

**PUBLICATIONS
UPDATE**

Operating System/3 (OS/3)

**Interactive Services
Commands and Facilities**

**User Guide/Programmer
Reference**

UP-8845 Rev. 3-B

This Library Memo announces the release and availability of Updating Package B to "SPERRY Operating System/3 (OS/3) Interactive Services Commands and Facilities User Guide/Programmer Reference", UP-8845 Rev. 3.

This update for the 8.1 release documents two new parameters for the SCREEN command. These parameters (PAGE and CONTINUOUS) support either continuous or page mode printing for the 0791 auxiliary printer.

All other changes are corrections or expanded descriptions applicable to features in interactive services prior to the 8.1 release.

Copies of Updating Package B are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry representative. To receive only the updating package, order UP-8845 Rev. 3-B. To receive the complete manual, order UP-8845 Rev. 3.

| LIBRARY MEMO ONLY | LIBRARY MEMO AND ATTACHMENTS | THIS SHEET IS |
|---------------------------------|--|---|
| Mailing Lists BZ, CZ, and MZ | Mailing Lists A00, A01, B00, B01, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package B to UP-8845 Rev. 3, 23 pages plus Memo) | Library Memo for UP-8845 Rev. 3-B RELEASE DATE: June, 1983 |

| |
|---|
| PUBLIC UPDATE |
| Operating System/3 (OS/3) |
| Interactive Services Commands and Facilities |
| User Guide/Programmer Ref. |

This Library Memo announces the release and availability of Updating Package A to "SPERRY UNIVAC Operating System/3 (OS/3) Interactive Services Commands and Facilities User Guide/Programmer Reference", UP-8845 Rev. 3.

This update documents the following new interactive services features for the 8.0 release:

- ability to downline load a program to a UTS 40 workstation
- new formats for the DLOAD command
- new parameters on the SCREEN command for specifying where and when you system prints your print files
- clarification of the use of the RP command

All other changes are corrections or expanded descriptions applicable to features already present in the 8.0 release.

Copies of Updating Package A are now available for requisitioning. Either the updating package only, or the complete manual with the updating package may be requisitioned by your local Sperry Univac representative. To receive only the updating package, order UP-8845 Rev. 3-A. To receive the complete manual, order UP-8845 Rev. 3.

| LIBRARY MEMO ONLY | LIBRARY MEMO AND ATTACHMENTS | THIS SHEET IS |
|--------------------------------|--|--------------------------------------|
| Mailing Lists BZ, CZ and MZ | Mailing Lists A00, A01, B00, B01, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package A to UP-8845 Rev. 3, 48 pages plus Memo) | Library Memo for UP-8845 Rev. 3-A |
| | | RELEASE DATE: February, 1983 |



**PUBLICATIONS
REVISION**

Operating System/3 (OS/3)

Interactive Services
Commands and Facilities

Programmer Reference

UP-8845 Rev. 3

This Library Memo announces the release and availability of "SPERRY UNIVAC[®] Operating System/3 (OS/3) Interactive Services Commands and Facilities User Guide/Programmer Reference", UP-8845 Rev. 3.

This revision documents the following new interactive services features:

- The DEFKEY and DEFKEY DISPLAY commands
- The RECALL workstation log command
- The PR/PU and RP spooling commands and spooling function codes
- The EXECUTE job facility
- Interactive Accounting
- The MENU calling command
- A downline program load (DLOAD) command and an upline dump (ULD) command for UTS-400 users
- Job control enhancements to support distributed data processing (DDP)
- Modifications to the SCREEN and SC/SI commands and TYPE and HOLD file parameters
- Enhancements to the VTOC command display
- Enhancement to the HELP command to get help with keyword parameters
Enhancement to the SCREEN command to specify that workstation logs include commands entered in both system and workstation mode.
- Addition/modification of the following interactive facilities:
 - Error File Processor
 - COBOL Editor

| LIBRARY MEMO ONLY | LIBRARY MEMO AND ATTACHMENTS | THIS SHEET IS |
|--------------------------------|--|--|
| Mailing Lists BZ, CZ and MZ | Mailing Lists A00,A01,B00,B01,18,18U, 19,19U,20,20U,21,21U,28U,29U,75,75U, 76 and 76U (Cover and 252 pages) | Library Memo for UP-8845 Rev. 3 RELEASE DATE: October, 1982 |

- Interactive Dump/Restore Hardware Utility
- MENU Services
- New commands for EDT
- Modifications to Screen Format Services

All other changes are corrections or expanded descriptions applicable to features present in interactive services prior to the 8.0 release.

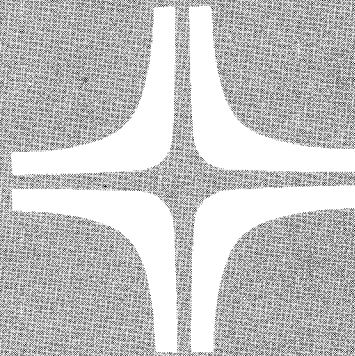
Destruction Notice: If you are going to OS/3 release 8.0, use this revision and destroy all previous copies. If you are not going to OS/3 release 8.0, retain the copy you are now using and store this revision for future use.

Copies of UP-8845 Rev. 2 and UP-8845 Rev. 2–A will be available for 6 months after the release of 8.0. Should you need additional copies of this edition, you should order them within 90 days of the release of 8.0. When ordering the previous edition of a manual, be sure to identify the exact revision and update packages desired and indicate that they are needed to support an earlier release.

Additional copies may be ordered by your local Sperry Univac representative.

Interactive Services Commands and Facilities

OS/3



User Guide/
Programmer Reference

This document contains the latest information available at the time of preparation. Therefore, it may contain descriptions of functions not implemented at manual distribution time. To ensure that you have the latest information regarding levels of implementation and functional availability, please consult the appropriate release documentation or contact your local Sperry Univac representative.

Sperry Univac reserves the right to modify or revise the content of this document. No contractual obligation by Sperry Univac regarding level, scope, or timing of functional implementation is either expressed or implied in this document. It is further understood that in consideration of the receipt or purchase of this document, the recipient or purchaser agrees not to reproduce or copy it by any means whatsoever, nor to permit such action by others, for any purpose without prior written permission from Sperry Univac.

Sperry Univac is a division of the Sperry Corporation.

FASTRAND, SPERRY UNIVAC, UNISCOPE, UNISERVO, and UNIVAC are registered trademarks of the Sperry Corporation. ESCORT, MAPPER, PAGEWRITER, PIXIE, and UNIS are additional trademarks of the Sperry Corporation.

This document was prepared by Systems Publications using the SPERRY UNIVAC UTS 400 Text Editor. It was printed and distributed by the Customer Information Distribution Center (CIDC), 555 Henderson Rd., King of Prussia, Pa., 19406.

PAGE STATUS SUMMARY

ISSUE: Update B – UP-8845 Rev. 3
RELEASE LEVEL: 8.1 Forward

| Part/Section | Page Number | Update Level |
|------------------|-------------|--------------|
| Cover/Disclaimer | | Orig. |
| PSS | 1 | B |
| Preface | 1 | A |
| | 2 | Orig. |
| | 3 | A |
| | 4 | A |
| Contents | 1 | Orig. |
| | 2 | A |
| | 3 thru 6 | Orig. |
| PART 1 | Title Page | Orig. |
| 1 | 1,2 | Orig. |
| 2 | 1 thru 15 | Orig. |
| PART 2 | Title Page | Orig. |
| 3 | 1 thru 4 | Orig. |
| | 5 | B |
| | 6 | Orig. |
| | 6a | Orig. |
| | 7 thru 17 | Orig. |
| | 18 | B |
| | 19 thru 24 | Orig. |
| | 25,26 | B |
| | 26a | A |
| | 27 thru 37 | Orig. |
| | 4 | 1,2 |
| 2a | | Orig. |
| 3 thru 9 | | Orig. |
| 10 | | B |
| 11 | | Orig. |
| 12 | | A |
| 12a | | B |
| 12b | | B* |
| 13,14 | | Orig. |
| 15 | | A |
| 16 | | Orig. |
| 17 | | A |
| 18,19 | | Orig. |
| 20 | | A |
| 21 | | Orig. |
| 22 | | A |
| 23,24 | | B |
| 25 thru 41 | | Orig. |
| 42,43 | | B |
| 44 thru 70 | | Orig. |
| 70a, 70b | | Orig. |
| 71 thru 101 | Orig. | |

| Part/Section | Page Number | Update Level |
|--------------------|-------------|--------------|
| PART 3 | Title Page | Orig. |
| 5 | 1 thru 6 | Orig. |
| | 7 | A |
| | 8 thru 20 | Orig. |
| | 20a | Orig. |
| | 21 thru 31 | Orig. |
| PART 4 | Title Page | Orig. |
| Appendix A | 1 thru 8 | Orig. |
| | 9 | A |
| | 10 thru 21 | Orig. |
| Appendix B | 1,2 | A |
| | 3,4 | Orig. |
| | 5 | A |
| | 6 | B |
| | 7 | A |
| | 8 | Orig. |
| | 9 | A |
| | 10 | B |
| 11 | A | |
| Appendix C | 1,2 | Orig. |
| | 3 | A |
| | 4 | Orig. |
| Appendix D | 1,2 | Orig. |
| Appendix E | 1,2 | Orig. |
| Index | 1 thru 5 | Orig. |
| | 6 | A |
| User comment Sheet | | |

| Part/Section | Page Number | Update Level |
|--------------|-------------|--------------|
| | | |

*New pages

All the technical changes are denoted by an arrow (⇒) in the margin. A downward pointing arrow (↓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (↑) is found. A horizontal arrow (⇒) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.



Preface

This manual is provided to help you use the interactive services of SPERRY UNIVAC Operating System/3 (OS/3). It can serve as a reference manual, to be kept with your workstation for ready use. It can also help you to familiarize yourself with the interactive commands, the procedures for initializing the interactive services and using the workstation, and interactive facilities such as interactive data utilities.

NOTE:

In this manual, you will encounter the term "workstation" a great deal. For purposes of this manual, we define a workstation as the terminal through which you use the OS/3 interactive services. Included in this definition are the workstation itself (UTS 20/20D or UTS 40/40D), UNISCOPE terminals, UTS 400 terminals, and the System 80 console workstation. These are all used in essentially the same way to communicate with the system. The differences in their operation are covered in Appendix A. The Series 90 system console cannot be used as a workstation.

This manual is divided into the following:

■ PART 1. USING INTERACTIVE SERVICES AT A WORKSTATION

Section 1. Introduction

Provides a brief overview of interactive services and describes the conventions used in describing commands in this manual.

Section 2. Communicating with Interactive Services and the System

Describes the two modes in which interactive services operate. Gives general information about the LOGON and LOGOFF commands and procedures, entering commands, and responding to messages at the workstation.

■ PART 2. INTERACTIVE SERVICES COMMANDS

Section 3. Running Jobs from the Workstation

Describes the procedures for running jobs from a workstation, including an explanation of interactive job control, the commands used to schedule and run jobs from the workstation, the EXECUTE facility, and the concept of a master workstation.

Section 4. General Workstation Commands

Describes the commands used to control job execution, run utility routines, and control the spooling process from the workstation.

■ PART 3. A GUIDE TO OTHER INTERACTIVE PRODUCTS

Section 5. Quick-Reference Guide to the Interactive Facilities, Distributed Data Processing, BASIC, and ESCORT

Provides reference information on the use of the interactive data utilities, the interactive dump/restore utility, the general and RPG II editors, the COBOL editor, the error file processor, screen format services, menu services, distributed data processing, and the interactive programming languages: BASIC and ESCORT.

■ PART 4. APPENDIXES

Appendix A. Interactive Terminals

Describes the different procedures used when accessing the OS/3 interactive services from the workstation, System 80 console workstation, UNISCOPE 100 and 200 terminals, and UTS 400 terminals. Includes the standard terminal dialog of the ICAM (Integrated Communications Access Method) terminal support facility.

Appendix B. Workstation Commands

Presents a quick reference to the interactive commands and their formats.

Appendix C. Sample Workstation Sessions

Presents four sample workstation sessions, each showing the interactive commands being used to accomplish a typical system task.

Appendix D. Positional Format for File Parameters

Explains the usage of the positional format to specify command file parameters.

Appendix E. Function Key Summary

Lists the function keys used by the various interactive facilities.

NOTE:

This manual contains references directing you to other manuals in the OS/3 system library. When you refer to those other manuals, make certain you are referring to the manual corresponding to the hardware on which you are running OS/3. For some OS/3 manuals, there are different UP-numbers for use with different hardware. The following table lists the correct UP-number of the manual you should use, depending on your hardware. Also, be sure to use the most current version of the manual.

| Manual Title | UP-Number | |
|--|-----------|-----------|
| | Series 90 | System 80 |
| Spooling and Job Accounting Concepts and Facilities | 8869 | 8869 |
| Screen Formatting Concepts and Facilities | 8802 | 8802 |
| Job Control User Guide | 8065 | 8065 |
| COBOL Editor User Guide/ Programmer Reference | 9106 | 9106 |
| Consolidated Data Management Concepts and Facilities | 8825 | 8825 |
| Data Utilities User Guide/ Programmer Reference | 8069 | 8834 |
| General Editor User Guide/ Programmer Reference | 8828 | 8828 |
| RPG II Editor User Guide/ Programmer Reference | 8803 | 8803 |
| Distributed Data Processing Concepts and Facilities | 8811 | 8811 |
| System Messages Programmer/Operator Reference | 8076 | 8076 |
| System Installation User Guide/Programmer Reference | 8074 | 8839 |

(continued)

| Manual Title | UP-Number | |
|--|-----------|-----------|
| | Series 90 | System 80 |
| System Service Programs User Guide | 8062 | 8841 |
| BASIC Programmer Reference | 9168 | 9168 |
| ESCORT Programming Language User Guide | 8855 | 8855 |
| UTS 400/UTS 4000 – OS/3 Interface User Guide/Programmer Reference | 8611 | 8611 |

