILE = SORTTAB, LIBRARY = SORTLIB,
            VOLUME TABVOL
        ENTER TYPE 16
RETURN
```

CHAPTER 25
THE TAPECOPY UTILITY

25.1 INTRODUCTION

The TAPECOPY utility performs copy functions, using tape files as the input/output (I/O) files. It also can be used to transfer disk files to tape. You can perform the following functions with the TAPECOPY utility:

- Copy files (including word processing (WP) documents and IBM files) from tape to tape, tape to disk, disk to tape, and disk to disk. TAPECOPY is not recommended for disk-to-disk copying, even though it is possible. The COPY utility is more efficient for disk-to-disk copying.
- Invoke the TRANSL utility to translate file contents to or from the Extended Binary Coded Decimal Interchange Code (EBCDIC) and the American Standard Code for Information Interchange (ASCII).
- Invoke the TRANSL utility to manipulate the fields within a record that you are copying.

Figure 25-1 shows an overview of TAPECOPY processing.

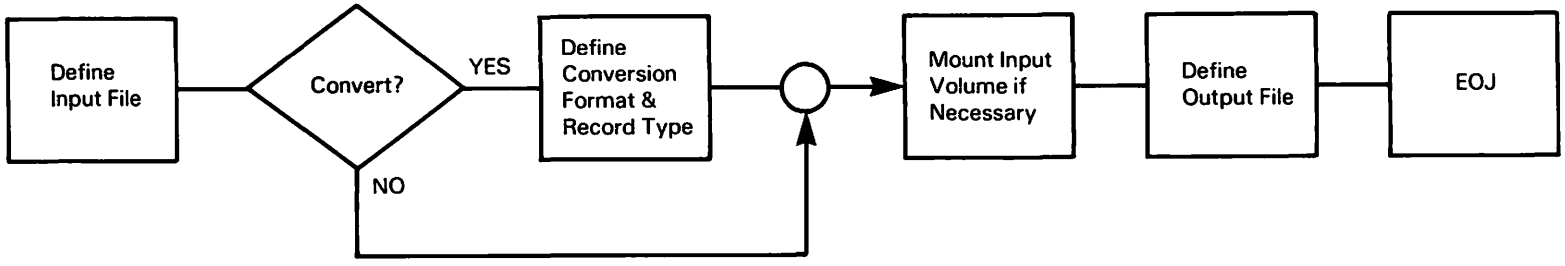


Figure 25-1. TAPECOPY Processing

25.2 DEFINING THE INPUT FILE, TAPE, OR DISK

To begin TAPECOPY processing, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify TAPECOPY in the Program field and press ENTER. When TAPECOPY processing begins, the TAPECOPY Input Definition screen (Figure 25-2) appears.

```
***  MESSAGE 0001 BY TPCOPY

                INFORMATION REQUIRED BY  PROGRAM  TAPECOPY
                            TO DEFINE INPUT


Wang VS Tapecopy Program Version  x.xx.xx

Please identify the input file:
    FILE      = ********   LIBRARY  = DOCMNTR*   VOLUME   = WP****
    DEVICE    = TAPE****
    CONVERT   = N*     N=No Conversion, E=EBCDIC to ASCII, A=ASCII to EBCDIC
                       BA=BCD Format A to ASCII, AB=ASCII to BCD Format A
                       HA=BCD Format H to ASCII, AH=ASCCI to BCD Format H
    MULTYPE   = N      Y=Multiple record types, N=Single record type

If input is from a tape file, please specify:
    FSEQ      = ***1   File sequence number
    LABEL     = AL     AL=ANSI Label, IL=IBM Label, NL=No Label

IBM DOS Tape:
    Does the file contain an HDR2 label?  HEADER2  = YES  (YES or NO)


                Press PF16 to terminate the TAPECOPY program.
```

Figure 25-2. Sample TAPECOPY Input Definition Screen

The fields for the TAPECOPY Input Definition screen are described as follows:

Field	Description
FILE, LIBRARY, and VOLUME	Specify a disk or tape file by entering the file, library and volume names in the input fields. Tape files can also be selected by specifying the tape volume name and the file sequence (Fseq) number. If you enter the file and library names in addition to the tape volume name and file sequence number, TAPECOPY positions the tape according to the Fseq number and then verifies the file name corresponding to that Fseq number.
DEVICE	Specify TAPE or DISK as the storage medium that contains the file(s) that you want copied. The default value is TAPE.

Field	Description
CONVERT	Specify the character set conversion option. The default value is N. N — No conversion E — EBCDIC to ASCII conversion A — ASCII to EBCDIC conversion BA — BCD Format A to ASCII conversion AB — ASCII to BCD Format A conversion HA — BCD Format H to ASCII conversion AH — ASCII to BCD Format H conversion For transactions to BCD Format A (AB conversion option), the ASCII-coded plus sign, left parenthesis, right parenthesis, equals sign, and apostrophe are converted to blanks due to the definition of the A Format. Data containing these characters should be converted through the BCD Format H option.
MULTYPE	Specify the record type. A value of N indicates that the file contains a single record type. A value of Y indicates a multiple record type file. The default value is N. This field must be specified when you invoke the TRANSL utility.

NOTE

The Convert field invokes the TRANSL utility when any value (except N) is selected; the TRANSL Field Options screen (Figure 27-4) then appears (refer to Chapter 27).

If Multype = Y, the TRANSL Record Types screen (Figure 27-3) is displayed, which prompts you to identify characters for each record type.

Refer to Chapter 27 for more information about the TRANSL utility.

If the input is from a tape file, you must specify the file sequence number and label type. The following is a description of the FSEQ and LABEL fields:

Field	Description
FSEQ	Specify the sequence number of the file on the tape. TAPECOPY uses it to position the tape and check the file name against the label. If you do NOT supply the file sequence number for a labeled tape (FSEQ = 0), TAPECOPY uses the file and library names to position the tape. A tape file is identified by the volume name and FSEQ. If an identification discrepancy occurs between the tape volume name and FSEQ, an error message informs you of the conflict. You can then respecify the input file name or leave it blank. If you leave the field blank, TAPECOPY uses only the file sequence number to position the tape. If you do not specify the file sequence number, TAPECOPY uses the file name to position the tape. When you rerun TAPECOPY in the same processing session, the file sequence number is automatically incremented by one. The file sequence range is from 1 to 9999. The default value is 1.
LABEL	Specify the type of tape label. Enter AL for ANSI, IL for IBM, or NL for No label. The default value is AL.

If the input is from an IBM DOS tape, specify whether or not the file contains a Header2 (HRD2) label in the Header2 field. When the input is defined and you press ENTER, TAPECOPY prompts you to mount the disk or tape (if the specified tape or disk volume is not already mounted). Specify a device number and press PF4. You can respecify the volume name.

NOTE
If you have files that reside on more than one tape volume, you must copy files in the correct sequence. Refer to Section 25.6 for details on multivolume tape copying.

25.3 DEFINING AN OUTPUT TAPE FILE

After you enter the input file fields and press ENTER, the TAPECOPY Output Tape File Definition screen (Figure 25-3) appears.

Labeled tape files are identified by file, library, and volume names. When you copy a WP document, you must provide file and library names that can be recognized by VS WP (if the document is to be accessed by VS WP).

WP file names consist of four digits and a WP library letter. The letter is taken from the suffix of the WP library name. For example if WP file 1214 existed in library DOCMNTM, the resulting WP file name would be 1214M. Double suffixes, such as DOCMNTMM denote a lowercase library letter (1214m, following the above example). Refer to the *VS System User's Guide to VS/IIS* for details about WP libraries and file names.

Unlike input tape files, you must specify the file sequence number for both labeled and unlabeled output tape files. An unlabeled tape file is identified by volume name and file sequence number.

```
***  MESSAGE 0001 BY TPCOPY

                INFORMATION REQUIRED BY  PROGRAM  TAPECOPY
                           TO DEFINE OUTPUT


Wang VS Tapecopy Program Version  x.xx.xx

Please identify the output file:

    FILE     = ********   LIBRARY  = ********   VOLUME   = ******

    DEVICE   = TAPE****

If output is to a tape file, please specify:

    FSEQ     = ***1  File sequence number

    LABEL    = AL   AL=ANSI Label, IL=IBM Label, NL=No Label
```

Figure 25-3. TAPECOPY Output Tape File Definition Screen

The TAPECOPY Output Tape File Definition screen fields are described as follows:

Field	Description
FILE, LIBRARY, and VOLUME	Specify the output file by entering the file, library, and volume names in the input fields. Tape files can also be selected by specifying the tape volume name and the file sequence (FSEQ) number.
DEVICE	Specify the device on which the output file is placed. The value for the Device field can be TAPE or DISK. The default value for the output device is DISK when the input device is TAPE. The default value for the output device is TAPE when the input device is DISK.

If the output file is on tape, specify the file sequence number and the label type as follows:

Field	Description
FSEQ	Specify the sequence number of the file on the tape. You must provide the file sequence number for both labeled and nonlabeled tapes.
LABEL	Specify AL (Wang standard (ANSI) label type), IL (IBM label type), or NL (no label is placed on the tape). TAPECOPY enters the library and file names into the output tape label using the format, LIBRARY.FILE. For a tape with an IBM label, TAPECOPY performs an automatic translation of the label into EBCDIC.

25.4 DEFINING TAPE FILE RECORD FORMAT CHARACTERISTICS

After the output tape file has been specified, TAPECOPY displays the TAPECOPY Record Format screen. TAPECOPY obtains the default output file record format characteristics for disks and labeled tapes from the input file fields discussed in Section 25.2. The TAPECOPY Record Format screen fields are described as follows:

Field	Description
RECFORM	Specify the tape record format. F Fixed length V Variable length (Wang format) I Variable length (IBM format) U Undefined length
LARGEST RECSIZE	Specify the length of the record in bytes. For variable-length records, enter the length of the largest record. The maximum record size for all record formats is 2048. The following table summarizes the minimum record sizes for all record formats:

Minimum Record Format	Minimum Record Size Input	Record Size Output
F	12	18
V	4	4
I	4	4
U	12	18

Field	Description
BLOCKED	Specify whether the file is blocked or unblocked. Files with fixed-length records can be blocked or unblocked. Files with variable-length records in Wang format are always blocked. Variable-length IBM format output records must be blocked. (Files with variable-length records in IBM format can be blocked or unblocked for input only.) Files with records of undefined length are unblocked. If the default value (Y) is used, the records are blocked. Value N indicates unblocked records.
LARGEST BLOCKSIZE	Specify the block length in bytes. The maximum block size is 32 KB (32,768 bytes). Minimum block size is 12 bytes for input files and 18 bytes for output. The block size entered for a fixed-length record must be an even multiple of the record size. The block size for variable-length records in Wang format must be at least 4 bytes longer than the maximum record size. The block size for variable-length records in IBM format must be at least 8 bytes longer than the maximum record size.
COMPRESS	Specify Y (Yes) or N (No) to select file compression. File compression saves tape or disk space. A value of Y compresses the file. **CAUTION** *If the tape copy is going to a non-Wang system, you should know whether the non-Wang system can easily decompress data from that tape. Therefore, you should use caution when specifying Y (Yes) for data compression.*

25.5 DEFINING AN OUTPUT DISK FILE

If the output file is to reside on a disk, the TAPECOPY Output Disk File Definition screen (Figure 25-4) appears.

```
***  MESSAGE 0301 BY TPCOPY

                INFORMATION REQUIRED BY  PROGRAM  TAPECOPY
                            TO DEFINE DISKFILE


Please modify output file attributes as necessary:

    FILEORG  = C       Options  = C-Consecutive, I-Indexed, X=Program, P-Print
    RECFORM  = V       Options  = F-Fixed Length, V-Variable Length
    COMPRESS = Y       Options  = Y-Yes, N-No
    FILECLAS = *       RETAIN   = *** Days

For indexed files, please specify:

    KEYLEN   = ***     KEYPOS   = ***** From 1
    IPACK    = 100 %   (Packing density for index blocks)
    DPACK    = 100 %   (Packing density for data blocks)
```

Figure 25-4. TAPECOPY Output Disk File Definition Screen

A description of the output disk file fields are described as follows:

Field	Description
FILEORG	Specify C (Consecutive), I (Indexed), P (Print), or X (program) file organization. The default value is C.
RECFORM	Specify F (Fixed-length) or V (Variable-length) record format. The default value is V.
COMPRESS	Specify Y (Yes) or N (No) for file compression. File compression can save space. The default value is Y. For more information about file compression, refer to the *VS Data Management System (DMS) Reference*.
FILECLAS	Specify the file protection class. This code determines which users are allowed to access the file. Values are A to Z, #, @, $, or blank. For more information about file protection classes, refer to the *VS System User's Introduction*.
RETAIN	Specify the number of days the file is protected from being deleted or renamed.

If the output file is indexed (Fileorg = I), specify the following fields:

Field	Description
KEYLEN	Specify the length of the key field.
KEYPOS	Specify the starting byte position of the key field.
IPACK, DPACK	Specify the packing density for index (IPACK) and data (DPACK) blocks. The range is 1 to 100 percent. The default value is 100 percent. For more information about packing densities, refer to the *VS Data Management System (DMS) Reference*.

The default values are obtained from the input file definitions. If the specified output volume is not mounted, a message is displayed that directs you to mount the volume or respecify the volume name.

After you have entered the output fields, the output file is created. When the copy operation is completed, a TAPECOPY End-of-Job screen displays how many records are in the output file, and gives you the option to run TAPECOPY again by pressing PF1, or to terminate processing by pressing PF16.

25.6 MULTIVOLUME TAPE COPYING

TAPECOPY can copy a labeled tape file that resides on two or more tape volumes. If a file that you want to copy resides on more than one tape volume, the whole file must be copied. TAPECOPY cannot copy a partial file.

TAPECOPY displays a message when it reaches the end of a tape volume. You are then prompted to mount the next input volume by specifying the volume name and device number. TAPECOPY checks the newly mounted volume for the correct volume sequence number, and then continues reading the input file.

Multivolume output is handled similarly to multivolume input. TAPECOPY displays a message when it reaches the end of an output tape. You are prompted to mount the next volume by specifying the volume name and device number. The write operation continues on the next volume.

25.7 A SAMPLE TAPECOPY PROCEDURE

You can control TAPECOPY processing through the VS Procedure language. Appendix A contains TAPECOPY GETPARMs. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

The following procedure copies a disk file to the twelfth file on an AL-tape. Character set conversion is not selected. Because the input disk file characteristics (record and block lengths) are appropriate, the default values obtained for the TAPEFILE parameter reference name (prname) are used by the procedure. The procedure exits TAPECOPY when the file is copied.

```
PROCEDURE
    RUN TAPECOPY
        ENTER INPUT FILE=1214, LIBRARY=DOCMNTM, VOLUME=WP,
            DEVICE=DISK
        ENTER OUTPUT FILE=12ACCTSR, LIBRARY=MONLIB,
            VOLUME=COTAPE, DEVICE=TAPE, FSEQ=12, LABEL=AL
        ENTER TAPEFILE
        ENTER EOJ 16
RETURN
```

CHAPTER 26
THE TAPEINIT UTILITY

26.1 INTRODUCTION

Before you can store data on a tape volume, you must initialize the tape volume. The functions provided by TAPEINIT are as follows:

- Initialize new (and reinitialize old) tape volumes for 4-, 7-, and 9-track tapes. All information previously stored on the volume is destroyed.
- Initialize an IBM-labeled tape volume (9-track only).
- Mount a tape that is not initialized through TAPEINIT rather than using the standard mount procedure.

Figure 26-1 shows an overview of TAPEINIT processing

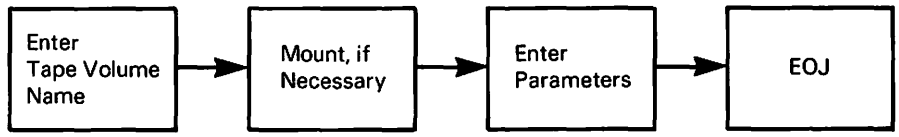


Figure 26-1. TAPEINIT Processing.

26.2 SPECIFYING THE VOLUME

The TAPEINIT Volume Definition screen (Figure 26-2) prompts you to specify a name for the tape volume that you want to initialize. When this is done, press ENTER. If you have not mounted the tape volume, press PF4 to initiate the mount procedure. Type in a device number and press ENTER. TAPEINIT processing continues after the completion of the mount procedure. If you have mounted the tape, the Mount option is not displayed.

Section 26.3 describes the TAPEINIT process for mounted, 7-track tapes. Section 26.4 describes the TAPEINIT process for mounted, 4- and 9-track tapes.

```
*** MESSAGE 0001 BY TPINIT

              INFORMATION REQUIRED BY  PROGRAM  TAPEINIT
                          TO DEFINE INPUT


         ***  Wang VS Tape Initialize Program - Version  x.xx.xx  ***

         The Tapeinit program is used to initialize a tape by writing

               (A) a standard ANSI label     for AL tape,
               (B) an IBM label in EBCDIC    for IL tape, or
               (C) an End-of-Tape indication for NL tape

                All previous data on the tape is destroyed.

                      Please enter the tape volume:

                           VOLUME   = ******

                    Press PF16 to terminate the TAPEINIT program.
```

Figure 26-2. TAPEINIT Volume Definition Screen

26.3 7-TRACK TAPE INITIALIZATION

After you mount a 7-track tape volume, specify the following initialization fields:

Field	Description
PARITY	Specify the error-checking system used for the tape's data. If you accept the default response of EVEN, the parity bit is set to 0 or 1, as required. This yields an even number value when all bits in the character are totalled. If you specify ODD in the Parity field, the total of all bits in the character is an odd number.
DENSITY	Specify the number of bits per inch (bpi) to be stored on each track of the tape that you want to initialize. By default, data is stored at 800 bpi.

NOTE
Seven-track tapes are always nonlabeled tapes.

After you have entered values into the fields, TAPEINIT initializes the tape. When the tape is initialized, you can start TAPEINIT processing again by pressing PF1, or terminate processing by pressing PF16.

26.4 4-TRACK OR 9-TRACK TAPE INITIALIZATION

If you mount a 4-track or a 9-track tape volume (and depending on the type of tape you mount), specify one or more of the following label attribute fields:

Field	Description
LABEL	Specify AL (Wang standard (ANSI) label type), IL (IBM label type), or NL (no label is placed on the tape). The default value for 9-track tapes is AL. The default value for 4-track tapes is NL which cannot be modified.
OWNER	Specify the user responsible for the tape's contents. You can leave this option blank, but you must fill in this option if you wish to restrict access rights. The owner name becomes part of the label. The owner name is up to 14 characters long for AL tapes, and up to 10 characters long for IL tapes.
DENSITY	Specify the recording density for this particular tape. Density is restricted by the tape drive model. Dual density drives support densities of 800 or 1600 bits per inch (bpi). Tridensity drives support densities of 800, 1600 or 6250 bpi. Four-track cartidge tape drives support a density of 6400 bpi.

NOTE
Only IBM operating system tape labels are supported for the IBM label type.

After you specify the label attributes, TAPEINIT initializes the tape. When the tape is initialized, you can restart TAPEINIT processing again by pressing PF1, or terminate processing by pressing PF16.

26.5 A SAMPLE TAPEINIT PROCEDURE

You can control TAPEINIT processing through the VS Procedure language. You can specify all TAPEINIT options through a procedure. Procedure language MOUNT and DISMOUNT statements, however, cannot be embedded in a RUN, ENTER (or DISPLAY) sequence; they must precede the RUN statement or follow the last ENTER statement. A TAPEINIT Mount operation cannot be embedded in the procedure because TAPEINIT uses a respecification GETPARM. Appendix A contains a list of TAPEINIT GETPARMs. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

The following procedure logically mounts and initializes a 9-track standard label tape (named ZZTAPE) with a density of 1600 bpi and an owner specified as YOU. The procedure logically dismounts the tape when TAPEINIT processing ends. Because all TAPEINIT processing is automated, you (or the system operator if the procedure is run in background mode) need only physically mount and dismount the tape on the tape drive.