Contents

PAGE STATUS SUMMARY

PREFACE

CONTENTS

PART 1. USING INTERACTIVE SERVICES AT A WORKSTATION

1. INTRODUCTION

- | | | |
|------|---------------------|-----|
| 1.1. | GENERAL | 1-1 |
| 1.2. | COMMAND CONVENTIONS | 1-1 |

2. COMMUNICATING WITH INTERACTIVE SERVICES AND THE SYSTEM

- | | | |
|--------|---|------|
| 2.1. | GENERAL | 2-1 |
| 2.2. | THE DUAL-MODE NATURE OF THE WORKSTATION | 2-1 |
| 2.2.1. | WORKSTATION Mode | 2-1 |
| 2.2.2. | SYSTEM Mode | 2-2 |
| 2.2.3. | Switching between Modes | 2-2 |
| 2.3. | INITIALIZING THE WORKSTATION | 2-3 |
| 2.3.1. | LOGON Procedure | 2-3 |
| 2.3.2. | The LOGON Command | 2-4 |
| 2.3.3. | The LOGON ACCEPTED Display | 2-6 |
| 2.4. | SYSTEM-SUPPLIED MENUS | 2-7 |
| 2.5. | ENTERING COMMANDS AT THE WORKSTATION | 2-9 |
| 2.6. | MESSAGES AT THE WORKSTATION | 2-10 |
| 2.6.1. | Output Messages | 2-10 |
| 2.6.2. | Solicited Input Messages | 2-11 |
| 2.6.3. | Unsolicited Input Messages | 2-12 |
| 2.6.4. | Restoring Response Messages (REBUILD Command) | 2-12 |

| | | | |
|--------|---|--|------|
| 2.7. | CONNECTING LOCAL WORKSTATIONS TO ICAM USER PROGRAMS | | 2-13 |
| 2.8. | THE LOGOFF COMMAND | | 2-13 |
| 2.8.1. | Interactive Accounting | | 2-14 |

PART 2. INTERACTIVE SERVICES COMMANDS

3. RUNNING JOBS FROM THE WORKSTATION

| | | | |
|---------|--|----------|-------|
| 3.1. | INTRODUCTION | | 3-1 |
| 3.2. | THE JOB CONTROL DIALOG | | 3-2 |
| 3.3. | FILING JOB CONTROL STREAMS (FILE COMMAND) | (FI) | 3-4 |
| 3.4. | RUNNING JOBS (RUN/RV COMMANDS) | (RU/RV) | 3-6a |
| 3.5. | PRERUN JOB EXECUTION | | 3-10 |
| 3.5.1. | The SI/SC Commands | (SI/SC) | 3-10 |
| 3.6. | RUNNING IBM SYSTEM/3 CONTROL STREAMS (OCL/OV COMMANDS) | (OC/OV) | 3-14 |
| 3.7. | THE EXECUTE FACILITY | | 3-16 |
| 3.7.1. | Choosing Jobs to Run Interactively | | 3-16 |
| 3.7.2. | Building the Super-Set Job Control Stream | | 3-17 |
| 3.7.3. | Using the EXECUTE Command | | 3-19 |
| 3.7.4. | Creating and Running a Sample Super-Set Job Control Stream | | 3-20 |
| 3.7.5. | Error Messages | | 3-23 |
| 3.8. | DOWNLINE LOADING PROGRAMS (DLOAD) | (DLOAD) | 3-25 |
| 3.9. | THE UPLINE DUMP COMMAND FOR UTS 400 TERMINAL USERS (ULD) | (ULD) | 3-26a |
| 3.10. | CHANGING JOB SCHEDULING | | 3-28 |
| 3.10.1. | Rescheduling Jobs (BEGIN Command) | (BE) | 3-28 |
| 3.10.2. | Deferring Jobs (HOLD Command) | (HO) | 3-30 |
| 3.10.3. | Displaying the Contents of a Job Queue (DISPLAY JBQ Command) | (DI JBQ) | 3-31 |
| 3.10.4. | Deleting Jobs from a Job Queue (DELETE Command) | (DE) | 3-32 |
| 3.10.5. | Changing the Scheduling Queue of a Job (CHANGE Command) | (CH) | 3-33 |
| 3.11. | CONNECTING A WORKSTATION TO A JOB | | 3-34 |
| 3.11.1. | The CONNECT Command | (CON) | 3-34 |
| 3.12. | THE MASTER WORKSTATION | | 3-36 |
| 3.12.1. | General | | 3-36 |
| 3.12.2. | Changing the Master Workstation for a Job | | 3-36 |

4. GENERAL WORKSTATION COMMANDS

| | | | |
|----------|---|----------------------|------|
| 4.1. | INTRODUCTION | | 4-1 |
| 4.2. | COMMANDS TO CONTROL THE JOB PROCESSING ENVIRONMENT | | 4-1 |
| 4.2.1. | Asking Questions of Other Workstation Users (ASK Command) | (AS) | 4-2 |
| 4.2.2. | Canceling a Job (CANCEL Command) | (CA) | 4-3 |
| 4.2.3. | Obtaining Job Status Information (DISPLAY JS Command) | (DI JS) | 4-4 |
| 4.2.4. | Disconnecting a Workstation from a Job (FREE Command) | (FR) | 4-6 |
| 4.2.5. | Suspending Job Processing (PAUSE Command) | (PA) | 4-7 |
| 4.2.6. | Reactivating Suspended Jobs (GO Command) | (GO) | 4-8 |
| 4.2.7. | Resuming Subsystem Execution (RESUME Command) | (RES) | 4-9 |
| 4.2.8. | Altering Display Characteristics (SCREEN Command) | (SCR) | 4-10 |
| 4.2.9. | Terminating a Job at the End of a Job Step (STOP Command) | (ST) | 4-13 |
| 4.2.10. | Sending Messages to the System Operator (TELL Command) | (TEL) | 4-14 |
| 4.3. | COMMANDS TO CONTROL SPOOLING | | 4-15 |
| 4.3.1. | General | | 4-15 |
| 4.3.2. | Spooling Command Directories and Modifiers | | 4-15 |
| 4.3.2.1. | Spool File Directories | | 4-15 |
| 4.3.2.2. | Spoiled File Modifiers | | 4-16 |
| 4.3.3. | The Output Writer | | 4-17 |
| 4.3.3.1. | Function Codes for the Output Writer | | 4-17 |
| 4.3.4. | Manually Loading an Output Writer (PR/PU) | (PR/PU) | 4-20 |
| 4.3.5. | Manually Loading an Output Writer to an Auxiliary Printer (RP) | (RP) | 4-22 |
| 4.3.6. | Obtaining Spooled File Information (DISPLAY ACT and DISPLAY SPL Commands) | (DI ACT/ DI SPL) | 4-25 |
| 4.3.7. | Holding Spooled Files (HOLD SPL Command) | (HO SPL) | 4-31 |
| 4.3.8. | Releasing Held Spooled Files (BEGIN SPL Command) | (BE SPL) | 4-32 |
| 4.3.9. | Deleting Spooled Files (DELETE SPL Command) | (DE SPL) | 4-33 |
| 4.3.10. | Breakpointing Spooled Files (BRKPT Command) | (BR) | 4-34 |
| 4.4. | COMMANDS USED TO PERFORM UTILITY ROUTINES | | 4-36 |
| 4.4.1. | File Parameters | | 4-36 |
| 4.4.2. | Keyword Parameters | | 4-37 |
| 4.4.3. | Allocating Files (ALLOCATE Command) | (AL) | 4-44 |
| 4.4.4. | Adding and Modifying Library Module Comments (COMMENT Command) | (COM) | 4-47 |
| 4.4.5. | Copying Files (COPY Command) | (COP) | 4-49 |
| 4.4.6. | Assigning Commands to Function Keys (DEFKEY) Command) | (DEFKEY) | 4-56 |
| 4.4.7. | Deleting Function Key Assignments (DEFKEY Delete Command) | (DEFKEY (delete)) | 4-58 |
| 4.4.8. | Displaying Function Key Assignments (DEFKEY DISPLAY Command) | (DEFKEY DISPLAY) | 4-59 |

| | | | |
|-----------|--|----------|-------|
| 4.4.9. | Using the Interactive Commands in a Batch Environment (ENTER Command) | (ENT) | 4-60 |
| 4.4.10. | Deleting Library Files and Modules and MIRAM Files (ERASE Command) | (ER) | 4-63 |
| 4.4.11. | Obtaining File Status Information (FSTATUS Command) | (FS) | 4-65 |
| 4.4.12. | Obtaining Command and Message Assistance (HELP Command) | (HE) | 4-68 |
| → 4.4.13. | Calling Menus (MENU Command) | (MENU) | 4-70a |
| 4.4.14. | Making Printed Copies of Files (PRINT Command) | (PRI) | 4-72 |
| 4.4.15. | Making Punched Card Copies of a File (PUNCH Command) | (PUN) | 4-78 |
| 4.4.16. | Displaying All or Part of the Workstation Log File (RECALL Command) | (RECALL) | 4-83 |
| 4.4.17. | Recovering Deleted Modules (RECOVER Command) | (REC) | 4-85 |
| 4.4.18. | Inserting a Comment in a Command Stream (REMARK Command) | (REM) | 4-88 |
| 4.4.19. | Obtaining System Status Information (STATUS Command) | (STA) | 4-89 |
| 4.4.20. | Listing the Contents of a VTOC (VTOC Command) | (VT) | 4-94 |
| 4.5. | COMMANDS TO CONTROL WORKSTATION LOGGING | | 4-98 |
| 4.5.1. | Breakpointing Workstation Log Files (BRKPT LOG Command) | (BR LO) | 4-98 |
| 4.5.2. | Obtaining Workstation Log Information (DISPLAY LOG Command) | (DI LO) | 4-101 |

PART 3. A GUIDE TO OTHER INTERACTIVE PRODUCTS

5. QUICK-REFERENCE GUIDE TO THE INTERACTIVE FACILITIES, DISTRIBUTED DATA PROCESSING, BASIC, AND ESCORT

| | | | |
|--------|---|--|-------|
| 5.1. | SCREEN FORMAT SERVICES | | 5-1 |
| 5.1.1. | The Screen Format Generator | | 5-2 |
| 5.1.2. | The Screen Format Coordinator | | 5-5 |
| 5.2. | MENU SERVICES | | 5-6 |
| 5.2.1. | The Menu Generator | | 5-6 |
| 5.2.2. | The Menu Processor | | 5-7 |
| 5.3. | The OS/3 INTERACTIVE DATA UTILITIES | | 5-8 |
| 5.3.1. | Initializing the Interactive Data Utilities | | 5-9 |
| 5.3.2. | Using the Interactive Data Utilities | | 5-9 |
| 5.3.3. | The HELP Function | | 5-10 |
| 5.4. | THE INTERACTIVE DUMP/RESTORE HARDWARE UTILITY (HU) | | 5-11 |
| 5.5. | THE GENERAL EDITOR | | 5-12 |
| 5.5.1. | The Error File Processor | | 5-20a |
| 5.5.2. | The RPG Editor | | 5-23 |
| 5.5.3. | The COBOL Editor (COBEDT) | | 5-25 |

| | | |
|------|-----------------------------------|------|
| 5.6. | DISTRIBUTED DATA PROCESSING (DDP) | 5-27 |
| 5.7. | BASIC | 5-30 |
| 5.8. | ESCORT | 5-30 |

PART 4. APPENDIXES

A. INTERACTIVE TERMINALS

| | | |
|--------|--|------|
| A.1. | GENERAL | A-1 |
| A.2. | THE SYSTEM 80 WORKSTATION | A-1 |
| A.2.1. | LOGON Procedure | A-6 |
| A.2.2. | LOGOFF Procedure | A-7 |
| A.3. | UNISCOPE 100 AND 200 TERMINALS | A-7 |
| A.3.1. | Resuming Data Output | A-10 |
| A.3.2. | LOGON Procedure | A-10 |
| A.3.3. | LOGOFF Procedure | A-11 |
| A.4. | UTS 400 TERMINAL | A-11 |
| A.4.1. | Resuming Data Output | A-15 |
| A.4.2. | LOGON Procedure | A-15 |
| A.4.3. | LOGOFF Procedure | A-15 |
| A.5. | SYSTEM 80 CONSOLE WORKSTATION | A-15 |
| A.5.1. | Mode Switching | A-15 |
| A.5.2. | LOGON Procedure | A-17 |
| A.5.3. | LOGOFF Procedure | A-17 |
| A.6. | INFORMATION FOR ICAM-CONNECTED TERMINALS | A-18 |
| A.6.1. | Standard Terminal Dialog | A-18 |
| A.6.2. | SIGN-ON Command (\$\$SON) | A-19 |
| A.6.3. | SIGN-OFF Command (\$\$SOFF) | A-20 |
| A.7. | PROCEDURE TABLE | A-20 |

B. WORKSTATION COMMANDS

C. SAMPLE WORKSTATION SESSIONS

| | | |
|------|---|-----|
| C.1. | GENERAL | C-1 |
| C.2. | A SESSION TO MAKE PUNCHED CARD COPIES OF A MODULE | C-1 |
| C.3. | A SESSION TO RUN A USER JOB | C-2 |
| C.4. | A SESSION TO USE THE GENERAL EDITOR | C-3 |
| C.5. | A SESSION TO ERASE OBSOLETE FILES | C-4 |

D. POSITIONAL FORMAT FOR FILE PARAMETERS**E. FUNCTION KEY SUMMARY****INDEX****USER COMMENT SHEET****FIGURES**

| | | |
|------|--|------|
| 3-1. | Sample Programs | 3-20 |
| 3-2. | Completed Super-Set Job Control Stream | 3-22 |
| 3-3. | How the User-id Controls Access to Jobs | 3-36 |
| A-1. | Workstation TYPEWRITER Keyboard | A-2 |
| A-2. | Workstation EXPANDED TYPEWRITER Keyboard | A-2 |
| A-3. | OS/3 LOGON Request Screen | A-6 |
| A-4. | UNISCOPE Terminal Keyboard | A-7 |
| A-5. | UTS 400 Keyboard | A-11 |
| A-6. | Console Workstation Keyboards | A-16 |

TABLES

| | | |
|------|--|------|
| 4-1. | Spool File Directories | 4-16 |
| 4-2. | Print Spooling Device Numbers | 4-17 |
| 5-1. | General Editor Commands | 5-13 |
| 5-2. | Error File Processor (EFP) Commands | 5-21 |
| 5-3. | DDP Commands | 5-28 |
| A-1. | System 80 Workstation Keyboard Functions | A-3 |
| A-2. | UNISCOPE 100 and 200 Terminal Keyboard Functions | A-8 |
| A-3. | UTS 400 Terminal Keyboard Functions | A-12 |
| A-4. | Procedure Table | A-21 |
| D-1. | Positional and Keyword Parameters | D-1 |
| E-1. | Function Key Summary | E-1 |

**PART 1. USING INTERACTIVE SERVICES
AT A WORKSTATION**



1. Introduction

1.1. GENERAL

The workstation is the principal means by which you and the system communicate. Through the workstation, you can perform almost every function available on the OS/3 system. You can use the workstation to write a source program, compile and debug it, create a job control stream to run the program, and actually run the program. Through the workstation, you can also use the interactive facilities to make your data processing tasks easier and more efficient.

The interactive commands are the means by which you use the workstation to perform various system tasks. Therefore, the bulk of this manual is concerned with describing the interactive commands. To help you understand the use of the various interactive commands to do real work on the system, Appendix C of this manual contains four sample workstation sessions, showing how the interactive commands are used together to perform tasks. There are examples given with each interactive command to help you better understand the use of each command.