```
PROCEDURE
    MOUNT TAPE ZZTAPE ON 040 WITH NO LABEL FOR EXCLUSIVE USAGE
        RUN TAPEINIT
            ENTER INPUT VOLUME=ZZTAPE
            ENTER TAPE LABEL=AL, OWNER=YOU, DENSITY=1600
            ENTER EOJ 16
    DISMOUNT TAPE ZZTAPE
RETURN
```

CHAPTER 27
THE TRANSL UTILITY

27.1 INTRODUCTION

The TRANSL utility enables you to translate a file from one character set to another, using a standard ASCII-to-EBCDIC (American Standard Code for Information Interchange to Extended Binary Coded Decimal Interchange Code), EBCDIC-to-ASCII, or user-defined translation table. In order to translate a file, you must specify an input file, a translation function, field manipulation options (indicating the fields from the input file contained in the output file), and the output file specifications.

The TRANSL utility creates a translated output file, which can be used to transfer files among systems with different character sets. You can also create a copy of a file in another character set. You can perform the following functions with the TRANSL utility:

- Translate files from ASCII-to-EBCDIC or EBCDIC-to-ASCII. When you select either of these functions, translation is performed using a standard conversion table.
- Translate files to or from a nonstandard character set. A user-defined translation table is used for the translation. You can create, retrieve, or modify a translation table, as well as save the table for subsequent file translation with this function.
- Perform field manipulation functions, bypassing all translation activity. You can rearrange, insert, or delete input record fields to create a modified output record.
- Combine both field manipulation functions and translation functions. You can rearrange, insert, or delete input record fields as a translated output file is being created.
- Perform selective translation of user-specified multiple record types. You specify indicator characters that identify each record type.

TRANSL processing involves the following steps:

1. Specify the input file and a TRANSL function. You can also create, select, or modify a user-defined translation table that performs translation functions.
2. Specify multiple record types (optional) to selectively translate the records of a file.
3. Specify field manipulation options to enable the utility to perform translation functions.
4. Specify the output file specifications.

You can also control processing through the VS Procedure language (refer to Section 27.7). Figure 27-1 shows an overview of TRANSL processing.

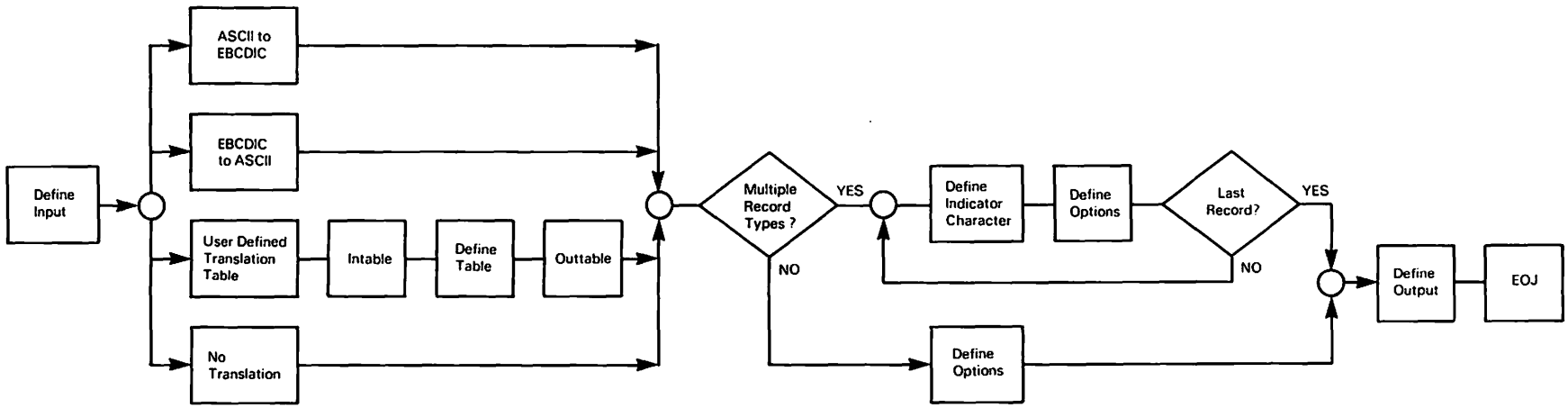


Figure 27-1. TRANSL Processing

27.2 DEFINING THE INPUT

To begin TRANSL processing, press PF1 (RUN Program or Procedure) from the Command Processor menu. Specify TRANSL in the Program field and press ENTER. When TRANSL processing begins, the TRANSL Input Definition screen (Figure 27-2) appears. Specify the file, library, and volume name of the file to be translated or reorganized. You need not specify the whole word for each option (the initial letter is sufficient). The Code option defaults to EBCDIC.

```
***  MESSAGE 0001 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                          TO DEFINE INPUT




                        Please supply input parameters

      FILE     = ********    LIBRARY  = DOCMNTR*    VOLUME   = WP****
         TYPES    = NO*       (YES/NO for multiple record types)

     Please specify translation to be performed on the output records

        CODE     = EBCDIC    Options: ASCII   -  ASCII-to-EBCDIC
                                      EBCDIC  -  EBCDIC-to-ASCII
                                      TABLE   -  User-supplied table
                                      NONE    -  No translation

                    Press ENTER to continue or PF16 to end

    <<<<<<  WANG VS File Translation Utility - VERSION x.xx.xx  >>>>>>
```

Figure 27-2. Sample TRANSL Input Definition Screen

The TRANSL Input Definition screen fields are described as follows:

Field	Description
FILE, LIBRARY, VOLUME	Specify the file that you want translated, and the library and volume on which the file resides.
TYPES	Specify YES if the file that you want translated has multiple record types.
CODE	Specify the type of translation to be performed:
	ASCII — This option translates the ASCII character set to the EBCDIC character set.
	EBCDIC — This option translates the EBCDIC character set to the ASCII character set.
	TABLE — The Table option enables you to create a translation table, or to access and modify a previously defined table. This option converts files to or from nonstandard character sets.

Field	Description	
CODE (cont.)	NONE	This option (NONE) enables you to use the field manipulation features of the TRANSL utility without any character translation. When you specify NONE (no translation), all translation table activities are skipped, and the TRANSL Field Options screen is displayed.

27.2.1 Performing a Standard Translation

Both ASCII-to-EBCDIC and EBCDIC-to-ASCII conversions are processed by use of a predefined conversion table. If you specify ASCII (ASCII-to-EBCDIC) or EBCDIC (EBCDIC-to-ASCII) for the Code field, a copy is made of the input file and that copy (the output file) has each ASCII character translated to the corresponding EBCDIC character by the TRANSL utility.

If you indicate a standard character set translation (assuming that you do not select Multiple Record Types, discussed in Section 27.3), the TRANSL Field Options screen (discussed in Section 27.4) appears. You must identify the output file's field attributes on the TRANSL Field Options screen before the utility can create an output file and perform the selected translation.

27.3 DEFINING MULTIPLE RECORD TYPES

When you specify YES in the Type field and ASCII, EBCDIC, or None in the Code field on the TRANSL Input Definition screen, the TRANSL Record Types screen (Figure 27-3) appears. If you specify TABLE in the Code field, the TRANSL Record Types screen appears after several other screens. Defining a translation table is discussed in Section 27.5.

When the TRANSL Record Types screen appears, you are prompted to identify various types of records by either the presence or absence of the specified character(s), known as the indicator character(s). TRANSL reorganizes each record type in a different manner.

```
***  MESSAGE 1007 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                           TO DEFINE TYPES



Please define the character(s) and location(s) to be used in identifying the
record types in the file. Each character may be specified as one ASCII
character or two HEX digits.
Leave this screen blank to indicate the end of the different record types.

*Record type 01 will be identified by:
              CHAR1    = ** in COLUMN1  = ****  ,  SWITCH1  = YES
          and CHAR2    = ** in COLUMN2  = ****  ,  SWITCH2  = YES
          and CHAR3    = ** in COLUMN3  = ****  ,  SWITCH3  = YES

          The above indicators are as found in which character set ?
                    CHARSET  = INPUT*    (INPUT or OUTPUT)

Note:  SWITCH(1,2,3) = "YES", Means that this character's presence will
       be used to identify this record type.  SWITCH(1,2,3) = "NO", Means
       That its absence will be used to identify this record type.
```

Figure 27-3. Sample TRANSL Record Types Screen

The TRANSL Record Types screen fields are described as follows ("x" indicates a numeric value):

Field	Description
CHARx	Specify an ASCII character (or other symbol) or a 2-digit hex value as the indicator character to distinguish each record type.
COLUMNx	Specify the column number of the input record in which the corresponding CHAR field appears.
SWITCHx	Specify whether a record type can be identified by the presence or absence of the character indicated in the corresponding CHAR field. A YES value indicates that the presence of the character identifies the record type (A NO value indicates that the absence of the character identifies the record type).
CHARSET	Specify which character set (INPUT or OUTPUT) contains the specified indicator character (CHAR).

27.4 SPECIFYING FIELD OPTIONS

To define the field attributes, the TRANSL Record Types (Figure 27-3) and TRANSL Field Options (Figure 27-4) screens are alternately displayed for each record type. The TRANSL Field Options screen enables you to specify field manipulation options and to define field attributes for each multiple record type. To terminate record type definitions and display the TRANSL Output Definition screen, leave the fields blank on a TRANSL Record Types screen (Figure 27-3) and press ENTER.

Field options determine the order, length, and data type of each field to be contained in the output file. You can also specify deletion, insertion, or rearrangement of input file fields to be contained in the output file.

You must specify field manipulation options on the TRANSL Field Options screen (Figure 27-4) before the utility performs the selected translation. This screen immediately follows the TRANSL Input Definition screen if you specify NO in the types field.

```
***  MESSAGE 0001 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                          TO DEFINE OPTIONS


Press ENTER after defining the necessary fields

                                                                  (Code I,D
Field attributes:                                               or leave blank)
       Position(Input)     Length(Bytes)     Type(B,C,P OR Z)   Insert   Delete
FLD1 :  POST01   = ****   LENGTH01 = ****     TYPE01   = *       SWITCH01 = *
FLD2 :  POST02   = ****   LENGTH02 = ****     TYPE02   = *       SWITCH02 = *
FLD3 :  POST03   = ****   LENGTH03 = ****     TYPE03   = *       SWITCH03 = *
FLD4 :  POST04   = ****   LENGTH04 = ****     TYPE04   = *       SWITCH04 = *
FLD5 :  POST05   = ****   LENGTH05 = ****     TYPE05   = *       SWITCH05 = *
FLD6 :  POST06   = ****   LENGTH06 = ****     TYPE06   = *       SWITCH06 = *
FLD7 :  POST07   = ****   LENGTH07 = ****     TYPE07   = *       SWITCH07 = *
FLD8 :  POST08   = ****   LENGTH08 = ****     TYPE08   = *       SWITCH08 = *

*Options for field type: B=Binary,C=Character ,P=Packed decimal,Z=Zoned decimal
Notes: If all 8 fields are used the option to define more fields will appear
  Selecting delete(D) causes this input field not to be transferred to output
  Selecting insert(I) creates a new field in output (Leave position zero )
```

Figure 27-4. TRANSL Field Options Screen

If the input file is a fixed-length record file, you must account for all bytes of the input record on the TRANSL Field Options screen. The Delete option is available to indicate that a specified field is not to be included in the output record. If the input file is a variable-length record file, and you do not want to include a specified field in the output record, you can omit the field on the TRANSL Field Options screen.

The field attributes on the TRANSL Field Options screen are arranged in the order in which they appear in the output record. FLD1 is the first field in the output record, FLD2 is the second output field, etc.

The TRANSL Field Options screen displays eight fields at a time. When you specify the eight fields on one screen, the program then displays another eight fields. You can specify a maximum of 25 fields. The TRANSL Field Options fields are described as follows ("xx" refers to a numerical value):

Field	Description
POSTxx	Specify the position of the field as it is contained in the input record. For example, if a field that begins at Byte 11 (in the input record) is to be the first field in the output record, FLD1 has a position value of 11. If the output record is to have the same order as the input record, the fields' positions for the output record must be listed in the order in which they appear in the input record. Fields of the same data type, except zoned fields, can be grouped. (The first position of a record is considered to be 1, not 0.) You can rearrange the order of records from the input file to the output file. For example, if the first two fields in the input file (beginning at Bytes 1 and 20, respectively) are to be switched for the output file, enter 20 in POST01 and 1 in POST02. This places the input file data (beginning at Byte 20) into Field 1 in the output file, and Field 2 of the output file receives input file data from Byte 1.
LENGTHxx	Specify the length of the field (in bytes) for the position that corresponds to this length.
TYPExx	Specify the data type of the input field. Valid Data type values are B (Binary), C (Character), P (Packed), or Z (Zoned decimal). Binary fields are not translated. When Zoned decimal characters are translated, the sign is included in the low-order byte of the field.
SWITCHxx	Specify I or D to insert or delete fields that are to be contained in the output file. The Switch field defaults to blanks. The default has no effect on either the input or output field. I (Insert) indicates that a new field of a specified length is to be inserted in the output file. You must specify POSTxx (the position of the field to be inserted) as a blank or zero (meaning that it has no position in the input file). To insert blanks in Bytes 21 to 30 in the output file, you indicate the field(s) that occupy Bytes 1 to 20. The next POSTxx is given a value of 0, LENGTHxx is given a value of 10, and SWITCHxx is given the value of I. (If TYPExx = C, a field of blanks is inserted; if TYPExx = B, a field with a value of Hex '00' is inserted. If TYPExx = P or Z, a field with a value of decimal zero is inserted.) D (Delete) indicates that the input field is not to be copied to the output file. This option is only significant for fixed-length files and does not apply to variable-length records. If you do not account for all bytes of a fixed-length input record, then the input and output record lengths differ and an error message screen appears. From this error message screen, press PF1 to redefine the field attributes. You can also press PF16 to terminate TRANSL processing.

When an indexed file is translated, an indexed screen (similar to the TRANSL Field Options screen) is displayed. You can change the file's primary key position and length in the KEYPOSxx and KEYLENxx fields. If you want these values to be identical to those of the input file, leave them blank. No duplicate keys are allowed because TRANSL cannot create an alternate indexed output file. To process the TRANSL Field Options screen, press ENTER. If you specified multiple record types on theTRANSL Input Definition screen (Figure 27-2), you must also specify the record types on the TRANSL Multiple Record Types screen before the field options are processed.

27.4.1 Manipulating Fields Without Translation

You can bypass all translation activities and only perform field manipulation functions by specifying None in the Code field and No in the Type field on the TRANSL Input Definition screen (Figure 27-2). If this is done, the TRANSL Field Options screen (Figure 27-4) appears, which prompts you to make field manipulation selections by indicating the output file's field attributes.

27.5 DEFINING A TRANSLATION TABLE

To edit an existing user-defined translation table, specify Table in the Code field on the TRANSL Input Definition screen; the TRANSL Intable screen appears (Figure 27-5). To create a new translation table, do not specify the fields of the Intable screen. Press ENTER to display a default translation table screen. If you are modifying an existing translation table, specify the file, library, and volume in which the translation table can be found and press ENTER.

```
***  MESSAGE T001 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                          TO DEFINE INTABLE



     Please specify the location of an existing translate table file
          (Leave FILE blank to specify a new translate table)

       FILE     = ********   LIBRARY  = DOCMNTR*   VOLUME   = WP****

         (The translate table (256-byte) record will be used as)
         ( the default value for the table modification screen )


              Press ENTER after specifying the input file
                              - or -
                Press PF1 to return to the INPUT screen
```

Figure 27-5. Sample TRANSL Intable Screen

A translation table is a 256-byte string used to convert one character set to another. When you create a new table or modify an existing table, the TRANSL Utility Translation Table screen (Figure 27-6) displays a table that contains a complete 256 two-digit set of hexadecimal values. If you create a new translation table, a default table is displayed. The default table contains each hexadecimal value assigned to itself. (The first entry in the TRANSL Translation Table screen shows 00010203 for the first four hex values: 00, 01, 02, and 03.)

You need to indicate only altered values. For example, to change all ASCII uppercase and lowercase characters to uppercase, change the hexadecimal values of HEX 61 through 7A to 41 through 5A, as shown in the sample Modified TRANSL Utility Translation Table screen (Figure 27-7). Each character value in the lowercase ASCII alphabet is translated into the corresponding uppercase character.