1.2. COMMAND CONVENTIONS

The conventions used to illustrate the commands presented in this manual are as follows:

- Underscoring of part of a command or parameter indicates that the command or parameter may be abbreviated, and only the underscored portion must be entered. However, you may enter the entire command or parameter; it will execute properly. When a command or parameter is not underscored, you must enter the entire command or parameter. In the following example, ER of the ERASE command is underscored. You must enter at least ER for the command to execute; but you may enter ERA, ERAS, or ERASE and the command will execute properly.

ERASE

- Parameters printed in lowercase letters designate undefined variables.

VSN = volume

- Parameters that are optional are enclosed in brackets.

ERASE [WRPASS = wp]

- Alternate choices for a parameter are enclosed in braces.

[ENTER DEVICE={nnn}
 {RDR}]

NOTE:

The brackets and braces just discussed are not to be entered as part of the commands; they are used only to denote optional parameters and alternate parameter choices.

- Default values are values automatically generated by the system when you do not specify a value for a parameter. Default values are shown shaded in each command format.
- You are permitted to enter commands and parameters in either uppercase or lowercase letters. The system accepts either uppercase or lowercase in all but a few special instances. When you are restricted to either uppercase or lowercase letters, a note informing you of this will be included.
- Spaces are indicated by a delta (Δ) symbol.

2. Communicating with Interactive Services and the System

2.1. GENERAL

This section covers the procedures you use to communicate with the interactive services, and through them, the system in general. Included are procedures for logging on to the system, entering commands, responding to messages, and using the two operational modes of the workstation. Also included are procedures for attaching locally-connected workstations to programs using the Integrated Communications Access Method (ICAM). One of these programs is the SPERRY UNIVAC Information Management System (IMS).

NOTE:

On System 80 workstations and console workstations, an alternate character set (besides English) is available. It is the Katakana character set, specified when you generate your system. If you use the Katakana character set, no lowercase-to-uppercase translation may be performed.

2.2. THE DUAL-MODE NATURE OF THE WORKSTATION

The workstation operates in two modes: WORKSTATION mode and SYSTEM mode. These two modes facilitate the use of the workstation as both a data transfer device for program input and output and as a control device to control processing.

2.2.1. WORKSTATION Mode

WORKSTATION mode makes the whole screen available to the user for data entry and output to user-created and system programs. To use WORKSTATION mode for data transfer, it must be connected to the program requesting or supplying the data. Connection is most often accomplished through job control (3.11).

The CONNECT command is available to connect WORKSTATION mode to a job that does not automatically connect workstations through job control. System programs requiring WORKSTATION mode automatically connect the WORKSTATION mode of the workstation from which they are invoked.

When data is output to your workstation by a program, enough data is output to fill your workstation screen. When the screen is full, the system tells the program to stop data output. Data output will not resume until you indicate that it should. There are different ways of resuming data output, depending on the type of terminal you are using as a workstation. Refer to Appendix A for detailed information on the use of each type of terminal.

When WORKSTATION mode is not connected to a job, it serves to display commands entered at the workstation, as well as any output produced by those commands. See 2.4 for a complete description of this aspect of the WORKSTATION mode function.

2.2.2. SYSTEM Mode

SYSTEM mode serves two functions. First, it is the mode used for entering unsolicited commands into the system. It is also the mode used to respond to system and program-generated messages.

SYSTEM mode uses the first two lines of the screen to display messages and accept responses. When you switch from WORKSTATION mode to SYSTEM mode, any data on the first two lines of the screen is saved in a buffer. The lines are cleared and SYSTEM mode is able to use them. Messages are displayed on line 2. You enter message responses and unsolicited commands on line 1.

When WORKSTATION mode is not connected to a job, you must still enter SYSTEM mode to enter commands and respond to messages. You still use line 1 to enter commands (the cursor automatically positions itself at the first character position of line 1). However, the rest of the screen is available for the display of messages or command output. If a command produces a message requiring a response, you always enter your response on line 1.

2.2.3. Switching between Modes

You can switch between WORKSTATION and SYSTEM modes to enter commands and respond to messages. The system can also prompt you to switch from WORKSTATION to SYSTEM mode if it has a message waiting to be displayed to you.

Since SYSTEM mode requires the top two lines of the workstation display, when you switch from WORKSTATION mode to SYSTEM mode, any data present on the top two lines of the screen is removed. The System 80 workstation and console workstation save the two lines of data, while the other terminals (the UNISCOPE 100 and 200 and UTS 400) do not save the data removed by the mode switch. Therefore, if you are working with a terminal which does not save the data, be careful when you switch from WORKSTATION to SYSTEM mode. The loss of the two lines of data could produce serious errors in some system and user programs. Consider waiting to acknowledge the message until you have finished with the contents of the screen.

When WORKSTATION mode *is not* connected to a job or system program, the workstation will automatically be switched back from SYSTEM mode to WORKSTATION mode. This permits the use of the entire workstation screen for the display of command output.

When WORKSTATION mode *is* connected to a job or system program, you must manually return to WORKSTATION mode after you have completed your tasks in SYSTEM mode. The system does not permit you to return to WORKSTATION mode, however, if a message requiring a response has not been answered.

Each type of terminal has a different procedure for switching between modes. For information on the correct procedure for the terminal you use, see Appendix A.

2.3. INITIALIZING THE WORKSTATION

Before you can use either mode of the workstation, you must prepare it for operation. This involves entering the LOGON command.

2.3.1. LOGON Procedure

Every time you use the workstation, you must begin by entering the LOGON command. The LOGON command serves a variety of purposes. Through its parameters, you are identified to the system as a legitimate user, and the scope of your activities on the system is defined.

The logon bulletin supplies you with information about the operation of your system. In addition, a workstation log file is maintained that consists of system messages, your responses, and all the commands you issued during your workstation session. Included in the workstation log file are accounting records that keep track of things like the amount of computer time you used. The logon procedure also lets you specify whether or not you want a printed listing of the workstation log file when your workstation session is finished.

When the workstation completes the power-on confidence test and requests that you log on, it is in WORKSTATION mode. In order to log on, the workstation must be in SYSTEM mode. When you are in SYSTEM mode, you may enter the LOGON command as detailed in 2.3.2. To send your LOGON command to the system for processing, press the XMIT key when you have completed entry of the command.

If you are unfamiliar with the format of the LOGON command or with the procedure for entering the LOGON command, the system will help you. Press the XMIT key. The screen will display a menu listing the parameters of the LOGON command and providing spaces in which to enter the parameters.

The system bulletin is displayed if you leave YES in the spaces following OPTIONS: BULLETIN. If you do not wish to see the system bulletin, overwrite YES with NO. The log is printed if you leave YES in the spaces following LOG. In order to suppress the log, overwrite the YES with NO. Just enter your user-id, account number, and password (if these have been supplied to you by your system administrator).

The system has automatically entered LOGON and is ready to receive the information supplied by the parameters you enter in the menu. When you have completed entry of your logon information, press XMIT again to send the logon information to the system. The logon menu looks like this:

```

OS/3 INTERACTIVE SERVICES

LOGON IDENTIFICATION:  USER-ID          >____-<
                       ACCOUNT NUMBER   >____-<
                       PASSWORD         >____-<

OPTIONS:               EXECUTION PROFILE >____-<
                       BULLETIN        >YES <
                       LOG              >YES <

```

The following screen shows the logon menu screen filled in with sample parameters:

```

OS/3 INTERACTIVE SERVICES

LOGON IDENTIFICATION:  USER-ID          >Payrol<
                       ACCOUNT NUMBER   >____-<
                       PASSWORD         >0339__<

OPTIONS:               EXECUTION PROFILE >editing_<
                       BULLETIN        >YES <
                       LOG              >YES <

```

↓
NOTE:

When you log on at either a UTS40, UTS40D, or UTS400, the password entry is not displayed.

↑

2.3.2. The LOGON Command

The following is the format of the LOGON command:

→ `LOGON△user id[,acct][,password][,exec-pro] [,BULLETIN={NO
YES}] [,LOG={YES
NO}]`

Parameters:

user-id

Is a 1- to 6-character alphanumeric code you enter to identify yourself to the system. The user-id is used by the system to correctly route messages, job and command output, and to determine which commands you may use on the system. *The user-id must begin with an alphabetic character.*

acct

Is a 1- to 4-character alphanumeric code that is used for system time accounting.

password

Is a 1- to 6-character alphanumeric code that controls your access to the overall system.

exec-pro

Specifies an execution profile attached to your user-id. Execution profiles are series of commands that are automatically executed by the system when you log on under a particular user-id. A typical use of an execution profile would be to go directly from LOGON to the general editor. Each user profile can have a default execution profile, which will be invoked if none is specified with the LOGON command. The default can also be *no* execution profile. A user-id may have any number of different execution profiles, but the system must know them as legitimate for the user-id. The system administrator controls which execution profiles may be invoked by different user-ids. To specify an execution profile, you enter the name assigned to the execution profile. It may be from one to eight alphanumeric characters. In the sample menu screen in 2.3.1, the execution profile is named *editing*.

BULLETIN={ NO }
{ YES }

Specifies whether or not you wish to see the system bulletin after your LOGON command has been accepted. The system bulletin is a display of one or more lines detailing current operational information, such as hours of operation, or peripherals in or out of service. If you do not specify that you *do not* wish to see the system bulletin, it will be displayed.

LOG={ YES }
{ NO }

Specifies whether or not you want the log of workstation commands and messages printed when you conclude your workstation session. See 4.5 for more information on workstation logging. Accounting records are included at the end of the workstation log file. Therefore, if you suppress printing of the log file, you also suppress printing of the accounting records. (Interactive accounting is discussed in detail in 2.8.1.)

NOTE:

If you enter the LOGON command with a parameter omitted, you must enter a comma for that parameter. For example, if you enter LOGON with a user-id and password but no account number, it would be written like this:

```
LOGON USRA,,SECRET
```

You do not need to enter additional commas between the last positional parameter you enter and the

BULLETIN={ YES }
{ NO }

parameter. The system can discern the keyword parameter from the positional parameters preceding it.

Example:

```
LOGON BILLP,4789,BULLETIN=NO,LOG=NO
```

In this example, a user has logged on with the user-id BILLP and the account number 4789. The system bulletin is not displayed because BULLETIN=NO is specified. The workstation log won't be printed at the end of the workstation session because LOG=NO is specified.

The following conditions could cause this command to be rejected:

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command parameters were entered improperly.

■ Insufficient Resources

The system does not have sufficient resources available to support another interactive user.

■ Nonunique User-id

Another user is already logged on the system with the user-id you entered in your LOGON command. Each user must have a *unique* user-id.

If your LOGON command is not accepted, the following message will be displayed on the screen:



```
IS23 INVALID FORMAT FOR THE LOGON COMMAND, REENTER LOGON
```

The workstation will have been shifted back to WORKSTATION mode to display this message. In order to retry the LOGON command, you must again enter SYSTEM mode, or, to see the menu screen, press the XMIT (or TRANSMIT) key. Reenter the LOGON command on the first line of the screen. The incorrect LOGON command will remain on the first line of the workstation screen if it is rejected.

2.3.3. The LOGON ACCEPTED Display

When your LOGON command is processed and accepted by the system, you will see a display, rolling up from the bottom of the screen, informing you that your LOGON is accepted, and the system bulletin if you so specified. The following is a typical LOGON ACCEPTED display:

```

OS/3 INTERACTIVE SERVICES
LOGON ACCEPTED AT 12:45:39 ON 80/01/28. REV. 7.0
TODAY'S BULLETIN IS:
*****
* SYSTEM WILL BE AVAILABLE FROM 09:00 TO 17:00 TODAY *
*****

```

If your site administrator does not provide a bulletin, the system will display the default bulletin. The default bulletin contains instructions for entering commands to the system. The following display shows the default bulletin.

```

IS27 TODAYS BULLETIN IS:
-- TO TYPE IN COMMANDS, DEPRESS 'FUNCTION' AND --
-- 'SYSTEM MODE' KEYS SIMULTANEOUSLY, THEN TYPE --
-- THE COMMAND AND DEPRESS TRANSMIT. --
-- ON UNISCOPES DEPRESS 'MESSAGE WAITING' KEY. --

```

2.4. SYSTEM-SUPPLIED MENUS

After your LOGON command is accepted, your workstation may display a menu, requesting that you select a system function. This is the first of a series of menus supplied by Sperry Univac to aid you in relating to and using the interactive facilities of OS/3.

The menus allow you to select the system function you need. Through help displays, they assist you in understanding the different system functions.

The SPERRY UNIVAC standard menus are displayed to you after logging on, provided the MENU command is present in your execution profile. If the menus are not displayed, you may see them by entering:

```
MENU
```

The following is an illustration of the first of the menus supplied by Sperry Univac:

```

*****
*
*                               *
*          SYSTEM MENU: OS301   *
*    1. RUN A JOB                3. END MENU   *
*    2. PERFORM A SYSTEM FUNCTION 4. LOGOFF   *
*
* FOR HELP ON A PARTICULAR ITEM NUMBER, ENTER A QUESTION MARK *
* FOLLOWED BY THE ITEM NUMBER (?N). HELP FOR THE ENTIRE DISPLAY *
* CAN BE ACQUIRED BY ENTERING A QUESTION MARK (?) THEN PRESSING *
* TRANSMIT.
*
*                               *
*          ENTER SELECTION: 2   *
*
*****

```

Making certain selections on this menu causes other menus to be displayed, giving you further choices of system facilities. For example, we've selected item 2 "SYSTEM FUNCTIONS" on the previous menu screen. By entering item 2, we cause the following menu to be displayed.

```

*****
*
*                               *
*          SYSTEM FUNCTION MENU: OS314       *
*
* 1. DATA UTILITIES              7. EDITOR   *
* 2. BASIC                        8. JCL DIALOG *
* 3. SCREEN FORMAT GENERATOR      9. SYSGEN DIALOG *
* 4. MENU GENERATOR              10. HARDWARE UTILITIES (HU) *
* 5. RPG EDIT                    11. RETURN TO SYSTEM MENU *
* 6. DDP
*
* OPTIONAL SOFTWARE. CONTACT YOUR SPERRY UNIVAC MARKETING OFFICE *
*
*                               *
*          ENTER SELECTION: ?2   *
*
*****

```


When WORKSTATION mode is *not* connected to a job or system program, entering commands requires a slightly different procedure. You may still enter commands only when the workstation is in SYSTEM mode. However, after you have entered a command, the system will switch back into WORKSTATION mode and use the entire screen to display command output. The command you entered on the first line will be redisplayed (*echoed*) on the bottom line of the screen. Output from a command will roll up from the bottom line of the screen. If the output is longer than the screen capacity, the system will continue to roll up the output, with the first lines of output being rolled off the second line of the screen and lost. If this happens, you can stop the output by switching to SYSTEM mode and viewing the output one line at a time. If you remain in WORKSTATION mode to view the output, you must return to SYSTEM mode before you can enter another command.