```
***  MESSAGE T002 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                          TO DEFINE TABLE


Please modify as desired and/or select from choices listed

    HEX00#0F = 00010203 04050607 08090A0B 0C0D0E0F
    HEX10#1F = 10111213 14151617 18191A1B 1C1D1E1F     -----------------
    HEX20#2F = 20212223 24252627 28292A2B 2C2D2E2F    *                 *
    HEX30#3F = 30313233 34353637 38393A3B 3C3D3E3F    * Please select:  *
    HEX40#4F = 40414243 44454647 48494A4B 4C4D4E4F    *                 *
    HEX50#5F = 50515253 54555657 58595A5B 5C5D5E5F    * ENTER - Continue*
    HEX60#6F = 60616263 64656667 68696A6B 6C6D6E6F    *                 *
    HEX70#7F = 70717273 74757677 78797A7B 7C7D7E7F    * PF1 - Return to *
    HEX80#8F = 80818283 84858687 88898A8B 8C8D8E8F    *  INTABLE screen *
    HEX90#9F = 90919293 94959697 98999A9B 9C9D9E9F    *                 *
    HEXA0#AF = A0A1A2A3 A4A5A6A7 A8A9AAAB ACADAEAF    *                 *
    HEXB0#BF = B0B1B2B3 B4B5B6B7 B8B9BABB BCBDBEBF    * PF14 - Further  *
    HEXC0#CF = C0C1C2C3 C4C5C6C7 C8C9CACB CCCDCECF    *     information *
    HEXD0#DF = D0D1D2D3 D4D5D6D7 D8D9DADB DCDDDEDF    *                 *
    HEXE0#EF = E0E1E2E3 E4E5E6E7 E8E9EAEB ECEDEEEF     -----------------
    HEXE0#FF = F0F1F2F3 F4F5F6F7 F8F9FAFB FCFDFEFF
```

Figure 27-6. Sample TRANSL Utility Translation Table Screen

```
***  MESSAGE T002 BY TRANSL

                INFORMATION REQUIRED BY  PROGRAM  TRANSL
                          TO DEFINE TABLE


Please modify as desired and/or select from choices listed

    HEX00#0F = 00010203 04050607 08090A0B 0C0D0E0F
    HEX10#1F = 10111213 14151617 18191A1B 1C1D1E1F     ----------------
    HEX20#2F = 20212223 24252627 28292A2B 2C2D2E2F    *                *
    HEX30#3F = 30313233 34353637 38393A3B 3C3D3E3F    * Please select: *
    HEX40#4F = 40414243 44454647 48494A4B 4C4D4E4F    *                *
    HEX50#5F = 50515253 54555657 58595A5B 5C5D5E5F    * ENTER - Continue*
    HEX60#6F = 60414243 44454647 48494A4B 4C4D4E4F    *                *
    HEX70#7F = 50515253 54555657 58595A7B 7C7D7E7F    * PF1 - Return to *
    HEX80#8F = 80818283 84858687 88898A8B 8C8D8E8F    *  INTABLE screen *
    HEX90#9F = 90919293 94959697 98999A9B 9C9D9E9F    *                *
    HEXA0#AF = A0A1A2A3 A4A5A6A7 A8A9AAAB ACADAEAF    *                *
    HEXB0#BF = B0B1B2B3 B4B5B6B7 B8B9BABB BCBDBEBF    * PF14 - Further  *
    HEXC0#CF = C0C1C2C3 C4C5C6C7 C8C9CACB CCCDCECF    *     information *
    HEXD0#DF = D0D1D2D3 D4D5D6D7 D8D9DADB DCDDDEDF    *                *
    HEXE0#EF = E0E1E2E3 E4E5E6E7 E8E9EAEB ECEDEEEF     ----------------
    HEXE0#FF = F0F1F2F3 F4F5F6F7 F8F9FAFB FCFDFEFF
```

Figure 27-7. Sample Modified TRANSL Utility Translation Table

When you are satisfied with the TRANSL Translation Table screen, press ENTER and the TRANSL Outtable screen appears. The TRANSL Outtable screen is very similar to the TRANSL Intable screen (Figure 27-5). The Outtable screen prompts you to specify a file, a library, and a volume in which the translation table is saved for future use by TRANSL. After you specify the appropriate information, press ENTER to process the screen.

If the name that you specify for the translation table already exists in the file, library, and volume specified on the Outtable screen, you can press PF3 to delete the existing table and then replace it with the newer one or you can respecify the name of the output file. You can also continue the current translation with the modified translation table but without saving the table in a file by pressing PF16.

After you process the TRANSL Outtable screen, the TRANSL Field Options screen (Figure 27-4) appears. Identify the output file's field attributes. This must be done before TRANSL can complete the output file and perform the selected translation with your translation table. (Refer to Section 27.4 for information on specifying field options.)

27.6 DEFINING THE OUTPUT FILE

When the TRANSL options are processed, the TRANSL Output Definition screen (Figure 27-8) appears. The TRANSL Output Definition screen prompts you to specify the output file fields.

```
***  MESSAGE 000  BY OPEN

                  INFORMATION REQUIRED BY  PROGRAM  TRANSL
                              TO DEFINE OUTPUT




PLEASE ASSIGN "OUTPUT"          (TO BE CREATED AS OUTPUT BY THE PROGRAM)

TO ASSIGN THIS FILE TO A DISK FILE, PLEASE SPECIFY:
    FILE     = ******** IN LIBRARY  = DOCMNTR* ON VOLUME   = WP****
    RECORDS  = 0000016     RETAIN   = *** DAYS    RELEASE  = YES
    FILECLAS = *

    DEVICE   = DISK*******
```

Figure 27-8. Sample TRANSL Output Definition Screen

Specify the file, library, and volume names of the output file, as well as the following fields:

Field	Description
RECORDS	By default, the Records field indicates the number of records obtained from the input file. If you plan to update the output file at a later date, you can increase the size of the output file. Increase the number of records to the number that you anticipate the file will need to support future updates. Also, specify NO in the Release field so that the extra space that you allocate is not released for other use.
RETAIN	Specify the number of days that the file is protected from deletion. A file can be protected for a maximum of 999 days and a minimum of zero days.
FILECLAS	Specify the protection class of the output file. The protection class determines which users can access the file. The following file protection classes are available: #, $, @, blank, and A to Z. (For more information on file class options, refer to the *VS System User's Introduction*.)

Field	Description
RELEASE	Specify (YES or NO) whether unused storage extents, previously allocated for data, can be released for use by other files. (Specify NO if you have allocated more space than the file requires and want the file to keep that space for future updates to it.) The Release field defaults to the input file's specification.
DEVICE	Specify the device on which the output file is to be located. Specify DISK in the Device field to copy the file to disk or diskette. If the output file is a print file, you indicate PRINTER as the device. (The TRANSL utility does not copy to tape.) The default value is DISK.

When TRANSL activity is complete, an output file is created, containing the translations and manipulations that you specified. The input file still exists in its original form. The TRANSL EOJ screen is displayed, and you can either exit from the TRANSL utility by pressing PF16 or rerun the utility by pressing PF1.

27.7 A SAMPLE TRANSL PROCEDURE

You can control TRANSL processing through the VS Procedure language. You can specify TRANSL options and output file organization. A complete list of TRANSL GETPARMs is provided in Appendix A. Refer to the *VS Procedure Language Reference* for details about VS Procedure language syntax.

The following sample procedure converts a 120-byte indexed file, containing only character fields, from the ASCII to the EBCDIC character set. The procedure specifies the input file and does not select output file reorganization. The procedure displays a set of default file fields for the output file and finally exits the utility.

```
PROCEDURE
    RUN TRANSL
        ENTER INPUT FILE=DATASCII, LIBRARY=TRANSME,
            VOLUME=SYSTEM, CODE=EBCDIC
        ENTER OPTIONS POST01=0001, LENGTH01=0120, TYPE01=C
        ENTER INDEXED
        DISPLAY OUTPUT FILE=DATEBCDI, LIBRARY=TRANSED,
            VOLUME=SYSTEM
        ENTER EOJ 16
RETURN
```

CHAPTER 28
THE VERIFY UTILITY

28.1 INTRODUCTION

The VERIFY utility performs a comprehensive check on the validity of the various components of an indexed file and identifies any damage. You should be familiar with the internal structure of VS indexed files before attempting to use the VERIFY utility. For more information on the internal structure of VS indexed files, refer to the *VS Data Management System (DMS) Reference*. You can perform the following functions with the VERIFY utility:

- Verify a single indexed file, all the indexed files in a library, or all the indexed files in all the libraries on a volume
- Verify the primary index structure only, or both the primary and all alternate index structures
- Verify the consistency between VTOC File Descriptor Record (FDR1) entry fields and the file data and index structures
- Verify the consistency between entries in the Alternate Index Descriptor (AXD1) block and file index structures
- Verify the consistency between the primary and alternate index structures of indexed files
- Verify the completeness of the alternate index structure against the data structure
- Verify the physical integrity of the file
- Display error messages as the system detects errors
- Display the summary statistics of the file, at the end of processing each file, and recommend recovery procedures for the file if the system detects damage
- Generate a printed copy that contains summary information on the file characteristics and, if the system detects any errors, generate an error report
- Run the VERIFY utility as a background task by using the VS Procedure language

In the process of verifying a file, VERIFY detects and records all permanent input/output (I/O) errors. The utility is designed to continue processing even if the input file contains significant structural errors, in order to extract as much information as possible about the file, library, or volume. However, if you request an error display, you can exit VERIFY processing at any point after the utility displays an error.

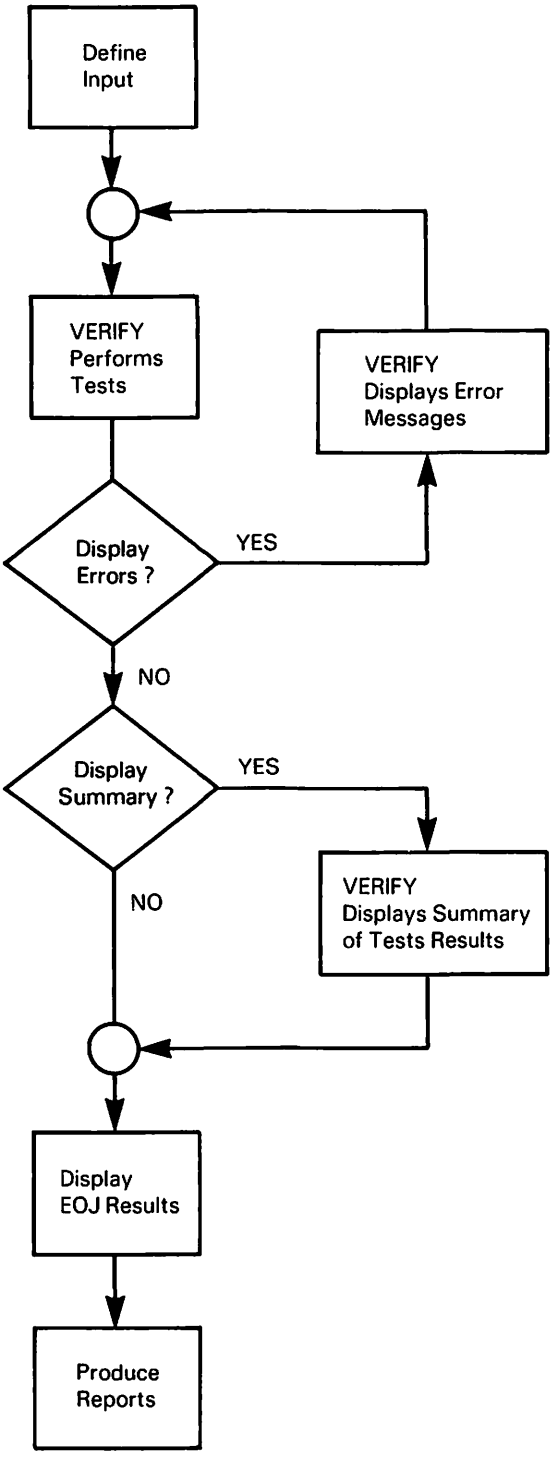


Figure 28-1. VERIFY Processing

Use of the VERIFY utility (after a system crash or in conjunction with the BACKUP utility) can assist system operators by ensuring quick discovery of any indexed file damage. The VERIFY utility operates as follows:

- VERIFY performs tests on the user-specified input, and if requested, displays any error messages and a summary of the test results of the file.
- VERIFY displays the end-of-job results.
- A hard copy of the VERIFY results is optional. If you specify a hard copy, the utility prints a summary report on the file characteristics and if the system detects any errors, an error report is processed.

Figure 28-1 shows an overview of VERIFY processing.

28.2 SPECIFYING THE INPUT OPTIONS

When the VERIFY utility is invoked, the VERIFY Utility Options screen (Figure 28-2) appears.

```
***  MESSAGE 0001 BY VERIFY

                INFORMATION REQUIRED BY  PROGRAM  VERIFY
                           TO DEFINE OPTIONS


         ***  WANG VS VERIFY Program - Version  x.xx.xx  ***
Please specify the following information and press ENTER.
Press PF16 to end the program.

Depending  on the options chosen, program will verify:

RANGE    = FILE                FILE    - File in the library on the volume.
                               LIBRARY - All indexed files in the library.
                               VOLUME  - All indexed files on the volume.

FILE     = ********   in LIBRARY  = DOCMNTR* on  VOLUME   = WP****

VERIFY   = ALL****  INDICES     ALL     - Primary and alternate indices.
                               PRIMARY - Primary index only.

ERROR  DISPLAY  = YES          YES - All errors will be displayed.
                               NO  - Errors will not be displayed.
```

Figure 28-2. VERIFY Utility Options Screen

A description of the VERIFY Utility Options screen fields is as follows:

Field	Description
RANGE	Specify the range (FILE, LIBRARY, or VOLUME) to be verified and whether all encountered errors are to be displayed on the workstation screen. Default values for LIBRARY and VOLUME are obtained from your workstation's default constants which are set through the SET Usage Constants function (PF2) from the Command Processor menu.
FILE, LIBRARY, and VOLUME	Specify the file, library, and volume if you are verifying a file. Specify the library and the volume if you are verifying a library. Specify the volume if you are verifying a volume.
VERIFY	Specify whether you want to verify all indexes or just the primary index.
ERROR DISPLAY	Specify whether you want an error display at the workstation. If you specify YES, error messages, as they are detected, and summary statistics of each file are displayed on the workstation. If you specify NO, you must rely on the summary report to identify errors. The default value is YES.

If you do not have access authorization for a specified file, VERIFY displays an error message, and the file is not verified. You must specify another file if you want to verify another single file. VERIFY skips the file if the LIBRARY or VOLUME input option is chosen. To terminate the program, press PF16.

The names of files do not matter during a library or volume verification because the files are verified in the order in which they appear in the Volume Table of Contents (VTOC); not in alphabetical or numerical order. Also, the name of any library does not matter during volume verification for the same reason.

28.3 TESTING FILES

After you specify the fields on the VERIFY Utility Options screen and press ENTER, VERIFY performs a comprehensive check on the validity of each indexed file. The tests for each file are as follows:

Test	Description
Test 1	Validation of the VTOC File Descriptor Record (FDR1) entry fields. (Refer to Section 28.3.1.)
Test 2	Validation of the entries in the Alternate Index Descriptor (AXD1) block if alternate indexes are present. (Refer to Section 28.3.2.)
Test 3	Validation of the consistency between the primary and alternate index structures of indexed files. (Refer to Section 28.3.3.)
Test 4	Validation of the completeness of the alternate index structure. (Refer to Section 28.3.4.)
Test 5	Validation of the physical integrity of the file. (Refer to Section 28.3.5.)

28.3.1 Validation of the VTOC File Descriptor Record (FDR) Entry Fields

VERIFY examines the FDR1 (Record 1) fields to determine if the FDR1 pointers to the root index block and the first data block exist in the file that you specified for validation. VERIFY also examines the number of index levels that you specified. The maximum number of index levels is 4.

The index structure and the data chain of the file cannot be verified if the pointers are not within the file. If both FDR1 pointers are invalid, the VTOC entry for that file is considered invalid. If only one pointer is invalid and the number of index levels are legal, VERIFY attempts to verify the integrity of the data chain.

The data chain is the linkage of information, which forms a file, library, volume, or other data structure. For example, a word can be thought of as a small data chain. The word "word" is linked by placing the "w" first, "o" second, "r" third, and "d" last. If a file contained only the word "word", a data chain would be set up to make sure that "w" points to "o", and "o" points to "r", and so forth. If the pointers were lost or disorganized, the data (which was "word") can no longer be accessed (or it might come out looking like "odrw," which has no meaning).

"Word" was a data chain of the smallest scale. Data chains in files, libraries, and volumes are enormous. The VERIFY utility ensures that the chain is intact and warns you of bad links.

VERIFY displays a message informing you that the data chain is either broken or intact, depending on the result of the VTOC FDR1 check. If the VTOC entry for a file is invalid or if only one pointer is invalid and the number of index levels are legal, VERIFY cannot check the validity of the index structure. If the input for verification is a library or volume, VERIFY proceeds to the next file. If the input for verification is a file and an error occurs, then processing cannot continue; press PF16 to terminate processing.

VERIFY also checks that the number of data records specified in FDR1 is the same as the number actually counted by VERIFY. If a discrepancy appears, an error code is displayed. (Refer to Section 28.7.1 for error code descriptions.) This code informs you of the discrepancy and provides both the specified count and the actual count. If the input is a file, press ENTER to continue verifying the file. If the input is a library or volume, VERIFY proceeds to the next file. Press PF16 to terminate processing.

28.3.2 Validation of Entries in the Alternate Index Descriptor (AXD) Block

To validate the entries in the Alternate Index Descriptor (AXD1) block (Block 1), VERIFY checks each alternate index to determine if the pointer to the alternate index root block and the pointer to the first alternate index block exist in the file. VERIFY also checks that the number of index levels in each alternate index is within the allowable range.

If one or more of these pointers is invalid, VERIFY informs you that the AXD1 block contains invalid data and that the alternate index cannot be verified. The program then attempts to verify the remaining alternate indexes.

28.3.3 Consistency Validation of Primary and Alternate Index Structures

After the FDR1 and AXD1 blocks are checked, VERIFY performs a series of consistency checks on the primary and alternate index structures. VERIFY tests for the existence of the following conditions:

- The chain pointer in the last block on each index level is equal to hexadecimal FFFFFF.
- The largest key in each lower level index and data block is equal to the key in the corresponding entry in the index structure (excluding the largest key in the last block at each level, which must equal hexadecimal FFFFFF).
- All keys are in ascending order at data level.
- All keys are in ascending order at each index level (excluding the lowest level of alternate index structures where duplicate alternate keys are allowed).
- The corresponding alternate keys are in ascending order if two or more primary keys are equal.

If a pointer inequality is detected in the primary index structure, VERIFY assumes that the index pointer is correct and attempts to verify the file on this basis. If the assumption leads to another discrepancy, VERIFY tests the chain pointer. If this attempt also fails, VERIFY informs you that the primary index structure cannot be verified. If you specify a library or a volume as input, VERIFY proceeds to the next file.

Pointer inequalities in the alternate index structures are processed in the same way as primary index structures, with one exception; if a second discrepancy occurs, you are informed that the current alternate index cannot be verified.VERIFY then proceeds to the next alternate index structure.

When the system first discovers a pointer error or a key discrepancy, VERIFY processing pauses and displays an error message. To continue processing until the next error or discrepancy occurs (if any), press ENTER. To obtain a more detailed description of the error, press PF1. To terminate processing, press PF16.

28.3.4 Validation of Completeness of the Alternate Index Structures

If the alternate indexes are structurally consistent, VERIFY tests them to ensure that they are complete. Tests on the alternate indexes are run concurrently with the consistency validation of the primary index (refer to the previous section). If any errors are found in the primary index, testing the completeness of the alternate index structures is suspended. Errors found by this check are recorded in the error report, but are not displayed individually. Errors are counted and recorded as errors in the alternate indexes.

VERIFY tests for the existence of the following conditions:

- An alternate path contains
  - an alternate key pair or a primary key pair.
  - where the primary key is the same as in the record, only one alternate key pair or one primary key pair exists in the alternate path.

  The following sample illustrates valid key pairs (in Alternate Path 1) and invalid key pairs (in Alternate Path 2):

Alternate Path 1		Alternate Path 2	
Alternate Key	**Primary Key**	**Alternate Key**	**Primary Key**
001	123	001	123
002	123	001	123

- An item in the alternate index contains a primary key. VERIFY checks that the record associated with the primary key exists and that the record access mask indicates that you can access the record through the alternate index.
- The alternate key in the index is equal to the alternate key in the record.

28.3.5 Validation of the Physical Integrity of the File

VERIFY reads all index and data blocks and detects all permanent I/O errors. If the system detects an I/O error when reading a primary index block, VERIFY attempts to check the integrity of the data chain and informs you that the index structure of the file cannot be verified. Also, depending on the outcome of the data chain check, VERIFY informs you that the data chain is either intact or broken.

If an I/O error is found in an alternate index block, VERIFY informs you that this alternate index cannot be verified, and proceeds to the remaining alternate index paths (or structures).

If an I/O error is found in a data block, VERIFY displays an error message, the number of the bad block, and if appropriate, the recommended corrective action. To continue processing of the next block in the same file (if any), press ENTER. To display the range of keys affected, press PF1. To terminate processing (or proceed to the next file in a library or volume), press PF16. If VERIFY encounters problems when expanding a compressed data record, the processing is the same as for an I/O error.

28.4 THE ERROR MESSAGE DISPLAY

If you specify Yes in the Error Display field on the VERIFY Utility Options screen (Figure 28-2), VERIFY displays an error message for each error encountered during processing. A sample error message display is shown in Figure 28-3. As each error message is displayed, VERIFY pauses for your response. You can continue processing by pressing ENTER. For some types of errors, you can display more detailed information on the error by pressing PF1. To terminate processing without completing file verification, press PF16.

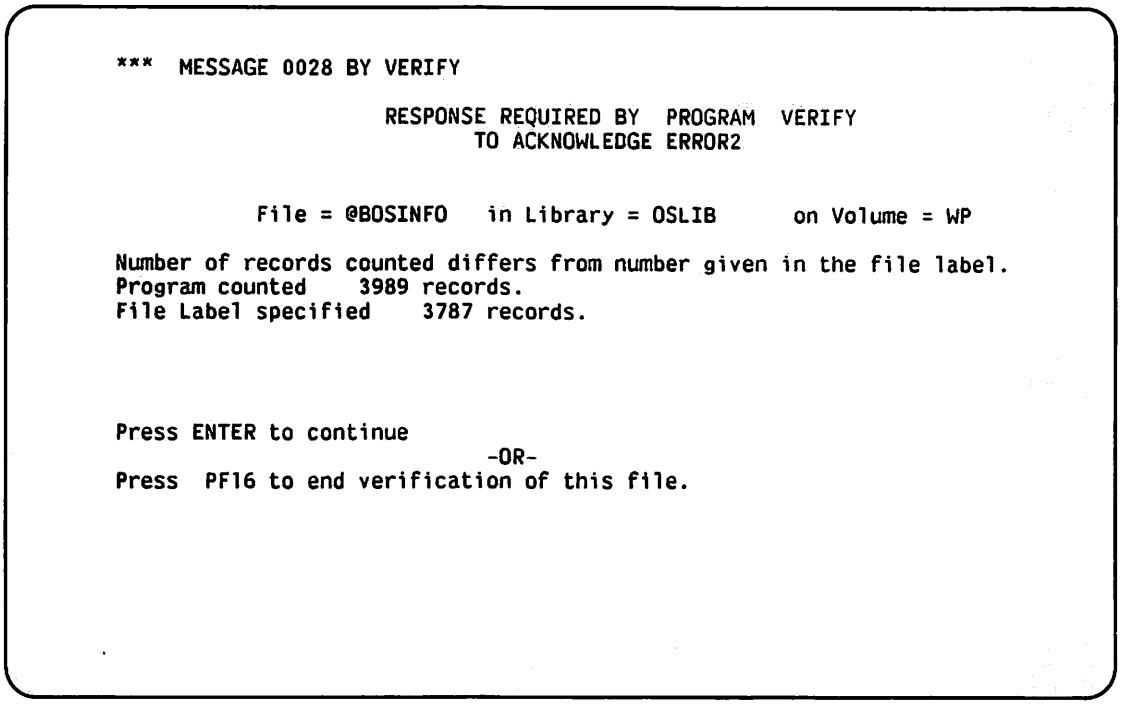


Figure 28-3. Sample Error Message Display

The three displays that are used for error messages are as follows:

Error	Description
ERROR1	ERROR1 reports errors that do not permit further processing of the file (e.g., errors in the VTOC or in a root block) or errors that are not serious enough to require immediate corrective action (e.g., access consistency errors in an alternate index tree). In either case, you must press ENTER to invoke the next error message or summary display.
ERROR2	ERROR2 reports errors for which no greater level of detail can be provided by the VERIFY utility (e.g., "primary index tree cannot be fully verified"). From this display, you can either resume file verification by pressing ENTER or (depending on the error) terminate processing by pressing PF16.
ERROR3	ERROR3 reports errors for which VERIFY can provide a more detailed explanation. For example, when the error message "key inconsistency detected" appears, you can press PF1 to display the error details, press ENTER to continue file verification, or press PF16 to terminate processing.

28.5 THE SUMMARY DISPLAY

The VERIFY Summary screen displays summary information on the file just verified and recommends recovery procedures for the file if it is damaged. (Refer to Section 28.5.1 for more information on determining corrective action for a damaged file.) If you specify Yes in the Error Display field on the VERIFY Utility Options screen (Figure 28-2), VERIFY provides a Summary screen in addition to displaying the individual error messages display. The Summary screen for each file is displayed only after VERIFY detects and displays all errors in that file. To gain access to the Summary screen, press ENTER after each error message is displayed. From the Summary screen, press ENTER to verify the next file, if any. A sample VERIFY Summary screen is shown in Figure 28-4.