If a command is unacceptable, the system responds with an error message informing you why the command is unacceptable. This message can be either a NAK (negative acknowledgment) message or a prefixed message. NAK messages appear in the last 14 character positions of the system message line. The up-to-12-character message is bracketed by blink characters (Δ Δ). Prefixed error messages and NAK messages can be found, with explanations of their meaning and actions to be taken, in the OS/3 system messages manual.

2.6. MESSAGES AT THE WORKSTATION

Messages are generated by commands, system programs, and users for the purpose of user-system and user-program communication. Messages are divided into three categories. The first is output messages, which are produced by the system, system programs, commands, or user programs. These messages provide you with information, direct you to take some action, or ask a question requiring your response. The second is solicited input messages. These are messages you enter in response to an output message requiring a response. The third category is unsolicited input messages. These are messages you enter that are not in response to an output message.

2.6.1. Output Messages

Output messages are displayed on the second (system message) line of the workstation. They are in the following format:

$$jj i \left\{ \begin{array}{l} ? \\ \Delta \\ * \end{array} \right\} \text{message-text}$$

where:

jj

Is a 1- or 2-digit number assigned to each active job in the system. The numbers 1 through 7 (Series 90 hardware) or 1 through 14 (System 80 hardware) are assigned to user jobs as they become scheduled for execution. They are used in output messages to identify the job producing the message. Messages produced by the system itself take the number 0.

i

Is a 1-character message-id. Message-ids are consecutively assigned to output messages. They begin with the letter A and end with the letter Q. (The letters I and O are omitted from the sequence to avoid possible confusion with job numbers 1 and 0.) Message-ids are used together with job numbers to explicitly identify each message in the system. *When an output message requires a reply, the reply message must be prefixed with the job number and message-id of the message requesting the reply.* Unsolicited input messages are identified by a message-id of zero. Thus, an unsolicited message to job number 1 would have the prefix 10, and an unsolicited message to the system would have the prefix 00.

?

Identifies an output message that must be answered before the job, command, or system program that issued the message can continue. If the WORKSTATION mode of your workstation is connected to a job, and you have switched to SYSTEM mode to receive messages, you may *not* return to WORKSTATION mode until all messages requiring responses have been answered.

△

Identifies an output message that requires no reply or action. It is displayed for your information only. Input messages, solicited and unsolicited, must have a space between the message-id and the message text.

*

Identifies an output message that requires some action be taken. Execution pauses for the job that produced the message. You must issue a GO command to reactivate the job.

message-text

The actual message content. Message texts may be a maximum of 60 characters. If the message is longer than 60 characters (as in some messages generated by the COBOL and RPG II compilers), the message is truncated.

2.6.2. Solicited Input Messages

Solicited input messages are messages you enter in response to an output message requiring a reply (question mark immediately follows message-id). The format for solicited input messages is:

```
jj i△message-text
```

where:

jj i

Identifies the job number and message-id of the message being responded to. *A space must be included between the message-id and the message text.*

message - text

Is the actual text of the reply.

2.6.3. Unsolicited Input Messages

Unsolicited input messages are those messages you enter that are not in response to an output message requiring a reply. The format for unsolicited messages is as follows:

jj \emptyset message - text

where:

jj

Is the job number of the job you want to receive the message. If the message is directed to the system, the job number is 0.

\emptyset

Zero is always the message-id that must be used to identify an unsolicited message. The message-id must be followed by a space.

message - text

Is the actual text of the message. It may be up to 60 characters long.

NOTE:

If you had been using the workstation for data entry before you changed modes to respond to the message, be sure the last screen of data you were working with was transmitted to the system. The audible alarm will sound once when the XMIT key is pressed and data transfer is not successful.

→ 2.6.4. Restoring Response Messages (REBUILD Command)

During your use of the workstation, especially in WORKSTATION mode, you may inadvertently clear the screen while there is an outstanding message requiring a response, or such a message may be rolled off by command output. A command is available to restore messages requiring a response or a GO command to the screen. The REBUILD command reconstructs any outstanding response messages in the system for your workstation.

Format:

REBUILD

There are no parameters associated with this command.

If you have lost a message, or *think* you have lost a response message, key in RE. If there are any outstanding response messages, they will be redisplayed to you. If there are no outstanding messages, you receive no response.

2.7. CONNECTING LOCAL WORKSTATIONS TO ICAM USER PROGRAMS

Some of the system programs of the OS/3 operating system interface with users through terminals connected by the Integrated Communications Access Method (ICAM). One of these system programs is the Information Management System (IMS). You can access such ICAM-connected programs from a directly connected (local) workstation by logging on, entering SYSTEM mode, and issuing the standard ICAM terminal sign-on command, \$\$\$SON. When you have finished with the program, enter SYSTEM mode, and issue the ICAM terminal sign-off command, \$\$\$SOFF. For complete information on the ICAM sign-on and sign-off commands, see Appendix A.

2.8. THE LOGOFF COMMAND

When you have finished all the tasks you want to do with the system and are ready to end your workstation session, enter the LOGOFF command. If you enter LOGOFF while one of the interactive facilities (such as the general editor) is running, the LOGOFF command will be rejected.

If you enter this command from a terminal connected through ICAM, your session will be ended and another user may log on. You must enter the ICAM sign-off command (\$\$\$SOFF) before the terminal will be released. See Appendix A for information on the ICAM sign-off command.

Any programs you initiated that are running and that do not require the workstation for input or output purposes will continue to run, with messages directed to the system console.

Format:

```
LOGOFF
```

There are no parameters associated with this command. *It may not be abbreviated.*

The following conditions may cause this command to be rejected:

- Functions Still in Use

You may not log off the workstation if an interactive services function you called from the workstation is still running.

- Workstation Is Reserved

The workstation is either allocated or reserved for a job.

- Unanswered Messages

You have not answered all the messages requiring a response sent to your workstation.

■ Incorrect Syntax

The command was misspelled or not entered in its entirety, or the command was entered with parameters appended.

If your LOGOFF command is not accepted by the system, the following message will be displayed:

```
IS74 LOGOFF IGNORED, WORKSTATION IS STILL IN USE
```

When your LOGOFF command is accepted by the system, the following message will be displayed:

```
IS73 LOGOFF ACCEPTED AT ___:__:__ ON __/__/__
```

The blank spaces show the time (hh:mm:ss) and date (yy/mm/dd) of logging off.

↓ In addition, after your logoff is accepted, a listing of the workstation log file for your workstation session is printed (unless you specified LOG=N when you logged on or unless you generated your system without workstation logging through the SYSGEN parameter, CONSOLOG).

↑ Your workstation log automatically shows all the commands you entered while in system mode. If you want your log to include commands entered both in system and workstation mode (EDT commands, for example), you can specify that with the LOG parameter of the SCREEN command (4.2.8).

↓ The OS/3 ...PLEASE LOGON screen display appears again until the workstation is shut off or signed off from ICAM, or until another user logs on.

2.8.1. Interactive Accounting

Accounting information is maintained for every workstation session. When you log off, this information is summarized and added to the end of your workstation log file. These accounting records appear as the last four messages on the workstation log file listing you get when you log off. (If you specified LOG=N when you logged on, you do not get a listing of account records, although the system still accumulates the records.)

↑ Accounting records list such things as CPU time used, the number of files you used, and the number of commands you issued. Here is a sample accounting listing:

LOGOFF

OJ IS73 LOGOFF ACCEPTED AT 10:08:35 ON 82/05/20

AC50 USER-ID=PHIL ACCT NO= LOGON AT 09:45:48.365 LOGOFF AT 10:08:36.049 CONNECT TIME 00:22:47.684
AC51 CPU TIME USED=00:00:26.818 TASK PRIORITY=01 DATE=82/05/20 NUMBER OF EXCP'S=00003329
AC52 NUMBER OF: COMMANDS=00010 FILES ACCESSED=00018 SVC CALLS=00005045 TRANSIENT CALLS=00000239
AC53 DEVICE EXCP'S 102=00002628 315=00000101 103=00000600

W 10:08:35
A 10:08:36
A 10:08:36
A 10:08:36
A 10:08:36





PART 2. INTERACTIVE SERVICES COMMANDS



3. Running Jobs from the Workstation

3.1. INTRODUCTION

This section describes the method you use to run jobs from the workstation. A job can be either a series of programs you have created, or an interactive facility supplied by Sperry Univac. A job occupies a job slot, and is scheduled and executed by either the RUN/RV, SC/SI, or OCL/OV commands.

The first step in running a job on OS/3 is the creation of a job control stream. The job control stream carries information to the system about the job, such as which files are used by the job. You can create job control streams interactively by using the job control dialog.

When you have created your job control stream, it may be stored on the job control stream library file (\$Y\$JCS) or an alternate file of your designation. You may also store job control streams or job control procedures (jprocs) on \$Y\$JCS or an alternate file by using the FILE command, discussed later in this section.

To run your job, the system must first read the job control stream you created for the job. After it has read the job control stream it must expand any job control procedures (jprocs) in the job control stream. A job control procedure is a series of job control statements, represented in a job control stream by a single statement. When a jproc is expanded, the single statement is replaced by the series of statements it represents. The system then schedules your job for execution so that all the system resources it needs will be available to it and finally executes it. This entire process is handled by the RUN/RV, OCL/OV, and SC/SI command processors.

When your job control stream is read, your job is scheduled for execution by being placed on a job scheduling queue, where it waits its turn for system resources. There are three job scheduling queues available for use based on the priority of your job. They are *preemptive* for jobs that must run immediately, *high* for jobs that must run as soon as possible, and *normal* for jobs that can run whenever the necessary system resources are available. These priorities are discussed in more depth in connection with the RUN, OCL, and SI commands. While your job is waiting in a scheduling queue for execution, you may change its scheduling queue. You may defer it from being scheduled for execution, reinstate it in a scheduling queue for execution, or remove it entirely from the scheduling queue. You may also produce a display of the contents of the three job queues to check on the progress of your job through the queue. The BEGIN, CHANGE, HOLD, DELETE, and DISPLAY commands detailed in 3.10.1 through 3.10.5 allow you to perform these functions.

3.2. THE JOB CONTROL DIALOG

You use the job control dialog to interactively build job control streams. Through menu screen displays, you are offered choices of statements needed to create a job control stream. The system prompts you through every step of creating a job control stream, tailored to the specific needs of your job.

You initialize the job control dialog by entering the RV command with the jobname JC\$BLD; thus:

```
RV△JC$BLD
```

When you enter this command, your workstation screen clears, and a display appears introducing the job control dialog:

```
PROGRAM=          DIALOG FOR JOB CONTROL
THIS DIALOG PREPARES A JOB CONTROL STREAM OR PROCEDURE(JPROC).  FOR AN
EXPLANATION OF THE DIALOG PROCESS, ENTER 'HELP' IN THE SPACE PROVIDED.  ----
```

If you choose to go on, and do not require an explanation of the dialog process, another screen will be displayed, the actual beginning of the job control dialog:

```
JOB CONTROL MODULE TYPES:
USE THIS MENU TO SELECT THE TYPE OF MODULE TO BE PREPARED:
1.  JOB CONTROL STREAM
2.  USER WRITTEN JOB CONTROL PROCEDURE (JPROC)
3.  HELP
SELECT ITEM BY ENTERING A NUMBER△__
```

If you need an explanation of the types of modules to be prepared, enter the number 3 in the space provided, as shown in the preceding illustration. Entering HELP will cause the following two screens to be displayed, explaining what a job control stream and JPROC are:

IN ORDER TO EXECUTE ANY JOB, IT IS NECESSARY TO CONVEY TO THE COMPUTER EXACTLY WHAT YOU WANT TO DO, AND WHAT RESOURCES (PRINTER, READER, DISKS, ETC) ARE NEEDED. THIS IS ACCOMPLISHED THROUGH THE USE OF JOB CONTROL. THERE ARE TWO TYPES OF JOB CONTROL MODULES. THE COLLECTION OF JOB CONTROL STATEMENTS USED TO RUN A JOB IS CALLED A JOB CONTROL STREAM, SOMETIMES REFERRED TO AS THE JOB STREAM OR CONTROL STREAM. IN IT, THERE MAY BE JOB CONTROL STATEMENTS, CALLS TO SYSTEM SUPPLIED PROCEDURES, AND THE SECOND TYPE OF MODULE - USER-WRITTEN PROCEDURES (JPROCS).

PUSH TRANSMIT KEY TO PROCEED

JOB CONTROL PROCEDURES HAVE TWO PARTS - THE DEFINITION AND THE CALL. THE DEFINITION IS THE JPROC MODULE CREATED BY THE DIALOG. THE CALL IS A STATEMENT IN THE CONTROL STREAM WHICH HAS THE JPROC NAME AS THE COMMAND, AND PROVIDES ANY NECESSARY PARAMETERS. THE JPROC CALL IS USED AS AN ABBREVIATION TO PREVENT CODING THE DEFINITION MANY TIMES. WHEN THE CONTROL STREAM IS PROCESSED, EACH CALL IS REPLACED BY THE APPROPRIATE DEFINITION WHICH HAS BEEN PUT AT THE BEGINNING OF THE STREAM OR STORED IN A SYSTEM FILE (\$Y\$JCS). THE RESULT IS THE SAME AS IF THE DEFINITION HAD BEEN CODED INSTEAD OF THE CALL.

PUSH TRANSMIT KEY TO PROCEED

The dialog continues as you enter information, make choices from menus, and receive help as you need it. When you have completed a job control stream, it is stored in the job control stream file, \$Y\$JCS, or an alternate file you can specify.

NOTE:

This section is written to give you an overview of job control and building job control streams interactively. For detailed information, refer to the OS/3 job control user guide.



FI

3.3. FILING JOB CONTROL STREAMS (FILE COMMAND)

The FILE command files jobs and jprocs, read from an input device to the permanent job control stream library file (\$Y\$JCS) or to an alternate SAT library file.

Format:

$$\text{FILE} \left\{ \begin{array}{l} ([did], label) \\ (RDR, label) \end{array} \right\} \left[\begin{array}{l} :alt-filename \\ \left(\begin{array}{l} :alt-filename, \left\{ \begin{array}{l} RES \\ RUN \\ vsn \end{array} \right\} \\ :alt-filename, \left[\begin{array}{l} RES \\ RUN \\ vsn \end{array} \right], write-pass \end{array} \right) \end{array} \right]$$

Parameters:

$([did], label)$

Specifies a diskette volume from which the job control stream or jproc is read. *did* is a 3-digit hardware address that specifies the drive on which the diskette is mounted. The first digit is the channel number; the second and third digits are the actual hardware address. The *did* is usually physically displayed on the device it represents. *label* specifies the name of the diskette data set containing the job control stream or jproc that is being filed. The diskette must have been recorded in data set label mode. The record size must be 128 bytes or less. The records must be unblocked and unspanned. *This parameter must be enclosed in parentheses.*

NOTE:

For more detailed information on the did, refer to the OS/3 system installation user guide/programmer reference.

$(RDR, label)$

Specifies that the job control stream or jproc you wish to store is located in a subfile of the input spool file. RDR specifies that the job control stream or jproc is located on the input spool file. *label* specifies the name of the subfile containing the job control stream or jproc. *This parameter must be enclosed in parentheses.*

NOTE:

A file in the reader queue can be accessed only once. So once you execute the file command, you can no longer access the file from the reader queue.

:alt-filename

Specifies the name of the alternate file, residing on the SYSRES disk, that is to receive the job control stream or jproc. The alternate file must have been allocated as a SAT file. If the alternate file is cataloged, you need only enter the name of the alternate file; the volume serial number for the file found in the catalog is used. You specify only the name of the file if no password is needed to write to the file. If you do not specify an alternate file, the job control stream or jproc will be filed in \$Y\$JCS.

$$: \left(\text{alt-filename}, \begin{Bmatrix} \text{RES} \\ \text{RUN} \\ \text{vsn} \end{Bmatrix} \right)$$

Specifies the name of the alternate file *and* identifies a volume serial number for the file. The alternate file must have been allocated as a SAT file. The alternate library file may reside on either a disk, or a diskette recorded in format label mode. If you want the alternate file to reside on the SYSRES disk, specify RES. If you want the alternate file to reside on the \$Y\$RUN disk, specify RUN. If you want the file to reside on another volume, specify the volume serial number for the volume. The volume serial number may be from one to six characters long. If another file with the same file name is in the catalog, the volume serial number you enter with the command distinguishes between the two files and overrides the catalog volume serial number. There must not be a password for the alternate file specified in the catalog.

$$: \left(\text{alt-filename}, \begin{Bmatrix} \text{RES} \\ \text{RUN} \\ \text{vsn} \end{Bmatrix}, \text{write-pass} \right)$$

Specifies the name of the alternate file, a volume serial number if needed or wanted, and a password, identified in the catalog, required to write to the alternate file. The alternate file must have been allocated as a SAT file. The volume serial number is specified as before. If you do not enter a volume serial number, the one listed in the catalog will be used.

NOTE:

You may not use the FILE command to access an actual card reader. You may use spooled readers only.

Examples:

```
FILE(320,MYJOB)
```

This command files a job control stream named MYJOB to \$Y\$JCS on SYSRES. The stream is stored on a diskette mounted on the device whose channel and address are 320. When the file command has successfully filed the stream, the user sees this message:

```
JC26 MYJOB   FILED ON REL080 IN $Y$JCS
```

In this example, the volume serial number of the SYSRES disk is RELO80.

↓
FILE(321,MYJOB) :(MYFILE,RES)

This command files a job control stream named MYJOB to an alternate file named MYFILE on SYSRES. The stream is stored on a diskette whose channel and address are 320. When the file command has successfully filed the stream, the user sees this message:

JC26 MYJOB FILED ON REL080 IN MYFILE

In this example, the volume serial number of the SYSRES disk is REL080.

FILE(RDR,MYJOB2)

This command files a job control stream named MYJOB2 to \$\$JCS on SYSRES. This stream is stored on the reader queue in the input spool file. When the file command has successfully filed the stream, the user sees this message:

JC26 MYJOB2 FILED ON REL080 IN \$\$JCS

↑
In this example, the volume serial number of the SYSRES disk is REL080.

RU/RV

3.4. RUNNING JOBS (RUN/RV COMMANDS)

The RUN/RV commands cause the job control stream associated with your job to be read and expanded, and your job to be scheduled for execution. The RUN command reads job control streams from the \$\$JCS library file, an alternate job control library file, a data set label diskette, or the input spool file. You use the RUN command when the job requires card-image input, either from a data set label diskette or the input spool file. The RV command reads job control streams only from the \$\$JCS file or an alternate job control library file, and is used when no card-image input is required. You may not specify an input device in the RV command.

NOTES:

1. *When you use the RUN command with a data set label diskette as input device, make sure, prior to entering the command, that the diskette you need is mounted on a diskette drive.*
2. *You may not use the RUN command to run a job requiring the use of an actual card reader; you may only use spooled readers.*

The RV command is used when no input device is required.

Format:

$$\text{RUN} \left\{ \begin{array}{l} ([did], label) \\ (RDR, label) \end{array} \right\} \Delta [jobname] [(new-name)] \left[\begin{array}{l} :alt-filename \\ \left(:alt-filename, \begin{array}{l} \{ RES \\ RUN \\ vsn \} \end{array} \right) \\ \left(:alt-filename, \begin{array}{l} \{ RES \\ RUN \\ vsn \} \end{array}, read-pass \right) \end{array} \right] \leftarrow$$

$$\left[\begin{array}{l} \{ PRE \\ HIGH \\ NOR \} \end{array} \right] [, key-1=val-1, \dots, key-n=val-n]$$

$$\text{RV} \Delta [jobname] [(new-name)] \left[\begin{array}{l} :alt-filename \\ \left(:alt-filename, \begin{array}{l} \{ RES \\ RUN \\ vsn \} \end{array} \right) \\ \left(:alt-filename, \begin{array}{l} \{ RES \\ RUN \\ vsn \} \end{array}, read-pass \right) \end{array} \right] \left[\begin{array}{l} \{ PRE \\ HIGH \\ NOR \} \end{array} \right] \leftarrow$$

$$[, key-1=val-1, \dots, key-n=val-n]$$



Parameters:

([did],label)

Specifies a diskette volume from which the job control stream and replacement embedded data may be read. *did* specifies the device address of the diskette drive on which the diskette is mounted. *label* specifies the name of the diskette volume containing the embedded data. The diskette must have been recorded in data set label mode. The record size must be 128 bytes or less. The records must be unblocked and unspanned. *This parameter must be enclosed in parentheses.*

NOTE:

For more detailed information on the did, refer to the OS/3 system installation user guide/programmer reference.

(RDR,label)

Specifies that the replacement embedded data is located on a subfile of the input spool file. RDR specifies that the data is located on the spool file. *label* specifies the name of the subfile containing the data. *This parameter must be enclosed in parentheses.*

jobname

Specifies the name of the job to be read from either $\$Y\JCS or the alternate JCS file specified. *The jobname is required if no input device (diskette or spool file) is specified to contain the job control stream you want to run.*

(new-name)

Specifies a new name for a job already stored under the job name. The job is read from $\$Y\JCS (or alternate) under the job name and is placed on a job queue and scheduled for execution under its new name. The job name and new name may be from one to eight alphanumeric characters long. *The new-name must be enclosed in parentheses.*

:alt-filename

Specifies the name of the alternate file, residing on SYSRES, that contains the job. If the alternate file has been cataloged, the vsn identified for that file in the catalog is used. In this case, the catalog does not contain a password for the file, so you do not need to enter one with the command.

```
: (alt-filename, { RES  
                { RUN  
                { vsn
```

Specifies the name of the alternate file containing the job and identifies a volume serial number (RES, RUN, or vsn) for the file. The alternate file may reside on either a disk, or a diskette recorded in format label mode. If the alternate file resides on SYSRES, specify RES. If it resides on \$\$RUN, specify RUN. If it resides on another volume, enter the serial number of that volume. If another file with the same file name is in the catalog, the volume serial number you enter with the command distinguishes between the two files and overrides the catalog volume serial number. There must not be a password for the alternate file in the catalog. *This command must include parentheses as shown.*

```
: (alt-filename, { [ RES  
                { [ RUN  
                { [ vsn ], read-pass)
```

Specifies the name of the alternate file containing the job, a volume serial number if needed or wanted, and a password, identified in the catalog, required to read from the file. The volume serial number is specified as before. If you do not enter a volume serial number, the one listed in the catalog will be used. *This parameter must include parentheses as shown.*

```
{ PRE  
 { HIGH  
 { NOR
```

Specifies the scheduling priority of your job. PRE specifies that the job should run at *preemptive* priority, the highest priority available. Preemptive jobs, in order to obtain sufficient main storage for execution, may cause other, lower priority jobs to be rolled out (on systems with rollin/rollout configured). *You should never schedule a job with preemptive priority without first consulting the system administrator.* HIGH specifies that the job should run at *high* priority. High priority jobs will be scheduled for execution before jobs of lower priority, but will not be scheduled until any preemptive jobs are scheduled. NOR specifies that the job should be run at *normal* priority. Normal priority jobs run only when there are no jobs left in either the preemptive or high priority queues. Normal priority is used for most jobs. If you do not specify a scheduling priority, the priority indicated in the // JOB statement of the job control stream is used. If no priority was specified in the // JOB statement, normal priority is assigned by default.

```
key-1=val-1, ..., key-n=val-n
```

Specifies a series of keywords to be used by the job you wish to run. The keywords and their values must be supplied by the person creating or running the job.

NOTE:

The total length of all parameters specified in this command may be no longer than 60 characters.

Examples:

```
RU(320,INFILE) OURJOB:(JOBFILE,MYVOL1)
```

In this example, replacement embedded data for the job named OURJOB is read from the diskette mounted on device 320. INFILE is the name of the diskette file that contains the embedded data. The job itself (OURJOB) resides on an alternate file called JOBFILE, which is located on a disk volume called MYVOL1. The job runs at normal priority.

```
RU(RDR,MYFILE) MYJOB,H
```

In this example, replacement embedded data for the job named MYJOB (which resides in the \$Y\$JCS file) is read from the input spool file (specified by RDR). MYFILE is the name of the subfile (or label) located on the input spool file. The job runs at high priority.

```
RV MYJOB:(JOBFILE,REL070,SECRET)
```

In this example, the job named MYJOB is being run from the file named JOBFILE, located on the volume that has the volume serial number of RELO70. SECRET is the password required to read the file. The job runs at normal priority.

```
RV PAYJOB,H
```

In this example, the job PAYJOB is to be run at *high* priority. Since no input device or alternate file was specified, the job control stream for PAYJOB resides on \$Y\$JCS, the system job control stream file.

The following conditions may cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- File Not Found

The system cannot find the file on which the job you want is located.

SI/SC

3.5. PRERUN JOB EXECUTION

The job control procedure (jproc) calls mentioned earlier are actually series of job control statements that perform a specific function. Before a jproc can be used by the system as part of the job control stream, it must be expanded, or changed from a single statement into all the statements that single statement represents. The expansion process is one of the most time-consuming steps in the job execution process. However, using two job control option statements, you may save job control streams in the expanded state, cutting down the time needed to run those jobs in the future. The two option statements are // OPTION SAVE and // OPTION NOSCHED. The SAVE option inserted in a job control stream allows the job to be run when a RUN command is entered, and it saves the job control stream after it is expanded. The NOSCHED option only expands the job control stream and saves it, without scheduling the job for execution.

→ Expanded job control streams are saved in the \$Y\$SAVE library file or an alternate MIRAM library file. It is important to note that they are not stored in a statement-by-statement format. The job control streams are translated into code recognizable only to the RUN processor. Therefore, you cannot make changes to a saved job control stream stored in the \$Y\$SAVE file. Note also, if the main storage requirements of a saved job increase, the job must be saved again.

3.5.1. The SI/SC Commands

The SI/SC commands initiate the reading of a job control stream from the \$Y\$SAVE library file (where it has been saved in its expanded state) and then schedule the job for execution.

The SI command is used to initiate the reading of a job control stream that requires the use of an input device, either a diskette or spool file, to replace data embedded in the job control stream. When you use the SI command with a diskette as input device, be sure, prior to entering the command, that the diskette you need is mounted. The SC command is used when no input device is required.

NOTE:

You may not use the SI command to run a job requiring the use of an actual card reader; you may only use spooled readers.

Format:

SI {([did],label)} Δ [jobname] [(new-name)] [[:alt-filename
:/alt-filename, {RES
RUN }
vsn)] [[:alt-filename, {RES
RUN }
vsn], read-pass)] [{PRE
HIGH
NOR }]

SC Δ jobname [(new-name)] [[:alt-filename
:/alt-filename, {RES
RUN }
vsn)] [[:alt-filename, {RES
RUN }
vsn], read-pass)] [{PRE
HIGH
NOR }]

Parameters:

([did],label)

Specifies a diskette volume from which replacement embedded data is to be read. The did specifies the device address of the diskette drive on which the diskette is mounted. The label specifies the name of the diskette volume containing the embedded data. The diskette must have been recorded in data set label mode. The record size must be 128 bytes or less. The records must be unblocked and unspanned. *This parameter must be enclosed in parentheses.*

NOTE:

For more detailed information on the did, refer to the OS/3 system installation user guide/programmer reference.

(RDR,label)

Specifies that the replacement embedded data is located on a subfile of the input spool file. RDR specifies that the data is located on the spool file. Label specifies the name of the subfile containing the data. *This parameter must be enclosed in parentheses.*

jobname

Specifies the name of the job to be read from \$YSSAVE and scheduled for execution.

(new-name)

Specifies a new name for a job already stored under the job name. The job is read from \$Y\$SAVE or alternate MIRAM library under the job name and is placed on a job queue and scheduled for execution under its new name. The job name and new name may be from one to eight alphanumeric characters long. The new name must be enclosed in parentheses.

:alt-filename

Specifies the name of the alternate MIRAM file, residing on SYSRES, that contains the job. If the alternate file has been cataloged, the vsn identified for that file in the catalog is used. In this case, the catalog does not contain a read password for the file, so you do not need to enter one with the command.

:(alt-filename, {RES
RUN
vsn})

Specifies the name of the alternate MIRAM file containing the job and identifies a volume serial number (RES, RUN, or vsn) for the file. The alternate MIRAM file may reside either on a disk or on a diskette recorded in format label mode. If the alternate file resides on SYSRES, specify RES. If it resides on \$Y\$RUN, specify RUN. If it resides on another volume, enter the serial number of that volume. If another file with the same file name is in the catalog, the volume serial number you enter with the command distinguishes between the two files and overrides the catalog volume serial number. There must not be a read password for the alternate file in the catalog. *This command must include parentheses as shown.*

:(alt-filename, [{RES
RUN
vsn}], read-pass)

Specifies the name of the alternate MIRAM file containing the job, a volume serial number if needed or wanted, and a password, identified in the catalog, required to read from the file. The volume serial number is specified as before. If you do not enter a volume serial number, the one listed in the catalog will be used. *This parameter must include parentheses as shown.*

{PRE
HIGH
NOR}

Specifies the scheduling priority of your job. PRE specifies that the job should run at *preemptive* priority, the highest priority available. Preemptive jobs, in order to obtain sufficient main storage for execution, may cause other, lower priority jobs to be rolled out (on systems with rollin/rollout configured). *You should never schedule a job with preemptive priority without first consulting the system administrator.* HIGH specifies that jobs should run at *high* priority. High priority jobs will be scheduled for execution before jobs of lower priority, but will not be scheduled until any preemptive jobs are scheduled. NOR specifies that the job should run at *normal* priority. Normal priority jobs run

only when there are no jobs left in either the preemptive or high priority queues. Normal priority is used for most jobs. If you do not specify a scheduling priority, the priority indicated in the // JOB statement of the job control stream is used. If no priority was specified in the // JOB statement, normal priority is assigned by default.