```
***  MESSAGE 0000 BY VERIFY

                    RESPONSE REQUIRED BY  PROGRAM  VERIFY
                          TO ACKNOWLEDGE SUMMARY


          File = @ADDDIR@   in Library = OSLIB      on Volume = WP

No errors were found in the file.



The file contains:        2   alternate indices
                       4236   records
                        268   data blocks
                          3   primary index blocks
                         85   alternate index blocks
                       1733   total blocks


Press ENTER to continue.
```

Figure 28-4. Sample VERIFY Summary Screen

28.5.1 Determining Corrective Action for a Damaged File

In addition to a summary of errors and other statistics, the Summary screen contains a recommended course of action to recover a damaged file (when applicable). There are three possible recommended actions:

1. For files with no errors or only access consistency errors, no immediate corrective action is necessary.
2. For certain types of index errors, running the COPY utility with the Reorg option repairs the error. (Refer to Chapter 3 for information about the COPY utility.)
3. For more serious errors, including a broken data chain, restoring the backup copy of the file or recreating the file manually are the only corrective actions available to you.

28.6 THE END-OF-JOB RESULTS

After the entire input file, library, or volume is processed, VERIFY displays the name(s) of the file, library, or volume verified, and a message informing you that VERIFY processing is completed. If you prematurely terminated VERIFY processing by pressing PF16, then the End-of-Job screen indicates that the verification of the input is incomplete. In either case, you can rerun the program by pressing PF1, or terminate processing by pressing ENTER or PF16.

28.7 GENERATING ERROR AND SUMMARY REPORTS

The VERIFY utility automatically produces print files containing a summary report and, if any errors are detected, an error report. The summary report cites the number of errors in the file, the number of records in the file, the number of data blocks, primary index blocks, and alternate index blocks allocated, the total number of unused blocks, and the total number of used blocks (the sum of the data, index, and alternate index blocks). Although the summary report lists each error code, the report does not contain the description of the code. (Refer to Section 28.7.1 for the error code descriptions and recommended course of action.) The error report identifies the file that is processed and lists the errors. Examples of the summary and error reports are shown in Figures 28-5 and 28-6.

```
WANG VS VERIFY PROGRAM - VERSION  x.xx.xx          SUMMARY REPORT                    12:04  09/04/85  PAGE 1

                                 PRIMARY  ALTERNATE                    PRIMARY  ALETRNATE  TOTAL
                          ERROR  INDEX    INDICES                 DATA  INDEX    INDEX      UNUSED   TOTAL
VOLUME  LIBRARY   FILE    CODE   ERRORS   ERRORS     RECORDS      BLOCKS  BLOCKS   BLOCKS     BLOCKS   BLOCKS

SYSTEM  DOCMNTR   BUGS    08         1        0      UNKNOWN      UNKNOWN UNKNOWN  UNKNOWN    UNKNOWN  UNKNOWN

END OF VERIFY SUMMARY REPORT.
```

Figure 28-5. Sample Summary Report

```
WANG VS VERIFY PROGRAM                          ERROR REPORT                        12:04  09/04/85  PAGE 1

VOLUME  LIBRARY   FILE      ERROR MESSAGE

SYSTEM  DOCMNTR   BUGS      Invalid block length was detected in      data block number      0.

END OF VERIFY REPORT.
```

Figure 28-6. Sample Error Report

28.7.1 Error Code Descriptions

VERIFY has no displayed return codes. In the printed Summary Report, however, VERIFY assigns an error code to each file. A description of the error codes are as follows:

Code	Description	Recommended Corrective Action
0	No errors detected.	None
1	Access inconsistency errors in alternate indexes only.	Copy/Reorg can restore damaged index path.
2	Alternate indexes cannot be verified (AXD1 block is bad.) Primary index structure in intact.	File must be restored from the backup volume. Not enough data is available for Copy/Reorg to reconstruct indexes.
4	Primary index errors found (FDR1 block or primary index block is bad). Data chain is intact.	Copy/Reorg can reconstruct index **unless** FDR1 block is bad, in which case, the file must be restored.
8	Data chain is broken.	File must be restored from a backup file.
16	File cannot be fully verified. VERIFY could not get enough information to diagnose the problem. The volume may have a bad VTOC.	You can run LISTVTOC to diagnose VTOC problems (refer to Chapter 17). In this case, the file must be restored from the backup or be recreated manually. Code 16 can also indicate that you terminated verification of the file.

28.8 A SAMPLE VERIFY PROCEDURE

You can control VERIFY processing through the VS Procedure language. A complete list of VERIFY GETPARMs is given in Appendix A. Refer to the *VS Procedure Language Reference* for details concerning VS Procedure language syntax.

The following example shows a procedure to run VERIFY to validate a library (PROGLIB on volume SYSTEM) without any workstation displays. If a file is found that cannot be validated, the procedure causes VERIFY to skip to the next file in the library. Any summary reports and error reports are written into a print file.

```
PROCEDURE
    RUN VERIFY
        ENTER OPTIONS RANGE=LIBRARY, LIBRARY=PROGLIB,
            VOLUME=SYSTEM, VERIFY=ALL, DISPLAY=NO
        ENTER EOJ 16
RETURN
```

APPENDIX A
SYSTEM UTILITY GETPARMS

A.1 INTRODUCTION TO GETPARMS

GETPARM processing is distinguished from other methods of processing runtime information primarily because it can interface with a procedure. GETPARM processing enables you to run system programs with little or no user interaction by supplying most or all of the required program parameter information.

During normal program execution, parameter information is specified interactively through GETPARM prompts. Using the VS Procedure language, you can create a procedure to specify the parameter information through the same GETPARM prompts. GETPARM prompts are the preferred source of information by VS programs and are used wherever possible to obtain parameter information. GETPARM prompts never appear on the workstation screen when they are satisfied by a Procedure language ENTER statement. Because of this, a procedure writer can explicitly control the interaction between a user and an executing program.

The VS Operating System supports a supervisor call (SVC) routine, the GETPARM SVC, that prompts you for (and accepts) runtime parameter information during normal program execution. GETPARM SVC-generated prompts are displayed on the workstation screen and await parameter information to be specified as necessary; they also display and await acknowledgement of runtime messages. All values that you specify either interactively or through a procedure are verified. If the values entered are not acceptable, the GETPARM SVC responds with an error message.

NOTE
There are no GETPARM prompts for the following utilities:

- *FORMCNTL*
- *POOLSTAT*
- *SHRSTAT*

A.2 THE STRUCTURE OF A GETPARM

Each GETPARM prompt in a program is identified by a parameter reference name (prname). The prname for each request is unique within that program. Programs generally observe certain conventions when identifying GETPARM prompts with prnames; for example, a GETPARM prompt soliciting information for an input file is usually identified with the prname INPUT, while a GETPARM soliciting parameters for an output file is identified with the prname OUTPUT.

Many GETPARM information prompts contain one or more modifiable fields into which you or a procedure can enter information. Each field is labeled with a keyword. When a GETPARM prompt is displayed, the keyword for each field provides a description of the information to be supplied for that field. Certain conventions are commonly used in keyword naming. For example, a prompt for a file name often uses the keyword FILE.

Many GETPARM prompts also solicit a PF key response (such as 16 = Exit Program). No keyword is associated with a PF key option; only the PF key number itself is specified.

Within a procedure, each ENTER or DISPLAY statement supplies parameters for a single GETPARM prompt. To associate a given ENTER or DISPLAY statement with a specific GETPARM prompt, you must specify the prname of the prompt in the statement (for example, ENTER INPUT).

GETPARM prompts that are issued by a user program are assigned any prname that the programmer chooses. For user-defined GETPARM prompts, modifiable fields and the keywords that identify them are specified by the person that issues the GETPARM.

When a procedure supplies field values, keywords in the ENTER or DISPLAY statement associate the specified values with the fields to which they are to be assigned in the GETPARM prompt (ENTER INPUT FILE = xxxx). Values associated with keywords in the procedure statement are passed to the corresponding keyword-identified fields in the GETPARM prompt. (Be sure to spell the keywords correctly in the procedure statement.) If the procedure does not assign new values to fields, they retain their default values.

The following procedure is a sample procedure that runs the COPYWP utility. If you compare this procedure to the list of prnames and keywords for COPYWP, you can see that certain default keyword values are used, since those keywords are not listed (for example, VOLUME for prname INPUT). Also, PF key options are selected for certain other prnames (for example, for prname FUNCTION, PF1, Copy a Single Document, was selected).