Example:

```
SI(321,MYFILE) PAYJOB:(PAYFILE,JOBVOL)
```

In this example, embedded data from file MYFILE is read in from a diskette mounted on device 321. The saved job using the embedded data is called PAYJOB. PAYJOB is located on the MIRAM file called PAYFILE, which resides on a volume called JOBVOL. PAYJOB runs at normal priority.

```
SC MYJOB,H
```

In this example, the job named MYJOB has previously been saved in its expanded state. It is being run by entering the SC command because there is no embedded data in the job control stream to be replaced. It will be scheduled as a high priority job.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- File Not Found

The job file cannot be found.

OC/OV

3.6. RUNNING IBM SYSTEM/3 CONTROL STREAMS (OCL/OV COMMANDS)

The OCL/OV commands enable you to run an IBM System/3 job control stream in an OS/3 environment. When the OCL or OV command is entered, the entire System/3 control stream is read and interpreted by the OCL processor. The control stream is checked for syntax and sequence errors, and the job is placed in a scheduling priority queue. The OCL command is used to read job control streams that require the use of an input device, either a diskette or spool file. The OV command is used to read job control streams that do not require an input device.

NOTE:

You may not use the OC command to run jobs requiring the use of an actual card reader device; you may only use spooled readers.

Format:

$$\text{OCL} \left\{ \begin{array}{l} ([did], label) \\ (RDR, label) \end{array} \right\} \Delta [\text{jobname-library-unit}[(new-name)]] \left[\begin{array}{l} \text{PRE} \\ \text{HIGH} \\ \text{NOR} \end{array} \right]$$

$$[, \text{key-1=val-1}, \dots, \text{key-n=val-n}]$$

$$\text{OV} \Delta \text{jobname-library-unit}[(new-name)] \left[\begin{array}{l} \text{PRE} \\ \text{HIGH} \\ \text{NOR} \end{array} \right] [, \text{key-1=val-1}, \dots, \text{key-n=val-n}]$$

Parameters:

$([did], label)$

Specifies a diskette volume from which replacement embedded data is read. did specifies the device address of the diskette drive on which the diskette is mounted. label specifies the name of the diskette volume containing the embedded data. The diskette must have been recorded in data set label mode. The record size must be 128 bytes or less. The records must be unblocked and unspanned. *This parameter must be enclosed in parentheses.*

NOTE:

For more detailed information on the did, refer to the OS/3 system installation user guide/programmer reference.

$(RDR, label)$

Specifies that the replacement embedded data is located on a subfile of the input spool file. RDR specifies that the data is located on the input spool file. label specifies the name of the subfile containing the data. *This parameter must be enclosed in parentheses.*

jobname-library-unit

Identifies the name of the job to be read from the library specified by the *library-unit* code. Possible library-unit codes are F1, F2, R1, and R2. The job name and library unit must be separated by a dash (-).

(new-name)

Assigns a new name to an OCL job. The job is stored in a scheduling priority queue under the new name to await execution. If you do not specify a job name with the command, the job name will be taken from the // JOB statement in the job control stream, or will be OCLnnnn if no job name can be found in the job control stream. *The new-name must be enclosed in parentheses.*

{ PRE
HIGH
NOR }

Specifies the scheduling priority of your job. PRE specifies that the job should run at *preemptive* priority, the highest priority available. Preemptive jobs, in order to obtain sufficient main storage for execution, may cause other, lower priority jobs to be rolled out (on systems with rollin/rollout configured). *You should never schedule a job with preemptive priority without first consulting the system administrator.* HIGH specifies that the job should run at *high* priority. High priority jobs will be scheduled for execution before jobs of lower priority, but will not be scheduled until any preemptive jobs are scheduled. NOR specifies that the job should run at *normal* priority. Normal priority jobs run only when there are no jobs left in either the preemptive or high priority queues. Normal priority is used for most jobs. If you do not specify a scheduling priority, the priority indicated in the // JOB statement of the job control stream is used. If no priority was specified in the // JOB statement, normal priority is assigned by default.

key-1=val-1,...,key-n=val-n

Specifies a series of keywords to be used by the job you wish to run. The keywords and their values must be supplied by the person creating or running the job.

Example:

OV OLDPAY

In this example, a job with the name OLDPAY is to run. The job requires no replacement of embedded data, so no did,label or RDR,label parameters are included. The job is to run at normal priority.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.



3.7. THE EXECUTE FACILITY

EXECUTE is an interactive facility designed to save you response time if you run more than one workstation-oriented job in a single workstation session.

Whenever you initiate a job, it does not begin executing immediately. First, the run processor scans your control stream to translate job control statements and to expand jprocs. Then it checks for order and syntax errors. If your control stream is error free, the run processor builds a table of control blocks that describe your job requirements. Next, it creates a file for your job on \$Y\$RUN.

When the run processor is finished, the job scheduler function is activated. It reserves devices and main storage for your job as they become available. It then selects your job for execution according to its priority when the devices and main storage needed to run it are available. Each time you run a job, the run processor and job scheduling steps are repeated.

The EXECUTE facility, however, allows you to run a set of programs without going through the run processor and job scheduling steps for each program. It does this by enabling you to define a single job control stream for a set of programs of your choosing. This job control stream is called a super-set job control stream. When you run the super-set job control stream, it creates an interactive job environment. All a workstation operator has to do is select a program from the super set and issue an EXECUTE command. Since the job environment has already been established, the program is loaded and executed without any wait for job control processing.

3.7.1. Choosing Jobs to Run Interactively

Before building a super-set job control stream, you must select the programs you will execute in an interactive job environment. When choosing programs, consider these points:

- Choose programs that use the workstation. Examples are programs that use screen format services or menu services.
- *Do not choose programs that call for embedded data (// PARAM cards or /\$.../* sequences). They will not work. EXECUTE is designed only for programs that interactively ask you for information.*
- Do not choose programs that are run in batch mode.



3.7.2. Building the Super-Set Job Control Stream

The super-set job control stream is based on the job control streams and resource requirements for each program you've selected for your super-set. The resources you must take into consideration are:

- Main storage
- Peripheral assignments
- Workstation assignments

The following steps outline the basic job control requirements needed to build a super-set job control stream.

Step 1:

The first card you build is the jobname card:

```
// JOB $$&USER$,,main storage requirement
```

The name `$$&USER$` is the jobname the EXECUTE facility expects to find. You can specify your own jobname but when you run your super-set job control stream, you must rename it to `$$&USER$`. (The RV and SC commands have a new-name parameter where you would specify `$$&USER$`.) It is easier to keep `$$&USER$` as the jobname and then, if necessary, write the super-set control stream to a file under another name.

The jobname card must also allocate enough main storage to run the largest program in your super-set. (You may need to consult a link map for the size of your user programs.)

You can insert any `// OPTION` cards you wish to include after the `// JOB` card.

Step 2:

A single workstation file definition is required. It must be of the form:

```
// DVC 200 // LFD lfd-name
```

You may define only one workstation file. Therefore, a single lfd name must be agreed upon for all programs in your super-set job control stream. This may mean recompiling your programs to use the agreed-upon lfd-name. A `// UID` must not be specified. Multiple lfd's for this file are also not permitted. A `// LBL` statement is allowed, but serves no purpose in this context.

Step 3:

You must allocate peripherals (printer, punch, diskette, disk, and tape devices). For easier construction, you should group all file definitions for like volume names together. DVC numbers may need to be resolved so that they reflect the correct volume names; for example, you should construct your statements so that DVC 50 always refers to volume X, DVC 51 refers to volume Y, etc.

The DVCVOL jproc should be used with care. If you run the super-set job control stream using the RV command, then any jproc processing could add to initiation time. On the other hand, if you run your super-set job control stream using the SC command, then jprocs have no adverse effect on initiation time.

If one or more programs use a different logical name for the same file, the file can be assigned more than one LFD name. For example:

```
// DVC 50 // VOL INVTRY
// LBL INVENTORY // LFD INVILE
// DVC 50 // VOL INVTRY
// LBL INVENTORY // LFD INVENTY
```

NOTE:

Devices allocated to a super-set job remain allocated for the duration of the user session.

Step 4:

All programs in your super-set must reside on the same library file. Thus, you may need to copy your programs to a common library. You must define this library in your super-set job control stream. In addition, the lfd-name for this file must be placed on the // EXEC statement for the super-set job control stream. If your common library is the \$Y\$LOD library, it isn't necessary to define the file.

The // EXEC statement executes a system-supplied program that actually creates the interactive job environment. The statement is:

```
// EXEC $$INT,lfd-name
```

where the lfd-name is the library where your programs reside. The super-set job control stream does not reference any job but \$\$INT, which resides on \$Y\$LOD. Thus, no special handling is required for \$\$INT.

You end the super-set job control stream with the following statements:

```
/&
// FIN (for card input only)
```

Step 5:

You must file the super-set job control stream on disk. You can do this with the FILE command or EDT. Once it's on disk, you can run it with the RV command. For additional speed, you can save the job by including a // OPTION SAVE,NOSCHED statement in your super-set control stream. Once a job is saved, it can be run with the SC command.

3.7.3. Using the EXECUTE Command

To begin an interactive job session, you must run the super-set job control stream. Using the RV command, you enter:

```
RV jobname
```

where jobname is the name of the module in which your super-set job control stream resides. If your super-set job control stream was saved in the expanded form (via the // OPTION SAVE/NOSCHED statement), you can run it with the SC command:

```
SC jobname
```

If you used any name other than \$\$&USER\$ on the // JOB statement, you must rename the job on the RV or SC command:

```
RV jobname($$&USER$)
SC jobname($$&USER$)
```

(Note that you can include the RUN statement for your super-set job control stream in a user's execution profile. This way, when the user logs on, the super-set job control stream is automatically initiated.)

```
EXECUTE program
```

Once your super-set job control stream is processed, the job (or interactive job environment) that is created has the name

```
$$userid
```

in the system, where userid is the user-id under which you logged on. This enables a single super-set job control stream to be used simultaneously at different workstations. Each time the super-set job control stream is run, a new interactive job environment is established with a different user-id.

With the interactive job environment ready, you can issue requests to execute programs:

```
EXECUTE program
```

where program is the 1- to 6-character name from the // EXEC statement of one of the user programs in your super set. The program is loaded into the job region and is given control. When the program terminates (either normally or abnormally), you receive the message:

```
IS126 EXECUTE COMMAND TERMINATED
```

You may issue only one EXECUTE command at a time. That is, you must wait for one program to finish before you can initiate another one. You can, however, issue as many EXECUTE commands in the course of a single session as you'd like. Dumps requested via the // OPTION statement are produced as usual, if the program terminates abnormally. Any spooled output is also printed when the EXECUTE command has terminated.

When an EXECUTE command terminates, the job region remains available and you can issue another EXECUTE command. You may also use other workstation commands, such as COPY, EDT, RUN, or CONNECT, between EXECUTE commands.

To end the interactive job environment, you simply logoff. If you want to use the CANCEL command, you must remember that, when an EXECUTE command is active (a program is running), you must first cancel the program that is running and then cancel the interactive job environment. If you use the CANCEL command when an EXECUTE command is not active, the complete job environment is cancelled. When cancelling the job environment, remember that the jobname is always \$\$userid, not \$\$INT.

3.7.4. Creating and Running a Sample Super-Set Job Control Stream

Suppose there are three programs that you consistently run in a single workstation session. These programs are an RPG II program that processes factory shipments, another RPG II program that updates the factory's vendor information, and a COBOL program that controls invoicing. In order to achieve faster response times, you want to create an interactive job environment for these three programs by using the EXECUTE facility. Figure 3-1 lists the job control streams for all three programs.

| RPG II SHPMNT PROCESSING PROGRAM | RPG II VENDOR UPDATE PROCESSING PROGRAM | COBOL INVOICING PROGRAM |
|-------------------------------------|--|---------------------------------|
| // JOB SHIPMENT,,4000 | // JOB VNDRUPDT,,A000 | // JOB INVOICE,,7800 |
| // DVC 20 // LFD PRNTR | // DVC 20 // LFD PRNTR | // DVC 20 // LFD PRNTR |
| // DVC 200 // LFD WKSTN | // DVC 200 // LFD WKSTN | // DVC 200 // LFD WORKSTAT |
| // DVC 50 // VOL INVTRY | // DVC 50 // VOL INVNTY | // DVC 50 // VOL INVTRY |
| // LBL INVENTORY // LFD INV | // LBL INVENTORY // LFD INV | // LBL INVENTORY // LFD INVFILE |
| // DVC 50 // VOL INVTRY | // DVC 50 // VOL INVNTY | // DVC 51 // VOL ACCTNG |
| // LBL BACKORDER // LFD BKORD | // LBL VENDORS // LFD VEND | // LBL BILLS // LFD CUSTBILL |
| // DVC 51 // VOL PROGRAM | // DVC 51 // VOL PROGRAM | // DVC 52 // VOL PROGRAM |
| // LBL LOADLIB // LFD LOAD | // LBL LOADLIB // LFD LOAD | // LBL LOADLIB // LFD LOAD |
| // EXEC SHPMNT,LOAD | // EXEC VENDOR,LOAD | // EXEC INVOIC,LOAD |
| /& | /& | /& |
| // FIN | // FIN | // FIN |

Figure 3-1. Sample Programs

On the // JOB statement, you use the jobname \$\$&USERS\$ and you evaluate the main storage requirements of the programs in your super set. The largest of the three programs is the program VENDOR. Its main storage requirement is A000. Thus, the finished // JOB statement is:

```
// JOB $$&USERS$,,A000
```

In addition, you want a job dump in case an error arises when you run the program. You include a // OPTION statement following the // JOB statement:

```
// OPTION JOBDUMP
```

Next, a workstation file assignment must be developed. Programs SHPMNT and VENDOR use an lfd-name of WKSTN, while INVOIC uses WORKSTAT. In order to include INVOIC in the super set, the program must be recompiled with the common lfd-name WKSTN. Once this is done, you can use the super-set workstation device assignment:

```
// DVC 200 // LFD WKSTN
```

Now you must make device assignments. You define a printer with a common lfd-name and make a series of disk assignments. You decide that DVC 50 refers to INVNTY, DVC 51 to ACCTNG, and DVC 52 to PROGRAM. Since duplicate definitions with the same lfd-names are permitted for all files except the workstation, you can construct the device assignment sets as:

```
// DVC 20 // LFD PRNTR  
// DVC 50 // VOL INVTRY  
// LBL INVENTORY // LFD INVFILE  
// DVC 50 // VOL INVTRY  
// LBL INVENTORY // LFD INV  
// DVC 50 // INVTRY  
// LBL BACKORDER // LFD BKORD  
// DVC 50 // VOL INVTRY  
// LBL VENDORS // LFD VEND  
// DVC 51 // VOL ACCTNG  
// LBL BILLS // LFD CUSTBILL  
// DVC 52 // VOL PROGRAM  
// LBL LOADLIB // LFD LOAD
```

Since all three programs already reside in the same library, it isn't necessary to copy them to a common library. The last three statements needed to complete the super-set job control stream are:

```
// EXEC $$INT,LOAD  
/&  
// FIN
```

Figure 3-2 shows you the completed super-set job control stream.