```
PROCEDURE FOR COPYWP
    RUN COPYWP
        ENTER FUNCTION 1
        ENTER INPUT DOCUMENT = 0001A
        ENTER OUTPUT DOCUMENT = 0002A, VOLUME=SYSTEM
        ENTER FUNCTION 16
    RETURN
```

Refer to the *VS Procedure Language Reference* for more information on the VS Procedure language and the use of GETPARM prompts. GETPARM prompts for the VS File Management utilities are not documented in this document, but are listed in the *VS File Management Utilities Reference*.

A.3 SYSTEM UTILITY GETPARM PROMPTS

This section contains a table for each utility. Each table contains a list of the prnames, keywords, the length of the keywords, options, and default values used by VS System Utility GETPARM prompts. The GETPARM tables are listed alphabetically by utility. The GETPARMs for each utility are listed in the sequence in which they appear for interactive processing.

The GETPARM tables also contain PF key options (if any) available for a given GETPARM prompt. The VS Procedure language does not define a screen's PF key option through a keyword but instead defines a PF key value by placing it after a prname. For example, the statement ENTER FUNCTION 1 is equivalent to pressing PF1 on a screen identified by the FUNCTION prname.

The ENTER key is also considered a PF key but cannot be explicitly specified. Listings of PF key options on the following pages refer to the ENTER key as b (blank), meaning that no PF key value is specified in the statement. The ENTER key is assumed if no other PF key value is specified in a statement that requires a PF key value.

Throughout the GETPARM tables, "PF Keys" is used in the Keyword column. PF Keys is NOT a required keyword but is used to signify the presence of PF key options.

Table A-1. Compress-in-Place (CIP) Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	VOLUME	6		
	(PF Keys)		16 = End Processing	
	VSID	3	(0–255)	0
MOUNT	DEVICE	3		
	(PF Keys)		1 = Return to Specify Input	

Table A-2. COPY Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	COPY	7	FILE, LIBRARY, VOLUME	FILE
	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	MODE	6	INPUT, SHARED	INPUT
LOCK	LOCK	3	YES, NO	YES
	TIMEOUT	3	0–255	10
	BYPASS	3	YES, NO	NO

(continued)

Table A-2. COPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
OPTIONS	FILEORG	1	C, I, R	Input value
	LENGTH	1	F, V	Input value
	COMPRESS	1	Y, N	Input value
	REORG	3	YES, NO	NO
	KEYLEN	3	1–999	Input value
	KEYPOS	5	1–99999	Input value
	IPACK	3	1–100	Input value
	DPACK	3	1–100	Input value
	RECBLK	1	A, N, U	A for DMS/TX files, else N
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	RECORDS	7	1–9999999	Input value
	RETAIN	3	0–999	
	RELEASE	3	YES, NO	YES for consecutive file; NO for indexed files.
	FILECLAS	1	A–Z, #, b, @, $	For indexed file, same as input.
	DEVICE	11	DISK, PRINTER	DISK
	PRTCLASS	1	A–Z	A
	FORM#	3	0–255	000
	COPIES	5	1–32,767	1
OPTIONS (allows duplicate file names)	OPTION	1	N, S, R, C	N
	NEWNAME	8		
EOJ	(PF Keys)		1 = Rerun COPY 16 = End processing	

Table A-3. COPYOIS Utility GETPARMs

Prname	Keyword	Length	Options	Default
MENU	(PF Keys)		1 = Initialize diskette 2 = Copy File from VS to diskette 4 = Copy/Convert OIS files to VS 5 = Create a diskette catalog listing 7 = Rename a file on diskette 8 = Delete a file from diskette 9 = Assign a diskette Volume/File password 12 = Display the file conversion listing (with PF4) 14 = Display the diskette catalog listing (with PF5) 16 = End Processing	
Initialize Diskette — PF1				
VOLUME	VOLUME PASSWORD (PF Keys)	8 8	OIS Volume Name Legal OIS password b = Continue 1 = Return to main menu	
MOUNT (if not already mounted)	VOLUME DEVICE UNIT PASSWORD (PF Keys)	6 8 3 8	VS Volume Name None When applicable b = Continue 1 = Return to previous screen	DISKETTE

(continued)

Table A-3. COPYOIS Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Copy File from VS to Diskette — PF2				
VSFILE	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	(PF Keys)		b = Continue 1 = Terminate this file copy 9 = Multiple file copy	
OISFILE (1 for each file)	VOLUME	8		
	FILE	63	OIS file name	
	(PF Keys)		b = Continue 1 = Respecify input	
MOUNT (if not already mounted)	VOLUME	6	OIS Volume Name	
	DEVICE	8	DISK or DISKETTE	
	UNIT	3		
	PASSWORD	8	When applicable	
	(PF Keys)		b = Continue 1 = Return to previous screen	
Copy/Convert OIS file to VS — PF4				
INPUT	VOLUME	8	OIS or VS Volume Name	User's INVOL
	FILE	63	VS, OIS	VS
	DEVICE	3		
	CONVERT	3	YES or NO	YES
	(PF Keys)		b = Continue 1 = Return to main menu 9 = Multiple file copy/convert	
MOUNT (if not already mounted)	VOLUME	6	OIS Volume Name	
	DEVICE	8	DISK or DISKETTE	
	UNIT	3		
	PASSWORD	8	When applicable	
	(PF Keys)		b = Continue 1 = Return to previous screen	
OPTIONS	FORMAT	8	FIXED, VARIABLE, KEYED	FIXED
	LENGTH	4		
	NUMERICS	8	INTEGER, FLOATING, PACKED	
	DECIMALS	2	Decimal Alignment for Packed Numeric Format Only	
	KEYFIELD	3	YES or NO	
	KEYSTART	4	Keyfield Start Position	
	(PF Keys)		b = Continue 1 = Respecify input 13 = Instructions	

(continued)

Table A-3. COPYOIS Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Copy/Convert OIS file to VS — PF4 (continued)				
DATAFILE	VOLUME	8	Invol, outvol or input file volume	
	FILE	63		
	DEVICE	3	OIS or VS	
	(PF Keys)		b = Continue 1 = Return to main menu	
OUTPUT	FILE	8		
	LIBRARY	8		
	VOLUME	6	User's OUTVOL	
	(PF Keys)		b = Continue 1 = Respecify input	
Create a Diskette Catalog Listing — PF5				
CATALOG	VOLUME	8		
	(PF Keys)		b = Continue 1 = Return to main menu	
MOUNT (if not already mounted)	VOLUME	6	OIS Volume Name	
	DEVICE	8	None	DISKETTE
	UNIT	3		
	PASSWORD	8	When applicable	
	(PF Keys)		b = Continue 1 = Return to previous screen	
Rename a Diskette File — PF7				
RENAME	VOLUME	8		
	FILE	63		
	(PF Keys)		b = Continue 1 = Return to previous screen	
MOUNT (if not already mounted)	VOLUME	6	OIS Volume Name	
	DEVICE	8	None	DISKETTE
	UNIT	3		
	PASSWORD	8	When applicable	
	(PF Keys)		b = Continue 1 = Return to previous screen	
NEWNAME	FILE	63		
	(PF Keys)		b = Continue 1 = Return to previous screen	

(continued)

Table A-3. COPYOIS Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Delete a file from Diskette — PF8				
DELETE	VOLUME FILE (PF Keys)	8 63	 b = Continue 1 = Return to previous screen	
MOUNT (if not already mounted)	VOLUME DEVICE UNIT PASSWORD (PF Keys)	6 8 3 8	OIS Volume Name None When applicable b = Continue 1 = Return to previous screen	 DISKETTE
PASSWORD	VOLUME NAME PASSWORD (PF Keys)	8 63 8	 b = Continue 1 = Return to main menu	
MOUNT (if not already mounted)	VOLUME DEVICE UNIT PASSWORD (PF Keys)	6 8 3 8	OIS Volume Name None When applicable b = Continue 1 = Return to previous screen	 DISKETTE

Table A-4. COPY2200 Utility GETPARMs

Prname	Keyword	Length	Options	Default
ACTION	(PF Keys)		1 = Create VS files from files on a 2200 diskette or in a VS image file 2 = Create a 2200 diskette or VS image file from VS files 9 = Create a VS image file from a diskette(s) 10 = Create a diskette(s) from a VS image file 16 = End processing	

(continued)

Table A-4. COPY2200 Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Create VS Files from Files on a 2200 Diskette or in a VS Image File — PF1				
INPUT	DISKETTE	6		
	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	STARTER	1	A–Z, 0–9, $, @	V
	FILLER	1	A–Z, 0–9, #, $, @	#
	DEVICE#	3		
	(PF Keys)		b = Continue 1 = Return to main menu 4 = Mount input volume	
COPYMODE	MODE	1	A, D, P, L	
	(PF Keys)		b = Continue 1 = Return to Input Definition screen	
FILELIST	(Keywords are names of files)	1	Blank, nonblank	blank
	(PF Keys)		b = Continue 1 = Return to Copy Mode screen	
PROGOUT	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	FILECLAS	1	A–Z, # , b, $, @	User's FILECLAS
	CONVERT	3	YES, NO	YES
	COMPRESS	3	YES, NO	YES
DATAOUT	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	FILECLAS	1	A–Z, #, b, $, @	Users FILECLAS
	TYPE	1	F, V, T, X, E, S, N	
	COMPRESS	3	YES, NO	YES
	RECSIZE	4	0–2048 (required only if TYPE = F, V, T, or X)	
	TRANSL	3	YES,NO (required only if TYPE = T or X)	NO
	(PF Keys)		b = Continue 14 = Display information	
DATAINFO	(PF Keys)		b = Display other information screen 1 = Return to Data Files Output screen	
NUMERICS	TYPE	1	D, F, B, H, P, N	
	LENGTH	2	1–16	
	PLACES	2	0–99	

(continued)

Table A-4. COPY2200 Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Create a 2200 Diskette or VS Image File from VS Files — PF2				
OUTPUT	DISKETTE	6		
	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	FILECLAS	1	A–Z, #, b, $, @	User's FILECLAS
	COMPRESS	3	YES, NO	YES
	SECTORS	5	1–32767, ALL	1024
	INDEX	3	1–255	16
	DEVICE#	3		
	(PF Keys)		b = Continue 1 = Return to main menu 4 = Mount output volume	
CONTINUE	CONTINUE	3	YES, anything else	
INPUT	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	DEVICE#	3		
	(PF Keys)		b = Continue specifying files 4 = Mount input volume	
FILETYPE	TYPE	1	T, X, E, S	
	FILENAME	8		Input file name
	RECSIZE	4	1–9999	
	TRANSL	3	YES, NO	NO
Create a VS Image File from a Diskette(s) — PF9				
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	FILECLAS	1	A–Z, #, b, $, @	User's FILECLAS
	COMPRESS	3	YES, NO	YES
	TYPE	1	T, N	T
	MULTIPLE	3	YES, NO	NO
	SECTORS	5	USED, ALL, 1–32767	ALL
	DEVICE#	3		
	(PF Keys)		b = Continue 1 = Return to main menu 4 = Mount output volume	

(continued)

Table A-4. COPY2200 Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Create a VS Image File from a Diskette(s) — PF9 (continued)				
INPUT	DISKETTE SECTORS DEVICE# (PF Keys)	6 4 3	 ALL, 1–3874 (SECTORS appears only if MULTIPLE = YES) b = Continue 1 = Return to Output Definition screen 4 = Mount diskette	
Create a Diskette(s) from a VS Image File — PF 10				
INPUT	FILE LIBRARY VOLUME TYPE MULTIPLE DEVICE# (PF Keys)	8 8 6 1 3 3	 T, N YES, NO b = Continue 1 = Return to main menu 4 = Mount input volume 16 = Copy no more files	 User's INLIB User's INVOL T NO
OUTPUT	DISKETTE SECTORS DEVICE# (PF Keys)	6 4 3	 ALL, 1–3874 (SECTORS appears only if MULTIPLE = YES) b = Continue 1 = Return to Input Definition screen 4 = Mount diskette	
CONTINUE	CONTINUE	3	YES, anything else	

Table A-5. COPYWP Utility GETPARMs

Prname	Keyword	Length	Options	Default
FUNCTION	(PF Keys)		1 = Copy Single Document 3 = Copy Document Library 4 = Delete Single Document 6 = Delete Document Library 7 = Rename Single Document 9 = Rename Document Library 10 = Reorganize Single Document 12 = Reorganize Document Library 13 = Convert Document to VS File 14 = Convert VS File to Document 15 = Document Merge 16 = Terminate Processing	
INTERNAT	DATE	1	A, E	A–if the system is generated with American dates E–if the system is generated with European dates
	DECALIGN	1	'.' or ','	'.'–American ','–European
	CURRENCY	1	Local currency symbols	'$'–American
	REQSPACE	2	Any hexadecimal value in the ASCII character set	5C
	DEVCHARS	3	YES, NO	YES
Copy or Rename Single Document — PF1 or PF7				
INPUT	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
PASSWORD	PASSWORD	6		
	(PF Keys)		b = Continue 1 = Return to main menu	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Copy or Rename Single Document — PF1 or PF7 (continued)				
OUTPUT	DOCUMENT	5	Valid document IDs, including next and NEXT	
	VOLUME	6		Document library's standard volume if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
Copy or Rename Document Library — PF3 or PF9				
INPUT	LIBRARY	1		
	VOLUME	6		Library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
OUTPUT	LIBRARY	1		
	VOLUME	6		Library's standard volume (if WP is installed)
	RENUMBER	3	YES, NO	NO
	DOCUMENT	4	Four digits	0001
	DUPFILES	6	PROMPT, NOCOPY, DELETE	PROMPT
	(PF Keys)		b = Continue 1 = Return to main menu	
PASSWORD	PASSWORD	6		
	(PF Keys)		b = Continue 1 = Return to main menu 2 = Bypass current document	
SAMEFILE	OPTION	1	N, D, R, C,	N
	NEWDOCID	4	Four digits, next, NEXT	
	(PF Keys)		b = Continue 1 = Return to main menu	
ERROR	(PF Keys)		1 = Return to main menu 2 = Skip current document (not available for some error conditions)	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Delete Single Document — PF4				
INPUT	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	ERASE (PF Keys)	3	YES, NO b = Continue 1 = Return to main menu	NO
PASSWORD	PASSWORD (PF Keys)	6	When applicable b = Continue 1 = Return to main menu	
Delete Document Library — PF6				
INPUT	LIBRARY	1		
	VOLUME	6		Library's standard volume (if WP is installed)
	ERASE (PF Keys)	3	YES, NO b = Continue 1 = Return to main menu	NO
PASSWORD	PASSWORD (PF Keys)	6	b = Continue 1 = Return to main menu 2 = Bypass current document	
ERROR	(PF Keys)		1 = Return to main menu 2 = Skip current document (not available for some error conditions)	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Reorganize Single Document — PF10				
INPUT	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
PASSWORD	PASSWORD (PF Keys)	6	When applicable b = Continue 1 = Return to main menu	
Reorganize Document Library — PF12				
INPUT	LIBRARY	1		
	VOLUME	6		Library's standard volume (if WP is installed)
	RENUMBER	3	YES, NO	NO
	DOCUMENT	4	Four digits, excluding next and NEXT	0001
	(PF Keys)		b = Continue 1 = Return to main menu	
ERROR	(PF Keys)		1 = Return to main menu 2 = Skip current document (not available for some error conditions)	
Convert Document to VS File — PF13				
INPUT	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
PASSWORD	PASSWORD (PF Keys)	6	b = Continue 1 = Return to main menu	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Convert Document to VS File — PF13 (continued)				
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	TYPE	6	DATA, PRINT, SOURCE, TC	
	(PF Keys)		b = Continue 1 = Return to main menu	
PRINT (only for Type = Print)	START	3		Value from last document print request
	FINISH	3		Total number of pages in document
	NUMBER	4		Value from last document print request
	HEADER	3		Value from last document print request
	FOOTER	3		Value from last document print request
	LINE	2		Value from last document print request
	MARGIN	3		Value from last document print request
	FORMAT	11	JUSTIFIED, UNJUSTIFIED, NOTES	Value from last document print request
	STYLE	5	FINAL, DRAFT	Value from last document print request
	SUMMARY	5	PRINT, OMIT	Value from last document print request
	(PF Keys)		b = Continue 1 = Return to main menu	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Convert VS File to Document — PF14				
INPUT	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	(PF Keys)		b = Continue 1 = Return to main menu	
OUTPUT	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume if WP is installed)
	TITLE	25		Input file's document title (TC file only)
	OPERATOR	20		Input file's document operator (TC file only)
	AUTHOR	20		Input file's document author (TC file only)
	COMMENTS	20		Input file's document comments (TC file only)
	TYPE	6	SOURCE, IMAGE (required only for 80-byte record input files)	SOURCE
	TABS	3	YES, NO (required only for Print and Source input files)	YES
	PAGE	3	1–255 (required only for Source inputfiles)	User's current LINES
	(PF Keys)		b = Continue 1 = Return to main menu	
OPTIONS (only for Type = Image)	TABS	59	Up to 15 sets of 1- to 3-digit numbers, separated by commas or spaces	
	LENGTH	3		The smaller value of either the file's maximum record length plus 1, or 158
	RETURN	3	YES, NO	YES
	TRUNCATE	3	YES, NO	YES
	(PF Keys)		b = Continue 1 = Return to main menu	

(continued)

Table A-5. COPYWP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Document Merge — PF15				
PRIMARY	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
SECOND	DOCUMENT	5	Valid document IDs, excluding next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	
PASSWORD	PASSWORD (PF Keys)	6	b = Continue 1 = Return to main menu	
OUTPUT	DOCUMENT	5	Valid document IDs, including next and NEXT	
	VOLUME	6		Document library's standard volume (if WP is installed)
	(PF Keys)		b = Continue 1 = Return to main menu	

Table A-6. DISKINIT Utility GETPARMs

Prname	Keyword	Length	Options	Default
FUNCTION	FUNCTION	10	INITIALIZE, REFORMAT, RELABEL, VERIFY, REMOVE, ERASE	VERIFY
	VOLUME	6		
	VSID	3	0–255	0

(continued)

Table A-6. DISINIT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Initialize Function				
ACTION (ONLY if input volume is NL)	(PF Keys)		b = Continue 1 = Go to EOJ	
INPUT	NEWVOL	6		Volume name specified on Function menu
	VSID	3	0–255	0
	LABEL	2	SL, NL	SL
	TOLERATE	5	NONE, CRASH, MEDIA	NONE
	VTOCSIZE	3		Standard size of VTOC for input disk volume type
	OWNER	14		
	DUMPFILE	3	YES, NO	NO
	PAGEPOOL	3	YES, NO	NO
	PASSES	6	BRIEF, NORMAL	BRIEF
	(PF Keys)		b = Continue 1 = Go to EOJ	
BADBLKKL (if the volume was previously initialized and already has bad blocks)	(PF Keys)		4 = Retain current bad block list 6 = Retain current bad block list and add any new ones to it 8 = Discard current bad block list and build a new one	
EOJ	(PF Keys)		1 = Rerun DISKINIT 16 = End processing	
Reformat Function				
ACTION (only if input volume is NL)	(PF Keys)		b = Continue 1 = Go to EOJ	

(continued)

Table A-6. DISKINIT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Reformat Function				
INPUT	NEWVOL	6		Volume name specified on Function menu
	VSID	3	0–255	0
	TOLERATE	5	NONE, CRASH, MEDIA	NONE
	VTOCSIZE	4		Standard size of VTOC for input disk volume type
	DUMPFILE	3	YES, NO	YES if volume has a dump file; NO if not
	PAGEPOOL	3	YES, NO	YES if volume has a page block or pool; NO if not
	XTNTOPEN	3	3–255	3
	XTNTTOTL	9	3–255 (for single volumes; Multivolume files can have up to 999999999 or 0 for unlimited extents)	13
EOJ	(PF Keys)	14	1 = Rerun DISKINIT 16 = End processing	
Relabel Function				
INPUT	NEWVOL	6		
	PAGEPOOL	3	YES, NO	YES if volume has a page block or pool; NO if not
	XTNTOPEN	3	3–255	3
	XTNTTOTL	9	3–255 (for single volumes; Multivolume files can have up to 999999999 or 0 for unlimited extents)	13
	(PF Keys)		b = Continue 1 = Go to EOJ	
EOJ	(PF Keys)	6	1 = Rerun DISKINIT 16 = End processing	
Verify Function				
ACTION (only if input volume is NL)	(PF Keys)		b = Continue 1 = Go to EOJ	
INPUT	(PF Keys)		b = Continue 1 = Go to EOJ	
EOJ	(PF Keys)		1 = Rerun DISKINIT 16 = End processing	

(continued)

Table A-6. DISKINIT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Remove Function				
ACTION (only if input volume is NL)	(PF Keys)		b = Continue 1 = Go to EOJ	
INPUT	BLOCK (PF Keys)	6	b = Continue 1 = Go to EOJ	
EOJ	(PF Keys)		1 = Rerun DISKINIT 16 = End processing	
Erase Function				
ACTION (ONLY if input volume is NL)	(PF Keys)		b = Continue 1 = Go to EOJ	
INPUT	NEWVOL	6		Volume name specified on Function menu
	VSID	3	0–255	0
	LABEL	2	SL, NL	SL
	TOLERATE	5	NONE, CRASH, MEDIA	NONE
	VTOCSIZE	3		Standard size of VTOC for input disk volume type
	OWNER	14		
	DUMPFILE	3	YES, NO	NO
	PAGEPOOL	3	YES, NO	NO
	(PF Keys)		b = Continue 1 = Go to EOJ	
EOJ	(PF Keys)		1 = Rerun DISKINIT 16 = End processing	
Allocate a Dump File (Only if DUMPFILE = YES)				
DUMPFILE	SIZE	5	0–65536; entry should correspond to size of main memory	Size of the current dump file (if it exists)
Allocate a Page Pool (Only if PAGEPOOL = YES)				
PAGEPOOL	SIZE	5	0–65536	Size of the current page pool if it exists
	LOCATION	1	0–9	Location of the current page pool
POOLSIZE	(PF Keys)		b = Continue	

Table A-7. DISPLAY Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	ACCESS	6	RECORD, BLOCK, PRINT	RECORD
	MODE	6	INPUT, SHARED	INPUT
LOCK	LOCK	3	YES, NO	NO
	TIMEOUT	3	0–255	10
	BYPASS	3	YES, NO	NO
EOJ	(PF Keys)		1 = Display another file 16 = End processing	
PRINT	PRINT is a default GETPARM; refer to Section A.4.			

Table A-8. EZFORMAT Utility GETPARMs

Prname	Keyword	Length	Options	Default
OPTIONS	LANGUAGE	10	ASSEMBLER, A, BASIC, B, COBOL, C, DATA(ENTRY), D, MENU, M, RPG, R	
	PROCEDUR	3	YES, NO	YES
	(PF Keys)		2 = Define new screen format 3 = Modify existing screen format	
INSCREEN	FILE	8		
	LIBRARY	8		User ID + SAVE
	VOLUME	6		User's INVOL
FUNCTION	(PF Keys)		1 = Return to Input Options screen 2 = Save only generated output 3 = Save only screen contents 4 = Save screen contents and generated output 16 = Exit without saving any files	
SCREEN	FILE	8		
	LIBRARY	8		User ID + SAVE
	VOLUME	6		User's OUTVOL
SOURCE (Menu option only)	(PF Keys)		2 = Assembly language 3 = RPG II	

(continued)

Table A-8. EZFORMAT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SOURCE (Data Entry Option only)	(PF Keys)		1 = COBOL 3 = RPGII 16 = Return	
CONTROL (Data Entry option only)	FILE LIBRARY VOLUME (PF Keys)	8 8 6	b = Continue 1 = Switch to COBOL option 2 = Invoke CONTROL 16 = Exit EZFORMAT	User ID + CTL User's INVOL
COPYLIBR	FILE LIBRARY VOLUME	8 8 6		User ID + COPY User's INVOL
PROGRAM	FILE LIBRARY VOLUME	8 8 6		User's OUTLIB User's OUTVOL
FIELDS (only BASIC, COBOL, and DATA ENTRY-COBOL Options)	FILE LIBRARY VOLUME (PF Keys)	8 8 6	ENTER = Use an existing field name file 2 = Create a field name file	Inscreen file User ID = FLD User's INVOL

Table A-9. FASTLINK Utility GETPARMs

Prname	Keyword	Length	Options	Default
FUNCTION	FILE LIBRARY VOLUME (PF Keys)	8 8 6	1 = Display perm-open files 2 = Edit program name file (leave blank for new file) 3 = Set programs perm-open 4 = Reset perm-open programs	

(continued)

Table A-9. FASTLINK Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
OPENONE	FILE LIBRARY VOLUME (PF Keys)	8 8 6	 b = Set program perm-open 16 = Return to display mode of perm-open files	
SELECT	(PF Keys)		b = Set all programs perm-open 1 = Select programs to be perm-open 16 = Return to main menu	
SAVEIT	(PF Keys)		1 = Return to edit mode 16 = Exit to main menu	

Table A-10. FLOPYDUP Utility GETPARMs

Prname	Keyword	Length	Options	Default
OPTIONS	OPTION (PF Keys)	1	C, D, G b = Continue 16 = End processing	
Copy a Diskette to a Diskette Image File — Option C				
INPUT	VOLSER LABEL DEVICE	6 2 3	 SL, NL	 SL
OUTPUT	FILE LIBRARY VOLUME RECORDS RETAIN RELEASE FILECLAS DEVICE	8 8 6 7 3 3 1 11	 YES, NO DISK	 User's OUTLIB User's OUTVOL 154 YES DISK
EOJ	(PF Keys)		1 = Rerun FLOPYDUP 16 = End processing	

(continued)

Table A-10. FLOPYDUP Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Duplicate Diskette — Option D				
INPUT	VOLSER LABEL DEVICE	6 2 3	 SL, NL	 SL
OUTPUT	VOLSER DEVICE	6 3		
EOJ	(PF Keys)		1 = Rerun FLOPYDUP 2 = Generate another diskette with the same data 16 = End processing	
Generate a Diskette from a Diskette Image File — Option G				
INPUT	FILE LIBRARY VOLUME DEVICE	8 8 6 11	 DISK, NONE	 User's INLIB User's INVOL DISK
OUTPUT	VOLSER DEVICE	6 3		
EOJ	(PF Keys)		1 = Rerun FLOPYDUP 2 = Generate another diskette with the same data 16 = End processing	

Table A-11. FONTCNTL Utility GETPARMs

Prname	Keyword	Length	Options	Default
FONTMAIN	(PF Keys)		1 = Review Fonts 2 = Install Font[a] 3 = Delete Font[a] 4 = Change Font[a] 15 = Print Catalog 16 = Exit	
DEVSCRN	DEVICE	8	Supported printers	First font in the screen list
FONTREVU	(PF Keys)		1 = Select New Device 4 = Previous (if enabled) 5 = Next (if enabled) 2 = File Information 16 = Main screen	

(continued)

Table A-11. FONTCNTL Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
FONTRVWX	(PF Keys)		1 = Select New Device 2 = Description Menu 4 = Previous (if enabled) 5 = Next (if enabled) 16 = Main screen	
FONTADD	FILE	8		
	LIBRARY	8		
	VOLUME (Only if not an OIS file)	6		
	OISFILE (Only if not a VS file)	52		
	FONTNUM	3	1–999	1
	FORMAT	3	VS or OIS	VS
	DEFAULT	3	YES or NO	NO
	GRAPHICS	3	YES or NO	NO
	NATCODE	3		Obtained from GENEDIT at system IPL time
	DEVICE	8	Supported printers	Device selected
	(PF Keys)		b = Continue 1 = Return 2 = Select National Code 16 = Main Screen	
FONTCOPY	FONTFILE	8		Input Selected
	FONTLIB	8		Input Selected
	XTABLE	8		Same as FONTFILE
	XLIB	8		Same as FONTLIB
	SYSVOL	6		System volume
	DESCRIPT	50		Taken from the font file data
	(PF Keys)		b = Continue 1 = Respecify Input 16 = Main screen	
FONTSCCS	(PF Keys)		1 = Install More Fonts 16 = Main screen	
FONTDEL	FONTNUM	3	Any font number from 1–999 that is already on the system	
	(PF Keys)		1 = Select New Device 8 = Delete 16 = Main screen	

(continued)

Table A-11. FONTCNTL Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
FONTMOD	(PF Keys)		1 = Return to Main Menu 2 = Change Font Number 3 = Change Default Font 4 = Modify Font Translation Table 16 = Exit	
FONTMN	OLDFONT	3	1–999	1
	NEWFONT	3	1–999	1
	DEVICE	8	Supported printers	Device Selected
	(PF Keys)		b = Continue 1 = Return 16 = Main Screen	
FONTMDFN	DFLTFONT	3	Any font number from 1–999 that is already on the system	1
	(PF Keys)		1 = Return 7 = Modify 16 = Main screen	
FONTPTCH	FONTNUM	3	Any font number from 1–999 that is already on the system	
	(PF Keys)		b = Continue 1 = Return 16 = Main screen	
PTCHXLAT	FONTNUM	3	Any unique font number from 1–999	1
	DESCRIPT	50		Taken from the font file data
	(PF Keys)		1 = Return 16 = Main screen	
PTCHSCCS	(PF Keys)		1 = Return 16 = Main screen	

* You must have valid privileges for these options

Table A-12. IBMCOPY Utility GETPARMs

Prname	Keyword	Length	Options	Default
FUNCTION	(PF Keys)		1 = Copy files from IBM to VS 2 = Copy files from VS to IBM 3 = Initialize diskette to IBM format 4 = Mount IBM diskette 5 = Display an IBM diskette directory 16 = End processing	
IBM to VS Copy — PF1				
Options	COPY	6	VOLUME, SELECT, FILE	VOLUME
	VOLUME	6		
	FILE	8	Valid IBM file name	
	(PF Keys)		b = Continue 1 = Return to the main menu	
MOUNT	DEVICE	3		
	(PF Keys)		b = Continue 1 = Return to IBM to VS Input Definition screen	
INPUT (COPY = SELECT only)	FILE0001 through FILE0020	1	A nonblank character selects the file	blank
	(PF Keys)		b = Continue 1 = Return to IBM to VS Input Definition screen 16 = End input definition	
OUTPUT	LIBRARY	8		
	VOLUME	6		
	TRANSL	3	YES, NO	YES
	(PF Keys)		b = Continue 1 = Return to IBM to VS Input Definition screen 2 = Specify output options separately for each file 16 = Return to the main menu	

(continued)

Table A-12. IBMCOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
IBM to VS Copy — PF1 (continued)				
IOUTPUT	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	TRANSL	3	YES, NO	YES
	RECORDS	7	1 to 9999999	Number of input file records
	RELEASE	3	YES, NO	YES
	FILEORG	1	C, I	C
	COMPRESS	3	YES, NO	YES
	FILECLAS	1	A–Z, $, #, @, b	User's FILECLAS
	KEYLEN	3		
	KEYPOS	5		
	IPACK	3	0–100	100
	DPACK	3	0–100	100
	(PF Keys)		b = Continue 16 = Cancel and go to EOJ screen	
NEWFILE	FILE	8		
	(PF Keys)		b = Continue 2 = Skip the current file 3 = Scratch the existing file 5 = Add new records to the existing file 16 = Cancel and go to the EOJ screen	
NEWVOL	VOLUME	6		
	(PF Keys)		b = Continue 2 = Skip the current file 16 = Cancel and go to the EOJ screen	
BADIO	(PF Keys)		b = Acknowlege error and go to EOJ screen	
CANCEL	(PF Keys)		b = Acknowlege error and go to EOJ screen	
EOJ	(PF Keys)		1 = Return to IBM to VS Input Definition screen 2 = Return to the main menu 16 = End processing	

(continued)

Table A-12. IBMCOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
VS to IBM Copy — PF2				
OPTIONS	COPY	6	LIBRARY, SELECT, FILE	LIBRARY
	VOLUME	6		
	FILE	8	Valid IBM file name	
	(PF Keys)		b = Continue 1 = Return to the main menu	
INPUT (COPY = SELECT only)	FILE0001 through FILE0020[a]	1	A nonblank character selects the file	blank
	(PF Keys)		b = Continue 1 = Return to VS to IBM Input Definition screen 16 = End input definition	
OUTPUT	VOLUME	6		
	PADCHAR	1	blank, 0	blank
	TRANSL	3	YES, NO	YES
	(PF Keys)		b = Continue 1 = Return to VS to IBM Input Definition screen 2 = Specify output options separately for each file 16 = Return to EOJ screen	
MOUNT	DEVICE	3		
	(PF Keys)		b = Continue 1 = Return to Output Definition screen	
IOUTPUT	FILE	8		
	VOLUME	6		
	PADCHAR	1	blank, 0	blank
	TRANSL	3	YES, NO	YES
	EXTRA	3	000–999	000
	(PF Keys)		b = Continue 16 = Go to EOJ screen	
CLEAROUT	NEWINDEX	3	YES, NO	NO
	(PF Keys)		b = Continue 16 = Cancel and go to EOJ screen	
BIGRECS	(PF Keys)		b = Continue 2 = Skip current file 16 = Cancel/Go to EOJ	

(continued)

Table A-12. IBMCOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
VS to IBM Copy — PF2 (continued)				
ABORT	(PF Keys)		b = Acknowledge error and go to the EOJ screen	
FULLVOL	VOLUME (PF Keys)	6	b = Continue 2 = Skip current file and copy 3 = Copy and create a multivolume file if needed 16 = Cancel and go to the EOJ screen	
DUPLNAM	FILE (PF Keys)	8	b = Continue 2 = Skip current file and copy 3 = Scratch existing file on output volume 16 = Cancel and go to the EOJ screen	
NEWFILE	(PF Keys)		2 = Skip current file 16 = Cancel and go to the EOJ screen	
NEWVOL	VOLUME (PF Keys)	6	b = Continue 2 = Skip the current file 16 = Cancel and go to the EOJ screen	
BADIO	(PF Keys)		b = Acknowledge error and go to the EOJ screen	
CANCEL	(PF Keys)		b = Acknowledge error and go to the EOJ screen	
EOJ	(PF Keys)		1 = Return to VS to IBM Input Definition screen 2 = Return to the main menu 16 = End processing	

(continued)

Table A-12. IBMCOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
Initialize Diskette to IBM Format — PF3				
VOLUME	VOLUME (PF Keys)	6	b = Continue 1 = Return to the main menu	
MOUNT	DEVICE (PF Keys)	3	b = Continue 1 = Return to the Define Volume screen	
ONESIDED	(PF Keys)		b = Initialize diskette 1 = Return to the Define Volume screen	
TWOSIDED	FORMAT (PF Keys)	5	BASIC, H b = Initialize diskette 1 = Return to the Define Volume screen	BASIC
BADIO	(PF Keys)		b = Acknowledge error and go to the EOJ screen	
CANCEL	(PF Keys)		b = Acknowledge error and go to the EOJ screen	
EOJ	(PF Keys)		1 = Return to the Define Volume screen 2 = Return to the main menu 16 = End processing	
Mount an IBM Diskette — PF4				
MOUNTCOM	VOLUME DEVICE (PF Keys)	6 3	b = Continue 1 = Return to the main menu	
BADIO	(PF Keys)		b = Acknowledge error and return to the main menu	
CANCEL	(PF Keys)		b = Acknowledge error and return to the main menu	
Display or Print an IBM Diskette Directory — PF5				
VOLUME	VOLUME DEVICE (PF Keys)	6 7	DISPLAY, PRINTER b = Continue 1 = Return to the main menu	DISPLAY

[a] These names are changed to the actual file names.

Table A-13. IOELOG Utility GETPARMs

Prname	Keyword	Length	Options	Default
FUNCTION	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	(PF Keys)		1 = Analyze contents of I/O Error Log File 2 = Print contents of I/O Error Log File 13 = Help 16 = Terminate Processing	
RANGE	STARTDAY	8		Defaults to first and last records contained in the I/O Error Log file.
	STARTTIME	5		
	ENDDAY	8		
	ENDTIME	5		
	(PF Keys)		1 = Return to Function Selection menu	

Table A-14. IOTRACE Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	LOWDEV	3		System-generated
	HIGHDEV	3		System-generated
	IOSWTRAP	16	0–9, A–F, X	XXXXXXXXX...X
	CIOTRAP	16	0–9, A–F, X	XXXXXXXXX...X
	SIOTRAP	16	0–9, A–F, X	XXXXXXXXX...X
	HIOTRAP	16	0–9, A–F, X	XXXXXXXXX...X
	VOLUME	6		System volume
	TRAPREQ	3	0–999	1
	(PF Keys)		16 = End processing	
CCTRAPS	SIODB	1	Y, N	N
	SIOIB	1	Y, N	N
	SIOIN	1	Y, N	N
	CIODB	1	Y, N	N
	CIOIB	1	Y, N	N
	CIOIN	1	Y, N	N
	HIODB	1	Y, N	N
	HIOIB	1	Y, N	N
	HIOIN	1	Y, N	N
	(PF Keys)		1 = Return	

Table A-15. LISTVTOC Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	VOLUME	6	Valid library or spaces	blanks
	VSID	3	0–255	0
	FILES	3	YES, NO	NO
	VTOCMAP	3	YES, NO	NO
	SCREEN	3	YES, NO	YES
	PRINT	2	4–99	55

Table A-16. OISCART Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	VOLUME	6		
	DEVICE	3		
	(PF Keys)		b = Continue 16 = End processing	
COPYMODE	MODE	1	A, L	A
	(PF Keys)		b = Continue 1 = Respecify input	
OUTPUT	VOLUME	6		User's OUTVOL
	(PF Keys)		b = Continue 1 = Respecify copy mode	
CONFLICT	ACTION	1	P, D, F	P
	(PF Keys)		b = Continue 1 = Respecify output	
DUPDOCID	DOCID	5		Next ID in Library
	(PF Keys)		b = Assign new ID 3 = Delete old document	
CONTVOL	VOLUME	6		
EOJ	(PF Keys)		b = End processing 1 = Respecify input 9 = View log file 16 = End processing	

Table A-17. PATCH Utility GETPARMs

Prname	Keyword	Length	Options	Default
IOFILE	FILE	8		
	LIBRARY	8		
	VOLUME	6		
	DEVICE	11		
COMMAND	OPTION	7	DUMP, VERIFY, REPLACE, QUIT	
	ADDRESS	5		
	BLOCK	8		
	LENGTH	6		
	DATA	32 (8 8 8 8)		
	PRINT	3	YES, NO	
	LINES	2		55

Table A-18. SORT Utility GETPARMs

Prname	Keyword	Length	Options	Default
OPTIONS	FUNCTION	5	SORT, MERGE	SORT
	MEMORY	3		Depends on the memory space available
	ADDROUT	3	YES, NO	NO
	KEYOUT	3	YES, NO	NO
	STABLE	3	YES, NO	NO
	REFORMAT	3	YES, NO	NO
INPUT (Up to 20 input files if MERGE; Up to 10 input files if SORT)	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	SHARED	3	YES, NO	NO
	SELECT	3	YES, NO	NO
	MOREFILE (only if FUNCTION = SORT)	3	YES, NO	NO
	DEVICE	4	DISK, TAPE	DISK
	FILESEQ	4	1–9999	1
	RECORDS	6	1–999999	1000
LOCK (Only if SHARED = YES)	LOCK	3	YES, NO	YES
	TIMEOUT	3	0–255	10
	BYPASS	3	YES, NO	NO

(continued)

Table A-18. SORT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT (Only if SELECT = YES in INPUT)	FLDPOS1 to FLDPOS4	4		
	LENGTH1 to LENGTH 4	3		
	FLDTYP1 to FLDTYP4	1	B, C, D, L, P, Z	C
	TSTREL1 to TSTREL4	2	EQ, NE, GT, GE, LT, LE	
	VALUE1 to VALUE4	18	Either position in bytes or literal data inside quotes	
	CONECT1 to CONECT4	3	AND, OR, *b*	*b*
SELECT2 (Only if CONECT4 ≠*b*	FLDPOS5 to FLDPOS8	4		
	LENGTH5 to LENGTH8	3		
	FLDTYP5 to FLDTYP8	1	B, C, D, L, P, Z	C
	TSTREL5 to TSTREL8	2	EQ, NE, GT, GE, LT, LE	
	VALUE5 to VALUE8	18	Either position in bytes or literal data inside quotes	
	CONECT5 to CONECT8	3	AND, OR, *b*	*b*