Once constructed, you now write the job control stream either to the \$\$JCS library file or to an alternate library file. You can perform the write function through EDT or the FILE command. For this example, the super-set control stream is written to a module called BILLING1 on the file MYFILE, which resides on disk volume MYVOL1.

```
// JOB $$&USERS$, ,A000
// OPTION JOBDUMP
// DVC 20 // LFD PRNTR
// DVC 50 // VOL INVTRY
// LBL INVENTORY // LFD INVFILE
// DVC 50 // VOL INVTRY
// LBL INVENTORY // LFD INV
// DVC 50 // VOL INVTRY
// LBL BACKORDER // LFD BKORD
// DVC 50 // VOL INVTRY
// LBL VENDORS // LFD VEND
// DVC 51 // VOL ACCTNG
// LBL BILLS // LFD CUSTBILL
// DVC 52 // VOL PROGRM
// LBL LOADLIB // LFD LOAD
// DVC 200 // LFD WKSTN
// EXEC $$INT,LOAD
/&
// FIN
```

Figure 3-2. Completed Super-Set Job Control Stream

Later, when you are ready to run the super-set job control stream, you log on:

```
LOGON ANDY
```

You then enter the run command:

```
RV BILLING1:(MYFILE,MYVOL1)
```

When the system has processed your super-set job, the interactive job environment is created under the name:

```
$$ANDY
```

where ANDY is your user-id. At this point, you can enter an EXECUTE command to run one of the programs in your super set:

```
EX VENDOR
```

When program VENDOR terminates, the message

```
IS126 EXECUTE COMMAND TERMINATED
```

is displayed on your workstation screen. You can now enter another EXECUTE command:

```
EX SHPMNT
```

When you've run the necessary programs from your super set, you end the interactive job environment by simply logging off:

```
LOGOFF
```

If you're not ready to log off, however, you can cancel the interactive job environment by issuing the CANCEL command:

```
CANCEL $$ANDY
```

Note that you can cancel \$\$ANDY only if an EXECUTE command is not active. If an EXECUTE command is active, you must first cancel the program that is running and then cancel the job environment. For example:

```
First command: CA VENDOR,N
```

```
Second command: CA $$ANDY
```

3.7.5. Error Messages

The following are error messages that could be displayed at your workstation in the event of a problem:

- IS124 NO LOAD MODULE NAME SPECIFIED

An EXECUTE command was entered without a program name. A program name must follow the command verb, separated by a space, and is one to six characters in length.

- IS78 EXECUTE COMMAND STILL ACTIVE, WAIT FOR IT TO FINISH

You must wait for one EXECUTE command to finish before issuing another.

- IS127 LOAD MODULE NAME TOO LONG

The program name specified on the command is longer than six characters. Consult the link map for the correct program name.

- IS128 EXECUTE COMMAND NOT ALLOWED NOW, W/S STILL IN USE

You cannot execute a program while the workstation is being used by another job or interactive utility. The workstation may have been connected (via the CONNECT command) to a job, or a command such as EDT or BASIC may be active.

↓

- IS129 EXECUTE COMMAND ENCOUNTERED ERROR CODE xx

A system error was encountered while trying to load your program. See the system messages manual for a full explanation. The most common error codes are 51 (program not found or spelled incorrectly) and 5B (insufficient main storage to run program).

- IS31 INVALID OPTION FOR EXECUTE COMMAND

The only parameter permitted on the EXECUTE command is the program name. Additional parameters entered after the program name cause this error.

If you try to use the EXECUTE command before the job environment has been established, an automatic DISPLAY JS command is issued by the system. This command tells you why the job environment is not ready. For example, the super-set job could still be under control of the run processor or enough main storage unavailable yet for it to run. Remember, the super-set job control stream is constrained by the same resource limitations as any other job; for example, main storage, job slots, and cpu utilization.

↑

DLOAD

3.8. Downline Loading Programs (DLOAD)

The DLOAD command allows a UTS 400 terminal user or a UTS 40/40D workstation user to downline load a program to terminal storage (memory). It is specifically designed for users of the PL/M and COBOL compilers and the MAC80 assembler. Any program you load with the DLOAD command must reside in the \$Y\$LOD library.

Downline loading is the process by which you load a program designed to execute on a UTS 400 terminal or on a UTS 40/40D workstation "down a communications line" to a terminal from a host processor. Aside from using the DLOAD command, this can also be done using an ICAM communications user program (CUP). You may also load a program offline, or manually, from an auxiliary device. If you're manually loading from an auxiliary device, you must have previously downline loaded your program from your host processor to your auxiliary device, such as a diskette. The UTS 400/UTS 4000 – OS/3 Interface user guide/programmer reference describes both downline loading through ICAM and offline loading from an auxiliary device.

Format:

```
DLOAD△(program-name)
      {
      /END
      /OFFLINE
      }
```

where:

program-name

Specifies the program, residing in \$Y\$LOD, that you want to downline load. This form of the DLOAD command allocates all of the auxiliary devices your system needs to downline load the program and then loads it. During the time your system loads and executes your program, DLOAD has sole control of those auxiliary devices, and you cannot run any other program that uses them. Also, you cannot successfully issue ASK or TELL commands during the time your system loads and executes your program. If issued, these commands are sent to the system console. (See 4.2.1 and 4.2.10 for information about the ASK and TELL commands, respectively.) As the loading takes place the message LOAD IN PROGRESS is displayed. After your program terminates, you must enter the DLOAD△/END command to free the devices for other uses.

/END

Frees all auxiliary devices that your system allocated for loading and executing your program. You must enter DLOAD△/END after the program You're downline loading or offline loading has terminated. Otherwise, your system won't be able to allocate those auxiliary devices to any other programs.

/OFFLINE

Allocates all of the auxiliary devices needed for offline, or manually, loading a program from an auxiliary device. Therefore, you must enter **DLOAD△/OFFLINE** before you offline load the program. Once you receive the message, **READY TO LOAD, AUXILIARY DEVICES ARE ALLOCATED**, you can manually load the program. See the **UTS 400/UTS 4000 - OS/3 Interface user guide/programmer reference** for information on offline loading programs. Once your program terminates, remember to enter the **DLOAD△/END** command to free the auxiliary devices for other uses.

Example:

```
DLOAD MYPROG1
```

In this example, you are downline loading the program **MYPROG1** to terminal storage (memory).

The following conditions could cause this command to be rejected:

■ Wrong Device

The **DLOAD** command can be initiated only from a **UTS 400** terminal or a **UTS 40/40D** workstation.

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command was entered improperly.

The following are error messages associated with the **DLOAD** command:

■ Errors unique to DLOAD signifying an internal problem

```
IS107 MEMORY START ADDRESSES NOT EQUAL IN DOWN-LINE LOAD  
IS109 ILLEGAL CONTROL CODE FOUND IN DOWN-LINE LOAD
```

■ Other errors unique to DLOAD

```
IS108 MEMORY ADDRESS OUT OF RANGE FOR DOWN-LINE LOAD  
IS110 DOWN-LINE LOAD NOT DIRECTED TO A MASTER WORKSTATION  
IS111 TERMINAL REJECTED DOWN-LINE LOAD, MAY NOT BE PROGRAMMABLE  
IS133 AUXILIARY DEVICES ARE NOT AVAILABLE
```



ULD

3.9. The Upline Dump Command for UTS 400 Terminal Users (ULD)

The upline dump command allows a UTS 400 user to obtain a dump of the terminal storage (memory). ULD reads from the user programmable region of the UTS 400 storage and writes the contents to a user-designated disk file.

There are two command options associated with the ULD command. You can specify that the contents of your dump file be printed and/or saved. If you specify the print option, ULD automatically schedules the job UPLDUMP for execution after the contents of terminal storage (memory) have been written to your dump file. UPLDUMP formats and prints the file contents. Once printing is completed, the file contents are scratched unless you specified the save option.

If you do not specify the print option, the contents of your dump file are automatically saved. You cannot (and would not want to) specify both the noprint and the scratch options. If you try to, the system will ignore your specification and save the file contents.

Format:

```
ULDΔ, filename, vsn, SIZE=nΔ 

|         |         |
|---------|---------|
| PRINT   | SCRATCH |
| NOPRINT | SAVE    |


```

Command Parameters:

filename

The name of the file to which you want the contents of the dump written. The filename doesn't have to be the name of an existing file. You can allocate file space through the ULD command. A filename must be unique and can be up to 44 alphanumeric characters long. It must also have a comma before and after it. You must enclose the filename in quotation marks or apostrophes if there are any spaces, commas, or parentheses embedded in it. If you use an already existing file, it must be a MIRAM file.

vsn

The vsn is the volume serial number of the disk where your dump file will reside.

SIZE=nn

You must specify the number of disk cylinders you will need for your dump file. If the file you intend to use already exists, this parameter is not needed.



Command Options:

{ PRINT }
{ NOPRINT }

If you accept PRINT (the default), ULD schedules job UPLDUMP to format and print your dump file. When printing is completed, UPLDUMP scratches (erases) your dump file unless you specify the SAVE option. If you specify the NOPRINT option, UPLDUMP is not scheduled and the contents of your dump file are automatically saved.

{ SCRATCH }
{ SAVE }

You can specify whether or not you want your dump file saved or erased after it has been printed. SCRATCH is the default only if the PRINT option was selected. If you don't specify the PRINT option, ULD automatically saves your dump file and you cannot scratch it.

Examples:

```
ULD ,MYFILE,RES,SIZE=2 SAVE
```

In this example, you are writing the contents of your UTS 400 terminal storage (memory) to the file called MYFILE. MYFILE is on the volume RES and occupies two cylinders of file space. The dump is printed and the file contents are saved.

```
ULD ,DUMPFIL,REL080,SIZE=2
```

In this example, the dump is written to DUMPFIL, which resides on the REL080 volume. Two cylinders of file space were allocated for DUMPFIL. The contents of DUMPFIL are printed and then scratched since the SAVE option was not specified.

```
ULD ,THEFILE,MYVOL01,SIZE=3 NOPRINT
```

In this example, THEFILE is not printed. The contents, however, are saved even though the SAVE option was not specified.

The following conditions could cause this command to be rejected:

■ Wrong Device

The ULD command can be initiated only from a UTS 400 terminal.

■ Wrong File Type

A UTS 400 terminal dump can only be written to a MIRAM file.

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command was entered improperly.

BE**3.10. CHANGING JOB SCHEDULING**

After your job control stream is read and, if necessary, expanded, it is placed on a priority scheduling queue. It then waits until all the system resources it needs become available. When the resources all become available, the job is executed. You may alter the scheduling of your jobs as they wait for system resources on the scheduling queues. The following commands permit you to defer and reinstate the scheduling of a job, delete a job from a scheduling queue, change the scheduling priority of a job, and display a listing of the jobs on each scheduling queue. You may alter only the scheduling of jobs initiated under the user-id you entered when you logged onto the system.

3.10.1. Rescheduling Jobs (BEGIN Command)

The BEGIN command enables you to allow the scheduling for execution of jobs deferred by the HOLD command. You may reschedule individual jobs initiated under your user-id, all jobs in a particular queue initiated under your user-id, or all jobs initiated under your user-id.

NOTE:

The BEGIN JBQ command only reschedules those jobs (initiated under your user-id) which are being held at the time you enter the command. When the system operator (working at the console) issues a HOLD JBQ command, not only are all jobs currently on the scheduling queue(s) specified held, but any jobs which are placed on the queue(s) after the HOLD JBQ command is issued are also held. When you issue a BEGIN JBQ command, you reschedule only those of your jobs being held when you issue the command.

Format for Rescheduling All Jobs or Jobs in a Particular Job Queue:

```
BEGIN△JBQ [ ( H N P ) ]
```

Format for Rescheduling Individual Jobs:

```
BEGIN△jobname
```


Parameters:

A/H/N/P

Specifies the scheduling priority queue you wish to reschedule. Entering P reschedules all jobs initiated under your user-id in the preemptive priority job queue. H reschedules all jobs initiated under your user-id in the high priority job queue. N reschedules all jobs initiated under your user-id in the normal priority job queue. A reschedules all jobs initiated under your user-id in all scheduling priority job queues.

jobname

Specifies the name of the job you wish to reschedule for execution.

Example:


```
BE JBQ,N
```

In this example, all jobs initiated under your user-id in the normal scheduling priority job queue, which has been placed on hold, are now rescheduled for execution.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.



HO

3.10.2. Deferring Jobs (HOLD Command)

The HOLD command enables you to defer the scheduling of jobs initiated under your user-id. You may hold individual jobs, all jobs on a job queue (initiated under your user-id), or all jobs on the system (initiated under your user-id). The job you hold is not scheduled until you enter a BEGIN command to remove the job from hold status. (To see what jobs are already on HOLD, use the DISPLAY JBQ command.)

NOTE:

The HOLD JBQ command only holds those jobs which are on a scheduling queue at the time the HOLD command is issued. Any jobs placed on a scheduling queue after the HOLD command is issued are not held.

Format for Holding All Jobs or All Jobs on a Particular Job Queue:

`HOLDΔJBQ` $\left[\begin{array}{c} \text{A} \\ \text{H} \\ \text{N} \\ \text{P} \end{array} \right]$

Format for Holding an Individual Job:

`HOLDΔjobname`

Parameters:

$\left[\begin{array}{c} \text{A} \\ \text{H} \\ \text{N} \\ \text{P} \end{array} \right]$

Specifies which job queue you wish to place on hold status. Entering P holds all jobs initiated under your user-id in the preemptive priority job queue. H holds all jobs initiated under your user-id in the high priority job queue. N holds all jobs initiated under your user-id in the normal priority job queue. A holds all jobs initiated under your user-id in all job queues.

Example:

`HO JBQ,N`

In this example, the jobs initiated under your user-id on the normal priority job queue are held from execution.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

DI JBQ**3.10.3. Displaying the Contents of a Job Queue (DISPLAY JBQ Command)**

The DISPLAY JBQ command enables you to display the contents of each scheduling priority job queue on the workstation screen. The display shows all the jobs residing in the queue, or queues you specified, by their job names. The job names of jobs deferred from scheduling by the HOLD command are displayed enclosed in parentheses.

Format:

```
DI DISPLAY△JBQ, { A
                  H
                  N
                  P }
```

Parameters:

A/H/N/P

Specifies the scheduling priority job queue you wish displayed. Entering P displays the contents of the preemptive priority job queue, H displays the contents of the high priority job queue, and N displays the contents of the normal priority job queue. Entering A displays the contents all three job queues: preemptive first, then high, and then normal.