SELECT3 (Only if CONECT8 ≠*b*	FLDPOS9 to FLDPOS12	4		
	LENGTH9 to LENGTH12	3		
	FLDTYP9 to FLDTYP12	1	B, C, D, L, P, Z	C
	TSTREL9 to TSTREL12	2	EQ, NE, GT, GE, LT, LE	
	VALUE9 to VALUE12	18	Either position in bytes or literal data inside quotes	
	CONECT9 to CONECT12	3	AND, OR, *b*	*b*

(continued)

Table A-18. SORT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT4 (Only if CONECT12 ≠b)	FLDPOS13 to FLDPOS16	4		
	LENGTH13 to LENGTH16	3		
	FLDTYP13 to FLDTYP16	1	B, C, D, L, P, Z	C
	TSTREL13 to TSTREL16	2	EQ, NE, GT, GE, LT, LE	
	VALUE13 to VALUE16	18	Either position in bytes or literal data inside quotes	
	CONECT13 to CONECT16	3	AND, OR, b	b
SELECT5 (Only if CONECT16 ≠b)	FLDPOS17 to FLDPOS20	4		
	LENGTH17 to LENGTH20	3		
	FLDTYP17 to FLDTYP20	1	B, C, D, L, P, Z	C
	TSTREL17 to TSTREL20	2	EQ, NE, GT, GE, LT, LE	
	VALUE17 to VALUE20	18	Either position in bytes or literal data inside quotes	
	CONECT17 to CONECT20	3	AND, OR, b	b
SELECT6 (Only if CONECT20 ≠b)	FLDPOS21 to FLDPOS24	4		
	LENGTH21 to LENGTH24	3		
	FLDTYP21 to FLDTYP24	1	B, C, D, L, P, Z	C
	TSTREL21 to TSTREL24	2	EQ, NE, GT, GE, LT, LE	
	VALUE21 to VALUE24	18	Either position in bytes or literal data inside quotes	
	CONECT21 to CONECT24	3	AND, OR, b	b

(continued)

Table A-18. SORT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT7 (Only if CONECT24 ≠b)	FLDPOS25 to FLDPOS28	4		
	LENGTH25 to LENGTH28	3		
	FLDTYP25 to FLDTYP28	1	B, C, D, L, P, Z	C
	TSTREL25 to TSTREL28	2	EQ, NE, GT, GE, LT, LE	
	VALUE25 to VALUE28	18	Either position in bytes or literal data inside quotes	
	CONECT25 to CONECT28	3	AND, OR, b	b
SELECT8 (Only if CONECT28 ≠b)	FLDPOS29 to FLDPOS32	4		
	LENGTH29 to LENGTH32	3		
	FLDTYP29 to FLDTYP32	1	B, C, D, L, P, Z	C
	TSTREL29 to TSTREL32	2	EQ, NE, GT, GE, LT, LE	
	VALUE29 to VALUE32	18	Either position in bytes or literal data inside quotes	
	CONECT29 to CONECT31	3	AND, OR, b	b
KEYS	KEYS	1	1–81	
	POST1 to POST8	4		
	LENGTH1 to LENGTH8	3	1–256	
	TYPE1 to TYPE8	1	B, C, D, F, L, P, Z	C
	ORDER1 to ORDER8	1	A, D	A
FORMAT (Only if Reformat = YES)	LENGTH	4	1–2096	Input value
	PAD	2	2 Hex digits or 1 character	
	INPOS1 to INPOS10	4	1–9999	
	LENGTH1 to LENGTH10	4	1–9999	
	OUTPOS1 to OUTPOS10	4	1–9999	

(continued)

Table A-18. SORT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	REPLACE	3	YES, NO	NO
	COMPRESS	3	YES, NO	YES for variable-length input; NO for fixed-length input
	DEVICE	4	DISK, TAPE	DISK
	FILESEQ	4	1–9999	1
WORKFILE	WORKFILE is a default GETPARM; refer to Section A.4.			
KEYFILE	KEYFILE is a default GETPARM; refer to Section A.4.			

Table A-19. SORTINT Utility GETPARMs

Prname	Keyword	Length	Options	Default
OPTIONS	FUNCTION	5	SORT, MERGE	SORT
	MEMORY	3		128
	ADDROUT	3	YES, NO	NO
	KEYOUT	3	YES, NO	NO
	STABLE	3	YES, NO	NO
	EXTERNAL	3	YES, NO	NO
	REFORMAT	3	YES, NO	NO
	SHARED	3	YES, NO	NO
INTABLE	FILE	8		
	LIBRARY	8		TRNTABLE
	VOLUME	6		User's INVOL
INPUT (Up to 20 input files if MOREFILE = YES)	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	SELECT	3	YES, NO	NO
	MOREFILE	3	YES, NO	NO
SELECT (Only if SELECT = YES in INPUT)	FLDPOS1 to FLDPOS4	4		
	LENGTH1 to LENGTH4	3		
	FLDTYP1 to FLDTYP4	1	B, C, D, L, P, Z	C
	TSTREL1 to TSTREL4	2	EQ, NE, GT, GE, LT, LE	
	VALUE1 to VALUE4	18	Either position in bytes or literal data inside quotes	
	CONECT1 to CONECT4	3	AND, OR, b	b

(continued)

Table A-19. SORTINT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT2 (Only if CONECT4 ≠b)	FLDPOS5 to FLDPOS8	4		
	LENGTH5 to LENGTH8	3		
	FLDTYP5 to FLDTYP8	1	B, C, D, L, P, Z	C
	TSTREL5 to TSTREL8	2	EQ, NE, GT, GE, LT, LE	
	VALUE5 to VALUE8	18	Either position in bytes or literal data inside quotes	
	CONECT5 to CONECT8	3	AND, OR, b	b
SELECT3 (Only if CONECT8 ≠b)	FLDPOS9 to FLDPOS12	4		
	LENGTH9 to LENGTH12	3		
	FLDTYP9 to FLDTYP12	1	B, C, D, L, P, Z	C
	TSTREL9 to TSTREL12	2	EQ, NE, GT, GE, LT, LE	
	VALUE9 to VALUE12	18	Either position in bytes or literal data inside quotes	
	CONECT9 to CONECT12	3	AND, OR, b	b
SELECT4 (Only if CONECT12 ≠b)	FLDPOS13 to FLDPOS16	4		
	LENGTH13 to LENGTH16	3		
	FLDTYP13 to FLDTYP16	1	B, C, D, L, P, Z	C
	TSTREL13 to TSTREL16	2	EQ, NE, GT, GE, LT, LE	
	VALUE13 to VALUE16	18	Either position in bytes or literal data inside quotes	
	CONECT13 to CONECT16	3	AND, OR, b	b

(continued)

Table A-19. SORTINT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT5 (Only if CONECT16 ≠b)	FLDPOS17 to FLDPOS20	4		
	LENGTH17 to LENGTH20	3		
	FLDTYP17 to FLDTYP20	1	B, C, D, L, P, Z	C
	TSTREL17 to TSTREL20	2	EQ, NE, GT, GE, LT, LE	
	VALUE17 to VALUE20	18	Either position in bytes or literal data inside quotes	
	CONECT17 to CONECT20	3	AND, OR, b	b
SELECT6 (Only if CONECT20 ≠b)	FLDPOS21 to FLDPOS24	4		
	LENGTH21 to LENGTH24	3		
	FLDTYP21 to FLDTYP24	1	B, C, D, L, P, Z	C
	TSTREL21 to TSTREL24	2	EQ, NE, GT, GE, LT, LE	
	VALUE21 to VALUE24	18	Either position in bytes or literal data inside quotes	
	CONECT21 to CONECT24	3	AND, OR, b	b
SELECT7 (Only if CONECT24 ≠b)	FLDPOS25 to FLDPOS28	4		
	LENGTH25 to LENGTH28	3		
	FLDTYP25 to FLDTYP28	1	B, C, D, L, P, Z	C
	TSTREL25 to TSTREL28	2	EQ, NE, GT, GE, LT, LE	
	VALUE25 to VALUE28	18	Either position in bytes or literal data inside quotes	
	CONECT25 to CONECT28	3	AND, OR, b	b

(continued)

Table A-19. SORTINT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
SELECT8 (Only if CONECT28 ≠b)	FLDPOS29 to FLDPOS32	4		
	LENGTH29 to LENGTH32	3		
	FLDTYP29 to FLDTYP32	1	B, C, D, L, P, Z	C
	TSTREL29 to TSTREL32	2	EQ, NE, GT, GE, LT, LE	
	VALUE29 to VALUE32	18	Either position in bytes or literal data inside quotes	
	CONECT29 to CONECT31	3	AND, OR, b	b
KEYS	KEYS	1	1–8	1
	POST1 to POST8	4		
	LENGTH1 to LENGTH8	3	1–256	
	TYPE1 to TYPE8	1	B, C, D, F, L, P, Z	C
	ORDER1 to ORDER8	1	A, D	A
REFORMAT	LENGTH	4	1–2048	
	PAD	2	Any valid hexadecimal code	20
	INPOS1 to INPOS12	4	1–2048	
	LENGTH1 to LENGTH12	3	1–999	
	OUTPOS1 to OUTPOS12	4	1–2048	
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	REPLACE	3	YES, NO	NO
	COMPRESS	3	YES, NO	YES for variable-length input; NO for fixed-length input
WORKFILE	WORKFILE is a default GETPARM; refer to Section A.4.			
KEYFILE	KEYFILE is a default GETPARM; refer to Section A.4.			

Table A-20. TABLEDIT Utility GETPARMs

Prname	Keyword	Length	Options	Default
TYPE	FILE	8		Blank
	LIBRARY	8		Blank
	VOLUME	6		Blank
	(PF Keys)		1 = One-to-one table 2 = One-to-two table 3 = Two-to-one table 4 = Case flip table 16 = Exit	
PREFILL1	(PF Keys)		1 = Standard ASCII 2 = Upper- and lowercase sorted equivalently 3 = Uppercase before corresponding lowercase 16 = Return	
EDIT	CODE1 to CODE32	2	Hexadecimal values (00–FF)	Dependent on the PREFILL1 selected.
	INCREMENT	2	Hexadecimal values (00–FF)	Retains last entered value
	(PF Keys)		b = Summary 2 = First 3 = Last 4 = Previous 5 = Next	
			6 = Increment	See Keyword INCREMENT
			7 = Define 8 = Delete 9 = Save	
			14 = Show One-to-Two	Dependent on the PREFILL1 selected.
			16 = Cancel	
DEFINE	DEFINEC	1 (2 if not a 1-1)	ASCII character value	Blank, retains previous value.
	DEFINEH	2 (4 if not a 1-1)	Hexadecimal values (00–FF)	Blank, retains previous value.
	DEFCODE	2 (4 if not a 1-1)	Hexadecimal values (00–FF)	Blank, retains previous value.
	(PF Keys)		b = Create Definition 16 = Return	

(continued)

Table A-20. TABLEDIT Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
DELETE	DEFINEC	1 (2 if not a 1-1)	ASCII character value	Blank, retains previous value.
	DEFINEH	2 (4 if not a 1-1)	Hexadecimal values (00–FF)	Blank, retains previous value.
	(PF Keys)		b = Delete Definition 16 = Return	
EQUIV	(PF Keys)		b = Accept Change 2 = First 3 = Last 4 = Previous 5 = Next 16 = Cancel Change	
SUMMARY	(PF Keys)		b = Continue	
WARNING	(PF Keys)		b = Continue 16 = Throw away modifications; edit another table	
SAVE	FILE LIBRARY VOLUME (PF Keys)	8 8 6	 b = Save 3 = Replace Input File 16 = Return to Edit	Blank Blank Blank

Table A-21. TAPECOPY Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	DEVICE	8	DISK, TAPE	TAPE
	CONVERT	2	N, E, A, BA, AB, HA, AH	N
	MULTYPE	1	Y, N	N
	FSEQ	4	1 – 9999	1
	LABEL	2	AL, IL, NL	AL
	HEADER2	3	YES, NO	YES
MOUNT	DEVICE	3		
	USAGE	2	SH, EX	SH
	(PF Keys)		b = Continue 1 = Respecify input	

(continued)

Table A-21. TAPECOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
TAPEFILE	RECFORM	1	F, V, I, U	Depends on label type
	RECSIZE	8	4 – 2048	Depends on input file
	BLOCKED	1	Y, N	Y
	BLKSIZE	8	18–3760	RECSIZE times 100
	COMPRESS	1	Y, N	N
TAPE7	DENSITY	3	556, 800	800
	PARITY	4	EVEN, ODD	EVEN
	IBM1401	3	YES, NO (Appears only if DEVICE = TAPE)	NO
OUTPUT	FILE	8		
	LIBRARY	8		User's OUTLIB
	VOLUME	6		User's OUTVOL
	DEVICE	8	TAPE, DISK	TAPE if input DEVICE = DISK; DISK if input DEVICE = TAPE
	FSEQ	4	1 – 9999	1
	LABEL	2	AL, IL, NL	AL
DISKFILE	NRECS	7	Specified only if input file is on tape	
	FILEORG	1	C, I, X, P	C
	RECFORM	1	F, V	Input file's format
	COMPRESS	1	Y, N	
	FILECLAS	1		
	RETAIN	3	YES, NO	
	KEYLEN	3	1 – 255	
	KEYPOS	5	1 – 2048	
	IPACK	3	1 – 100	100
	DPACK	3	1 – 100	100
TYPES (for multiple record type conversions only)	CHAR1 to CHAR3	2		
	COLUMN1 to COLUMN3	4		
	SWITCH1 to SWITCH3	3	YES, NO	
	CHARSET	6	INPUT, OUTPUT	INPUT
OPTIONS (for multiple record type conversions only	POST01 to POST08	2		
	LENGTH	3		
	TYPE01 to TYPE08	1	B, C, P, Z	
	SWITCH01 to SWITCH08	1	A – Z	

(continued)

Table A-21. TAPECOPY Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
PAD (only for variable- to fixed-length conversions)	PAD	1	b, 0	b
EOJ	(PF Keys)		1 = Rerun TAPECOPY 16 = End processing	

Table A-22. TAPEINIT Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	VOLUME	6		
MOUNT	DEVICE	3		
TAPE (for 9-track tape)	LABEL	2	AL, IL, NL	AL
	OWNER	14		
	DENSITY	4	800, 1600, 6250	1600
TAPE (for 7-track tape)	PARITY	4	EVEN, ODD	EVEN
	DENSITY	3	556, 800	800
EOJ	(PF Keys)		b = End processing 1 = Rerun TAPEINIT 16 = End processing	

Table A-23. TRANSL Utility GETPARMs

Prname	Keyword	Length	Options	Default
INPUT	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	TYPES	3	YES, NO	NO
	CODE	6	EBCDIC, ASCII, TABLE, NONE	EBCDIC
INTABLE	FILE	8		b
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
TABLE	HEX00#OF HEXFO#FF	35	Hex chars with embedded spaces at positions 9, 20, and 29	Defaults taken from the Default Table or INTABLE.

(continued)

Table A-23. TRANSL Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
EXPLAIN	(PF Keys)		b = Continue	
OUTTABLE	FILE LIBRARY VOLUME	8 8 6		Defaults taken from INTABLE
TYPES	CHAR1 to CHAR3	2	Any character	b
	COLUMN1 to COLUMN3	4	Any numeric	b
	SWITCH1 to SWITCH3	3	YES, NO	YES
	CHARSET	6	INPUT, OUTPUT	INPUT
OPTIONS	POSTO1 to POST24	4		
	LENGTH01 to LENGTH24	4		
	TYPE01 to TYPE24	1	B, C, P, Z	
	SWITCH01 to SWITCH24	1	b, I, D	b
INDEXED	KEYLEN KEYPOS	3 3		Same as input file
OUTPUT	FILE LIBRARY VOLUME RECORDS	8 8 6 7		 User's OUTLIB User's OUTVOL Size of input file
	RETAIN RELEASE FILECLAS DEVICE	3 3 1 11	b YES, NO A – Z, #, b, @, $ DISK	 NO b
EOJ	(PF Keys)		1 = Restart TRANSL 16 = End processing	
WARNING	(PF Keys)		b = Accept 16 = Restart field definition	
ERROR01	(PF Keys)		1 = Restart field definition 16 = End processing	
ERROR02	(PF Keys)		1 = Start record type again 16 = End processing	
ERROR03	(PF Keys)		1 = Restart field definition 16 = End processing	

(continued)

Table A-23. TRANSL Utility GETPARMs (continued)

Prname	Keyword	Length	Options	Default
ERROR04	(PF Keys)		1 = Start record type again 16 = End processing	
ERROR05	(PF Keys)		b = Acknowledge	
ERROR06	(PF Keys)		b = Acknowledge	

Table A-24. VERIFY Utility GETPARMs

Prname	Keyword	Length	Options	Default
OPTIONS	RANGE	7	FILE, LIBRARY, VOLUME	FILE
	FILE	8		
	LIBRARY	8		User's INLIB
	VOLUME	6		User's INVOL
	VERIFY	7	ALL, PRIMARY	ALL
	DISPLAY	3	YES, NO	YES
MOUNT	DEVICE (PF Keys)	3	b = Start mount 1 = Respecify mount	
NOSPACE	(PF Keys)		b = Continue without printed report 16 = End processing	
ENDOFJOB	(PF Keys)		b = End processing 1 = Rerun VERIFY 16 = End processing	
ERROR1	(PF Keys)		b = Continue	
ERROR2	(PF Keys)		b = Continue 16 = End file verification	
ERROR3	(PF Keys)		b = Continue 1 = Display error details 16 = End file verification	
SUMMARY	(PF Keys)		b = Continue	
PRINT (Prname for Summary Report)				
ERPRT (Prname for Error Report)				

A.4 DEFAULT GETPARMS

Default GETPARMS are screens that do not appear at runtime because they have been previously supplied through the SET Usage Constants function (PF2) of the Command Processor menu. To change default values, include the appropriate default prname, keywords, and values from the following list in a procedure.

Table A-25. Default GETPARMs

Prname	Keyword	Length	Options	Default
PRINT	FILE	8		
	LIBRARY	8		# plus User ID plus PRT
	VOLUME	6		User's SPOOLVOL
	RECORDS	7		
	RETAIN	3		
	RELEASE	3	YES, NO	YES
	FILECLAS	1	A–Z, #, b	#
	DEVICE	11	DISK, PRINTER	DISK
	FORM#	3	000–255	User's FORM#
	PRTCLASS	1	A – Z	User's PRTCLASS
	COPIES	5	1 – 32767	1
WORK	FILE	8		
	LIBRARY	8		# plus User ID plus WORK
	VOLUME	6		User's WORKVOL
	RETAIN	3		
	RELEASE	3	YES, NO	YES
	FILECLAS	1	A–Z, #, b	#
	RECORDS	7		
	DEVICE	11	DISK, NONE	DISK

APPENDIX B
SYSTEM UTILITY RETURN CODES

System Utility Return codes indicate the reasons for a program failure or warn of conditions that could cause program failure. VS programs and some Procedure statements generate return codes during execution. A successful program execution generates a return code of 0, and is not displayed.

Since most of the system utilities have been designed specifically for interactive use, many of the utilities display any error messages that require attention at the workstation during the processing of one of the utilities. All of the utilities have a return code of 0 if the program completes successfully. If an error occurs during the processing of one of the utilities, return codes may or may not be returned, depending on the utility. Table B-1 is a quick reference to the utilities that have return codes returned (marked by Yes) and the utilities that do not (marked by No).

Table B-1. Utility Return Code Quick Reference

Utility	Return	Utility	Return	Utility	Return
CIP	No	FLOPYDUP	No	POOLSTAT	No
COPY	Yes	FONTCNTL	Yes	SHRSTAT	No
COPYOIS	No	FORMCNTL	No	SORT	Yes
COPY2200	No	IBMCOPY	No	SORTINT	Yes
COPYWP	No	IOELOG	No	TABLEDIT	No
DISKINIT	Yes	IOTRACE	No	TAPECOPY	No
DISPLAY	No	LISTVTOC	Yes	TAPEINIT	No
EZFORMAT	Yes	OISCART	No	TRANSL	No
FASTLINK	No	PATCH	No	VERIFY	Yes

The system utilities that display a return code for error conditions are listed, along with their corresponding return codes, in the following list:

Program Name	Return Code	Meaning
COPY	100	No copy took place
	104	Some program privileges lost
	108	Disk error; run LISTVTOC
	112	No space in the output volume
	116	I/O error on output
	120	Boundary violation
	124	Key out of sequence

Program Name	Return Code	Meaning
COPY (cont.)	128	Duplicate key
	132	The primary extent was exceeded
	136	DMS error
	140	Duplicate file name encountered
	144	Copy completed in IO mode after primary extent was exceeded
	148	O/P VTOC full
	152	Input RAM error
DISKINIT	4	Termination by user
	8	Insufficient space in I/O buffer
	12	Mount operation unsuccessful
	16	Bad disk sector encountered
	20	Bad MOUNT SVC return code
	24	Bad FREEBUF return code
	28	Bad XIO return code
	32	Disk I/O error
EZFORMAT	1-4	Warning; program will probably run
	5-7	Severe Error; program will not run correctly
	8-16	Fatal error; object code not generated
FONTCNTL	1	Operating System version is not compatible
LISTVTOC	4	Extent lost on disk
	8	Error on file label
	12	Error in library directory
	16	Error in VTOC
SORT and SORTINT	4	No records to be sorted; input records did not meet selection criteria, or you specified an empty input file
	8	Insufficient space in stack or I/O buffer
	12	Record size is more than 2024 bytes long
	16	Invalid sort key
	20	Unexpected program check
	24	Input records out of order in file to be merged; program cannot proceed
	28	Input record count problem
VERIFY	2	The library cannot be verified
	3	The library was deleted when VERIFY was in progress
	4	Volume specified is empty and cannot be verified
	5	The volume specified is NL and cannot be verified
	6	Range should be "VOLUME", "LIBRARY", or "FILE"
	7	Due to buffer allocation problems, alternate indexes cannot be verified
	8	VERIFY option should be "PRIMARY" or "ALL"
	9	DISPLAY option should be "YES" or "NO"
	10	Volume mounted is NL instead of SL
	11	I/O error
	12	Device is detached
	13	Device is in use
	14	The file was deleted when VERIFY was in progress (skipped)
	15	Parameter is blank
	16	File in use (skipped)

Program Name	Return Code	Meaning
VERIFY (cont.)	17	User not authorized to access file
	18	File not found
	19	File in use
	20	User not authorized to access the file
	21	File is empty
	22	File not an indexed file
	25	No space for WORK FILE on volume; alternate indexes cannot be verified
	32	Not enough space on any eligible work volume for the report file
	37	Device is not a disk
	38	Volume is already mounted at another device
	39	Volume has been mounted instead of the one specified
	45	Volume specified not mounted
	46	Volume specified is mounted for exclusive use by another user
	48	Library not found
	100	VERIFY processing completed
All compilers[a]	1-4	Warning
	5-8	Severe error (program will not execute correctly)
	9-16	Fatal error (program will not execute)

[a] Compilers that are used by some utilities (EZFORMAT has an option that uses the COBOL compiler, for example) and generate return codes; the utilities themselves may not necessarily generate return codes of their own.

APPENDIX C
WISCII CHARACTER SET

C.1 INTRODUCTION

There are eight Wang International Standard Code for Information Interchange (WISCII) character sets that are currently supported for font-loadable printers. Whether you have one or more of the character sets in files for your printers depends on whether they were ordered with your font-loadable printer. This appendix contains a matrix of the WISCII-1 character set (United States and Western Europe). For information about the other WISCII character sets, contact your Wang representative. A list of the supported WISCII character sets is as follows:

WISCII-1	United States and Western Europe character set
WISCII-2	Scientific character set
WISCII-3	Telex character set
WISCII-4	Arabic character set
WISCII-5	Hebrew character set
WISCII-6	United States and Eastern Europe character set
WISCII-7	United States, Bulgaria, and the Union of Soviet Socialists Republic character set
WISCII-8	United States and Greece character set

C.1.1 Reading the Character Set Table

The WISCII-1 character set table is a matrix of characters that are cross-referenced by their hexadecimal code. To determine the hexadecimal value for a specific character, read the column heading of the specified character first and then the row heading. For example, a capital A is found in Column 4 and Row 1, hence the hexadecimal value for A is 41. Hexadecimal values range from 20 to FF.

Table C-1. WISCII-1 Character Set

Hex Code	0_	1_	2_	3_	4_	5_	6_	7_	8_	9_	A_	B_	C_	D_	E_	F_
_0				0	@	P	`	p	°	Â	â	Ğ	ğ	Þ	þ	£
_1			!	1	A	Q	a	q	◆	À	à	Ij	ij	Đ	ð	ƒ
_2			"	2	B	R	b	r	▶	Á	á	İ	ı	Ý	ý	¥
_3			#	3	C	S	c	s	◀	Ä	ä	Î	î	Ş	ş	¼
_4			$	4	D	T	d	t	→	Ã	ã	Ì	ì	'	'	½
_5			%	5	E	U	e	u	⌙	←	→	Í	í	Û	û	¾
_6			&	6	F	V	f	v	|	Å	å	Ï	ï	Ù	ù	ˆ
_7			'	7	G	W	g	w	■	↓	†	L·L	l·l	Ú	ú	˜
_8			(	8	H	X	h	x	‼	Æ	æ	Ñ	ñ	Ü	ü	˙
_9			)	9	I	Y	i	y	↕	Ç	ç	Ô	ô	©	™	¨
_A			*	:	J	Z	j	z	↓	‡	□	Ò	ò	®	¤	˚
_B			+	;	K	[	k	{	↑	•		Ó	ó	Ĳ	↔	¸
_C			,	<	L	\	l	|	←	Ê	ê	Ö	ö	ª	º	˝
_D			-	=	M	]	m	}	±	È	è	Õ	õ	«	»	˛
_E			.	>	N	↑	n	~	¡	É	é	Œ	œ	§	ß	
_F			/	?	O	_	o	¢	¿	Ë	ë	Ø	ø	¶	·	

Display Character

NOTE

- *Hexadecimal 20 is a space.*
- *Hexadecimal AB is a required space.*
- *Hexadecimal FE is a Dead Key 1.*
- *Hexadecimal FF is a Dead Key 2.*

C.2 WISCII-1 CHART

This section presents the WISCII-1 character set in chart form. The decimal equivalent of each character has been added to the chart for easy binary and hexadecimal conversion. To read the chart, find the binary, decimal, graphic or hexadecimal value that you want, then read across for the corresponding values. For example, capital A has the following values:

Base	Value
Binary	01000001
Decimal	65
Graphic	A
Hexadecimal	41

The ranges for the chart values are as follows:

Base	Range
Binary	00100000 to 11111111
Decimal	32 to 255
Hexadecimal	20 to FF

The WISCII-1 character set is shown in Table C-2.

Table C-2. WISCII-1 Character Set Chart

Binary	Hex	Graphic/Description	Decimal
00100000	20	space	32
00100001	21	! exclamation	33
00100010	22	" double quote	34
00100011	23	# pound	35
00100100	24	$ dollar	36
00100101	25	% percent	37
00100110	26	& ampersand	38
00100111	27	' single quote	39
00101000	28	(left parenthesis	40
00101001	29	) right parenthesis	41
00101010	2A	* asterisk	42
00101011	2B	+ plus	43
00101100	2C	, comma	44
00101101	2D	− minus, hyphen	45
00101110	2E	. period	46
00101111	2F	/ slash	47
00110000	30	0 zero	48
00110001	31	1 one	49
00110010	32	2 two	50
00110011	33	3 three	51
00110100	34	4 four	52
00110101	35	5 five	53
00110110	36	6 six	54
00110111	37	7 seven	55
00111000	38	8 eight	56
00111001	39	9 nine	57
00111010	3A	: colon	58
00111011	3B	; semicolon	59
00111100	3C	< less than	60
00111101	3D	= equal	61
00111110	3E	> greater than	62
00111111	3F	? question	63

(continued)

Table C-2. WISCII-1 Character Set Chart (continued)

Binary	Hex	Graphic/Description		Decimal
01000000	40	@	at	64
01000001	41	A	capital A	65
01000010	42	B	capital B	66
01000011	43	C	capital C	67
01000100	44	D	capital D	68
01000101	45	E	capital E	69
01000110	46	F	capitaL F	70
01000111	47	G	capital G	71
01001000	48	H	capital H	72
01001001	49	I	capital I	73
01001010	4A	J	capital J	74
01001011	4B	K	capital K	75
01001100	4C	L	capital L	76
01001101	4D	M	capital M	77
01001110	4E	N	capital N	78
01001111	4F	O	capital O	79
01010000	50	P	capital P	80
01010001	51	Q	capital Q	81
01010010	52	R	capital R	82
01010011	53	S	capital S	83
01010100	54	T	capital T	84
01010101	55	U	capital U	85
01010110	56	V	capital V	86
01010111	57	W	capital W	87
01011000	58	X	capital X	88
01011001	59	Y	capital Y	89
01011010	5A	Z	capital Z	90
01011011	5B	[	left bracket	91
01011100	5C	\	back slash	92
01011101	5D	]	right bracket	93
01011110	5E	↑	up arrow	94
01011111	5F	_	underscore	95
01100000	60	‘	open quote	96
01100001	61	a	lowercase a	97
01100010	62	b	lowercase b	98
01100011	63	c	lowercase c	99
01100100	64	d	lowercase d	100
01100101	65	e	lowercase e	101
01100110	66	f	lowercase f	102
01100111	67	g	lowercase g	103
01101000	68	h	lowercase h	104
01101001	69	i	lowercase i	105
01101010	6A	j	lowercase j	106
01101011	6B	k	lowercase k	107
01101100	6C	l	lowercase l	108
01101101	6D	m	lowercase m	109
01101110	6E	n	lowercase n	110
01101111	6F	o	lowercase o	111

(continued)

Table C-2. WISCII-1 Character Set Chart (continued)

Binary	Hex	Graphic/Description	Decimal
01110000	70	p lowercase p	112
01110001	71	q lowercase q	113
01110010	72	r lowercase r	114
01110011	73	s lowercase s	115
01110100	74	t lowercase t	116
01110101	75	u lowercase u	117
01110110	76	v lowercase v	118
01110111	77	w lowercase w	119
01111000	78	x lowercase x	120
01111001	79	y lowercase y	121
01111010	7A	z lowercase z	122
01111011	7B	{ left brace	123
01111100	7C	| verticle bar	124
01111101	7D	} right brace	125
01111110	7E	~ tilde, appx.	126
01111111	7F	¢ cent	127
10000000	80	° degree	128
10000001	81	◆ center	129
10000010	82	▶ tab	130
10000011	83	◀ return	131
10000100	84	→ indent	132
10000101	85	⌙ decimal tabulation	133
10000110	86	| format	134
10000111	87	■ stop	135
10001000	88	!! note	136
10001001	89	↕ merge	137
10001010	8A	↓ down arrow	138
10001011	8B	↑ up arrow	139
10001100	8C	← left arrow	140
10001101	8D	± plus or minus	141
10001110	8E	¡ inverted exclam.	142
10001111	8F	¿ inverted question	143
10010000	90	Â A circumflex	144
10010001	91	À A grave	145
10010010	92	Á A acute	146
10010011	93	Ä A umlaut	147
10010100	94	Ã A tilde	148
10010101	95	← left arrow	149
10010110	96	Å A angstrom	150
10010111	97	↓ down arrow	151
10011000	98	Æ AE ligature	152
10011001	99	Ç C cedilla	153
10011010	9A	‡ double dagger	154
10011011	9B	• bullet	155
10011100	9C	Ê E circumflex	156
10011101	9D	È E grave	157
10011110	9E	É E acute	158
10011111	9F	Ë E umlaut	159

(continued)

Table C-2. WISCII-1 Character Set Chart (continued)

Binary	Hex	Graphic/Description	Decimal
10100000	A0	â a circumflex	160
10100001	A1	à a grave	161
10100010	A2	á a acute	162
10100011	A3	ä a umlaut	163
10100100	A4	ã a tilde	164
10100101	A5	→ right arrow	165
10100110	A6	å a angstrom	166
10100111	A7	† dagger	167
10101000	A8	æ ae ligature	168
10101001	A9	ç c cedilla	169
10101010	AA	□ ballot box	170
10101011	AB	required space	171
10101100	AC	ê e circumflex	172
10101101	AD	è e grave	173
10101110	AE	é e acute	174
10101111	AF	ë e umlaut	175
10110000	B0	Ğ G breve	176
10110001	B1	IJ IJ ligature	177
10110010	B2	I I dotted	178
10110011	B3	Î I circumflex	179
10110100	B4	Ì I grave	180
10110101	B5	Í I acute	181
10110110	B6	Ï I umlaut	182
10110111	B7	L·L LL catalonian lig.	183
10111000	B8	Ñ N tilde	184
10111001	B9	Ô O circumflex	185
10111010	BA	Ò O grave	186
10111011	BB	Ó O acute	187
10111100	BC	Ö O umlaut	188
10111101	BD	Õ O tilde	189
10111110	BE	Œ OE ligature	190
10111111	BF	Ø O slashed	191
11000000	C0	ğ g breve	192
11000001	C1	ij ij ligature	193
11000010	C2	ı i dotless	194
11000011	C3	î i circumflex	195
11000100	C4	ì i grave	196
11000101	C5	í i acute	197
11000110	C6	ï i umlaut	198
11000111	C7	l·l ll catalonian lig.	199
11001000	C8	ñ n tilde	200
11001001	C9	ô o circumflex	201
11001010	CA	ò o grave	202
11001011	CB	ó o acute	203
11001100	CC	ö o umlaut	204
11001101	CD	õ o tilde	205
11001110	CE	œ oe ligature	206
11001111	CF	ø o slashed	207

(continued)

Table C-2. WISCII-1 Character Set Chart (continued)

Binary	Hex	Graphic/Description	Decimal
11010000	D0	Þ Icelandic thorn	208
11010001	D1	Ð Icelandic eth	209
11010010	D2	Ý Y acute	210
11010011	D3	Ş S cedilla	211
11010100	D4	‘ open quote	212
11010101	D5	Û U circumflex	213
11010110	D6	Ù U grave	214
11010111	D7	Ú U acute	215
11011000	D8	Ü U umlaut	216
11011001	D9	© copyright	217
11011010	DA	™ reg. trademark	218
11011011	DB	℞ prescription	219
11011100	DC	ª a superior	220
11011101	DD	« open quote, Europe	221
11011110	DE	§ section	222
11011111	DF	¶ paragraph	223
11100000	E0	þ lower Ice. thorn	224
11100001	E1	ð lower Ice. eth	225
11100010	E2	ý y acute	226
11100011	E3	ş s cedilla	227
11100100	E4	’ close quote	228
11100101	E5	û u circumflex	229
11100110	E6	ù u grave	230
11100111	E7	ú u acute	231
11101000	E8	ü u umlaut	232
11101001	E9	™ trademark	233
11101010	EA	¤ Int. monetary	234
11101011	EB	↔ double arrows	235
11101100	EC	º o superior	236
11101101	ED	» close quote, Euro.	237
11101110	EE	ß double S, German	238
11101111	EF	· centered dot	239
11110000	F0	£ pound sterling	240
11110001	F1	ƒ florin	241
11110010	F2	¥ yen	242
11110011	F3	¼ one quarter	243
11110100	F4	½ one half	244
11110101	F5	¾ three quarters	245
11110110	F6	ˆ accent circumflex	246
11110111	F7	` accent grave	247
11111000	F8	´ accent acute	248
11111001	F9	¨ accent umlaut	249
11111010	FA	˜ accent tilde	250
11111011	FB	¸ accent cedilla	251
11111100	FC	ˇ accent hacek	252
11111101	FD	˘ accent breve	253
11111110	FE	dead key 1	254
11111111	FF	dead key 2	255

INDEX BY UTILITY

CIP (Compress-in-Place) Utility

COPY Utility

COPYOIS Utility

COPY2200 Utility

COPY2200 Utility (continued)

COPYWP Utility

COPYWP Utility (continued)

DISKINIT Utility

DISKINIT Utility (continued)

DISPLAY Utility

DISPLAY Utility (continued)

EZFORMAT Utility

FASTLINK Utility

FASTLINK Utility (continued)

FLOPYDUP Utility

FONTCNTL Utility

FORMCNTL Utility

IBMCOPY Utility

IBMCOPY Utility (continued)

IOELOG Utility

IOTRACE Utility

IOTRACE Utility (continued)

LISTVTOC Utility

OISCART Utility

PATCH Utility

POOLSTAT Utility

SHRSTAT Utility

SORT Utility

SORTINT Utility

SORTINT Utility (continued)

TABLEDIT Utility

TAPECOPY Utility

TAPECOPY Utility (continued)

TAPEINIT Utility

TRANSL Utility

TRANSL Utility (continued)

VERIFY Utility

INDEX

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z

WANG

Customer Comment Form

Publication Number 715-0421A

Title VS SYSTEM UTILITIES REFERENCE

Help Us Help You . . .

We've worked hard to make this document useful, readable, and technically accurate. Did we succeed? Only you can tell us! Your comments and suggestions will help us improve our technical communications. Please take a few minutes to let us know how you feel.

How did you receive this publication?

- ☐ Support or Sales Rep
- ☐ Wang Supplies Division
- ☐ From another user
- ☐ Enclosed with equipment
- ☐ Don't know
- ☐ Other _____

How did you use this Publication?

- ☐ Introduction to the subject
- ☐ Classroom text (student)
- ☐ Classroom text (teacher)
- ☐ Self-study text
- ☐ Aid to advanced knowledge
- ☐ Guide to operating instructions
- ☐ As a reference manual
- ☐ Other _____

Please rate the quality of this publication in each of the following areas.

	EXCELLENT	GOOD	FAIR	POOR	VERY POOR
Technical Accuracy — Does the system work the way the manual says it does?	☐	☐	☐	☐	☐
Readability — Is the manual easy to read and understand?	☐	☐	☐	☐	☐
Clarity — Are the instructions easy to follow?	☐	☐	☐	☐	☐
Examples — Were they helpful, realistic? Were there enough of them?	☐	☐	☐	☐	☐
Organization — Was it logical? Was it easy to find what you needed to know?	☐	☐	☐	☐	☐
Illustrations — Were they clear and useful?	☐	☐	☐	☐	☐
Physical Attractiveness — What did you think of the printing, binding, etc?	☐	☐	☐	☐	☐

Were there any terms or concepts that were not defined properly? ☐ Y ☐ N If so, what were they? _____

After reading this document do you feel that you will be able to operate the equipment/software? ☐ Yes ☐ No ☐ Yes, with practice

What errors or faults did you find in the manual? (Please include page numbers) _____

Do you have any other comments or suggestions? _____

Name _____ **Street** _____

Title _____ **City** _____

Dept/Mail Stop _____ **State/Country** _____

Company _____ **Zip Code** _____ **Telephone** _____

Thank you for your help.

Printed in U.S.A. 14-3140 7-83

WANG

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 16 LOWELL, MA

POSTAGE WILL BE PAID BY ADDRESSEE

WANG LABORATORIES, INC.
PUBLICATIONS DEVELOPMENT
ONE INDUSTRIAL AVENUE
LOWELL,MASSACHUSETTS 01851

Fold

Cut along dotted line.

WANG

To Order by Phone, Call:
(800)225-0234

Order Form for Wang Manuals and Documentation

① Customer Number (If Known)

② Bill To: | Ship To:

③ Customer Contact: ( ) ( ) — Phone — Name

④ Date | Purchase Order Number

⑤ Taxable Yes ☐ No ☐ ⑥ Tax Exempt Number ⑦ Credit This Order to A Wang Salesperson Please Complete — Salesperson's Name | Employee No. | RDB No.

⑧ Document Number	Description	Quantity	⑨ Unit Price	Total Price

⑩ _____ Authorized Signature _____ Date

Sub Total	
Less Any Applicable Discount	
Sub Total	
Local State Tax	
Total Amount	

☐ Check this box if you would like a free copy of **The Literature Catalog** (700-7647)

Ordering Instructions

1. If you have purchased supplies from Wang before, and know your Customer Number, please write it here.
2. Provide appropriate Billing Address and Shipping Address.
3. Please provide a phone number and name, should it be necessary for WANG to contact you about your order.
4. Your purchase order number and date.
5. Show whether order is taxable or not.
6. If tax exempt, please provide your exemption number.
7. If you wish credit for this order to be given to a WANG salesperson, please complete.
8. Show part numbers, description and quantity for each product ordered.
9. *Pricing extensions and totaling can be completed at your option; Wang will refigure these prices and add freight on your invoice.*
10. Signature of authorized buyer and date.

Wang Supplies Division Terms and Conditions

1. **TAXES** — Prices are exclusive of all sales, use, and like taxes.
2. **DELIVERY** — Delivery will be F.O.B. Wang's plant. Customer will be billed for freight charges; and unless customer specifies otherwise, all shipments will go best way surface as determined by Wang. Wang shall not assume any liability in connection with the shipment nor shall the carrier be construed to be an agent of Wang. If the customer requests that Wang arrange for insurance the customer will be billed for the insurance charges.
3. **PAYMENT** — Terms are net 30 days from date of invoice. Unless otherwise stated by customer, partial shipments will generate partial invoices.
4. **PRICES** — The prices shown are subject to change without notice. Individual document prices may be found in the Corporate Publications Literature Catalog (700-5294)
5. **LIMITATION OF LIABILITY** — In no event shall Wang be liable for loss of data or for special, incidental or consequential damages in connection with or arising out of the use of or information contained in any manuals or documentation furnished hereunder.

Printed in U.S.A. 14-3141 7-83

WANG

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 16 LOWELL, MA

POSTAGE WILL BE PAID BY ADDRESSEE

WangDirect
Attention: Order Entry
800 Chelmsford Street
Lowell, MA 01851

Fold

WANG

ONE INDUSTRIAL AVENUE
LOWELL, MASSACHUSETTS 01851
TEL. (617) 459-5000
TWX 710-343-6769, TELEX 94-7421

Printed in U.S.A.
715-0421A
6-86