Example:

```
DI JBQ,P
```

In this example, the contents of the preemptive priority job queue will be displayed.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

DE**3.10.4. Deleting Jobs from a Job Queue (DELETE Command)**

The DELETE command enables you to remove jobs initiated under your user-id from scheduling job queues. You may delete individual jobs, or all jobs initiated under your user-id in a particular queue, or all jobs initiated under your user-id in all queues.

Format for Deleting an Individual Job:

```
DELETEΔ jobname [,LOG]
```

Format for Deleting All Jobs on a Queue or All Jobs in the System:

```
DELETEΔJBQ, (A) [,LOG]
              {
              H
              N
              P
              }
```

Parameters:

jobname

Is the 1- to 8-character name of the job you wish to delete.

A/H/N/P

Specifies the job queue from which you wish to delete all jobs initiated under your user-id. Entering P causes all such jobs to be deleted from the preemptive job priority queue; H causes all such jobs to be deleted from the high priority queue; N causes all such jobs to be deleted from the normal priority queue; and A causes all such jobs to be deleted from all queues.

LOG

Causes the job log to be printed.

Example:

```
DE JBQ,N
```

In this example, all jobs initiated under the user-id of the user entering the commands that are on the normal priority job queue are deleted.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

CH**3.10.5. Changing the Scheduling Queue of a Job (CHANGE Command)**

The CHANGE command enables you to move a job from one job queue to another, thus changing the scheduling priority of the job. If the job was in HOLD status on its original job queue, it will retain that HOLD status on its new job queue. *You may change only the jobs that were initiated under your user-id.*

Format:

`CHANGEΔjobname, { H }
 { N }
 { P }`

Parameters:

jobname

Is the name of the job whose job queue you wish to change.

H/N/P

Specifies in which job queue you wish to place the job. H specifies the high priority queue, N specifies the normal priority queue, and P specifies the preemptive priority queue. *Do not move jobs from the normal priority queue to the high or preemptive queues, or from the high to the preemptive queue without first consulting your site administrator.*

Example:

`CH MYJOB,N`

In this example, the job MYJOB is changed from either a high priority or a preemptive priority to normal priority.

CON

3.11. CONNECTING A WORKSTATION TO A JOB

In order to use workstations with a job, the workstations must be *connected* to the job. The workstations are actually connected to one or more files, created by the job to interface with workstations. The WORKSTATION mode of the workstations is connected to the files. There are several ways to connect workstations to the files. One method is through job control. When you create the job control stream for a job using workstations, certain options of the UID job control statement can be entered to cause workstations to be automatically connected to a job. Complete information on using job control to connect workstations may be found in the OS/3 job control user guide.

If you want to connect to a job that has no automatic connection provision for your particular workstation, use the CONNECT workstation command, described in the following pages. System programs and interactive facilities automatically connect the workstation from which they were invoked.

NOTE:

A job using workstation files cannot be restarted from a checkpoint taken when the workstation files are open.

3.11.1. The CONNECT Command

The CONNECT command enables you to connect WORKSTATION mode to a job. You may connect only to jobs that have been written to recognize workstations. You may issue the CONNECT command when the job is executing or on a job queue.

Format:

```
CONNECT△job[,filename]
```

Parameters:

job

Specifies the name of the job to which you wish to connect the WORKSTATION mode of your workstation.

filename

Specifies the name of the workstation file to which you wish to connect. This parameter is required when there is more than one workstation file defined within the job. The file name may be up to 17 alphanumeric characters long.

Example:

```
CON COLLECTN,BILLS
```

In this example, the workstation is connected to the workstation file `BILLS` of the job `COLLECTN`.

The following conditions could cause this comand to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were entered improperly.

- **Attempt to Connect to a Job That Cannot Handle Any More Workstations**

The number of workstations that may be connected to a job is set by a statement in the job control stream used to run the job. No more than the number specified will be accepted.

- **Attempt to Connect to a Job That Does Not Expect Workstations**

You may connect only to a job that has been specifically written to use workstations.

- **Attempt to Connect to a Nonexistent File or Job**

You may connect only to jobs either executing or waiting for execution on a job queue. You may connect only to workstation files defined within the program.

- **Implied Disconnect Cannot Be Accomplished**

For some reason, the system is unable to disconnect your workstation from the job it is currently connected to. Cannot accomplish the `CONNECT` command.

- **Attempt to Connect to an Inactive Job**

You may connect only to a job that is active, i.e., executing on the system or on a job queue.

3.12. THE MASTER WORKSTATION

3.12.1. General

When a job is initiated at a workstation, that workstation normally has control of that job. The workstation functions as a minisystem console for the job. The workstation initiating the job has control regardless of the number of workstations subsequently connected to it. The controlling workstation is designated the *master workstation* for the job. Messages from the system concerning the job are routed to the master workstation, and responses to those messages and commands entered to control the job may be issued only from the master workstation. This is particularly important for commands. Many commands in this manual state that you may perform the command only on jobs initiated or running under your user-id. Since the user-id designates the master workstation, this means that only the master workstation of a job may perform commands on that job. As we shall see subsequently, the master workstation may be changed. However, whichever workstation is designated the master workstation is the *only* workstation controlling a particular job.

As we said earlier, the master workstation status of a workstation is maintained by the user-id entered when the master workstation was logged on. The user-id is saved by the system; it is appended to all commands entered at the workstation and all jobs for which that workstation is designated as the master workstation. If your system has DDP, you can also designate the host-id of a particular host that you want to control a job.

When a command is entered against a particular job, the system matches the user-id of the command against that of the job the command is directed to. If the user-ids match, the command is executed. If they do not match, the command is rejected. Figure 3-13 illustrates the manner in which the user-id controls program access. A command issued by the workstation logged on under user-id `USERA` cannot affect `MYJOB`, because `MYJOB` is under the control of a master workstation having the user-id `USERB`.

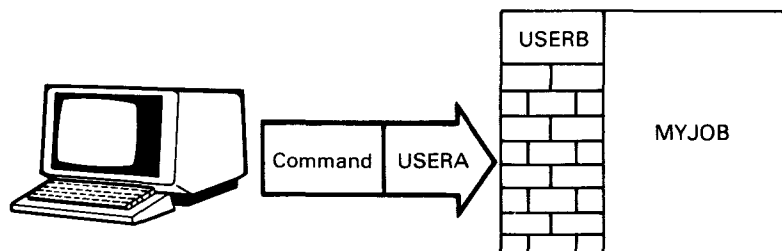


Figure 3-3. How the User-id Controls Access to Jobs

3.12.2. Changing the Master Workstation for a Job

You may designate a workstation other than the initiating workstation as the master workstation for a job. There are two methods you can use. The first is to log off from the workstation designated as master workstation for a particular job, and log on to another workstation. If you log on under the same user-id you used to log on to the initiating workstation, the workstation you log on to will become the master workstation.

The second method of designating a master workstation uses two parameters of the // OPTION job control statement.

`ORI=[host-id:]user-id`

Designates whatever workstation is logged on under the specified user-id as the master workstation. If your system has DDP, you can use the host-id to specify a particular host. If you omit the host-id, the local host (the processor on which the job is executing) is assumed. The host-id is optional but, if specified, must be followed by a user-id. The ORI option takes effect as soon as it is detected in the control stream, before the job is placed in the job queue. The workstation from which the command to start the job was initiated is no longer designated the master workstation for the job. The user-id may be from one to six alphanumeric characters long.

NOTE:

ORI=user-id cannot be specified in a job initiated from a remote batch terminal.

`MAS=[host-id:]user-id`

Designates whatever workstation is logged on under the specified user-ID as the master workstation. The MAS option takes effect when the job is placed in the job queue. If your system has DDP, you can use the host-id to specify a particular host. If you omit the host-id, the local host (the processor on which the job is executing) is assumed. The host-id is optional but, if specified, must be followed by a user-id.

NOTE:

MAS=user-id cannot be specified in a job initiated from a remote batch terminal.

You may use ORI and MAS together, designating one workstation as the master while the job is being acted upon the RUN processor, and another when the job goes onto a scheduling priority queue. By entering (EXEC) immediately after the user-id with MAS, the user-id specified will not change the master workstation designation until the job begins executing. Thus, the workstation specified by ORI controls the job while it is on a scheduling priority queue, and relinquishes control to the workstation specified by MAS only when the job begins execution.

NOTE:

For more detailed information on the use of these job control statements, refer to the OS/3 job control user guide.

If you initiate a job from a workstation, then log off that workstation and do not designate another workstation to take over, messages concerning the job will be routed to the system console. The system console may always control all jobs running on the system.



4. General Workstation Commands

4.1. INTRODUCTION

This section covers those workstation commands not already covered in Sections 2 or 3. The commands covered in this section are divided into three parts:

1. Commands to control the job processing environment
2. Commands to control output spooling
3. Interactive utility commands

4.2. COMMANDS TO CONTROL THE JOB PROCESSING ENVIRONMENT

This part covers those commands you may use to control the environment in which your job is running. These commands enable you to communicate with the system operator; to obtain information on the status of your job as it is scheduled and while it is being executed; and to stop, restart, and terminate your job.

AS

4.2.1. Asking Questions of Other Workstation Users (ASK Command)

The ASK command enables you to ask questions of other users or the system operator. The command displays your question to the other user, accepts the reply, and returns the reply to you.

Format:

```
ASKΔ[user-id,]'text'
```

Parameters:

user-ID

Specifies the user-id of the user you wish to question. If you do not specify a user-id, the question will be routed to the console operator. The console, however, has a user-id, which you may use in communicating with the console operator. It is \$Y\$CON. If you wish to ask a question of a user who has initiated an ENTER file, enter \$Y\$MAS as the user-id.

'text'

Is the text of your question to the operator. The text may be a maximum of 48 characters long. A longer text causes the command to be rejected. A comma must separate the text from the user-id. If there is no user-id, you do not need to enter a comma. *The text must be enclosed in apostrophes, and therefore, can't contain apostrophes itself. So, be sure not to use contractions in your text. For example, enter 'WHERE IS DOABCD?' instead of 'WHERE'S DOABCD?'*

Example:

A user with userid P15801 keys in the following question and sends it to a user with userid P15802 (in system mode):

```
ASK 'IS DISK ''RELPAK'' AVAILABLE NOW?'
```

User P15802 sees this question on his workstation screen in the form:

```
OF? IS67 P15802: IS DISK ''RELPAK'' AVAILABLE NOW? OF ANSWER
```

User P15802 responds:

```
IS67 OF YES
```

User P15801 then sees the following message on his screen:

```
IS67 P15801: YES
```

NOTE:

All responses are entered in system mode.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, the user-id was longer than six characters, or apostrophes were used incorrectly.

- Message Too Long

The message text is longer than 48 characters. The command is rejected.

NOTE:

Only one ASK command may be processed at a time. You may not enter another ASK until the command currently being processed is completed.



CA

4.2.2. Canceling a Job (CANCEL Command)

The CANCEL command enables you to immediately halt processing of a job. When you enter the CANCEL command, the job is brought to an immediate halt; the job step currently executing is not completed, and any remaining job steps are not executed. CANCEL may be entered any time during job processing. *You may perform a CANCEL only of jobs executing under your user-id.*

Format:

`CANCEL△jobname [, { D }]`

Parameters:

jobname

Specifies the name of the job you wish to cancel.

D/N

Specifies whether or not you want a dump to be taken when the job is terminated, regardless of the dump option specified in the job control stream for the job. D specifies that you want a dump taken; N specifies that you do not want a dump taken. If you omit this parameter, the dump options contained in the job control stream for the job will remain in effect.

Example:

```
CA MYJOB,D
```

In this example, the job named MYJOB is canceled. A dump is taken, regardless of the dump option specified in the job control stream for the job.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

DI JS

4.2.3. Obtaining Job Status Information (DISPLAY JS Command)

The DISPLAY JS command enables you to obtain information about jobs either initiated or running under your user-id. The command produces a 1-line display of the current status of the job about which you inquired. *You may request information only about jobs either initiated or running under your user-id.*

Format:

```
DISPLAY△JS[, jobname]
```

Parameter:

jobname

Is the 1- to 8-character name of the job about which you want information. If you include a job name with the DI JS command, you will receive information about the job, whether it is in main storage or on a job queue. If you do not include a job name, you will receive information about all jobs under your user-id that are in main storage, but not those on job queues. Information about jobs is displayed one line at a time. If you entered no job name, press TRANSMIT after the first line of job information is displayed to see information about other jobs. The command concludes when DISPLAY END appears after you press the transmit key.

Example:

```
DI JS,MYJOB
```

In this example, a display is produced of status information about a job named MYJOB.

The following screens are examples of the messages produced by the DISPLAY JS command. The examples show what information DI JS might display if it were entered against MYJOB as it proceeds through the various steps in job processing:

```
MYJOB IN STEP 01(LNKEDT00)-PRI=10 CPU-TIME=00:01:43.874
```

In this example, MYJOB is active in its first step, performing linkage editing. The CPU TIME portion of the display indicates that the linkage editor has had control of the CPU for 1 minute, 43 seconds, and 874 milliseconds. If the job is proceeding, you can reenter DI JS for MYJOB and see an increase in the CPU TIME figure.

```
MYJOB IN STEP 02(LIBS0000)-WAITING FOR I/O #00005736
```

In this example, MYJOB is in its second step, executing the librarian. Currently, the 5736th I/O operation of this step is being performed. If you reenter DI JS, you may see the I/O number increase. If MYJOB remains at #00005736, it might be stuck, requiring your intervention.

```
MYJOB IN STEP 03 -IN STEP PROCESSOR
```

In this example, MYJOB is between job steps. Step 03 either has just completed or is about to start.

```
MYJOB NOT YET SCHEDULED-INSUFFICIENT MAIN STORAGE
```

This example shows a message you would receive if MYJOB is not executing. In this case, MYJOB has been placed on a job queue but has not been scheduled for execution because not enough main storage is available.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- Invalid Access Attempt

An attempt was made to access a job that is not on the system or is under another user-id. A message is displayed indicating that the job is not on the system in either case. You are not permitted to obtain status information from jobs other than those under your user-id.

FR

4.2.4. Disconnecting a Workstation from a Job (FREE Command)

The FREE command enables you to manually disconnect a workstation from a job. After you issue a FREE command, the WORKSTATION mode of the workstation disconnected is unassigned and may be reassigned to another job. The FREE command is not normally required, because end-of-job processing procedures automatically disconnect all workstations assigned to the job. Also, when you issue a CONNECT command to a workstation already assigned to another job, the CONNECT command will issue a FREE command to disconnect the workstation before the CONNECT command is accomplished. The FREE command is most useful in allowing a frequently accessed job to be idled for short periods of time, but not removed from main storage. This permits the job to be accessed quickly, without having to wait for it to be rescheduled.

Format:

FREE

There are no parameters associated with this command.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or syntax was entered improperly.

- Workstation Not Assigned

You cannot disconnect a workstation that was not connected to a job in the first place.

- Command Not Permitted with Job Control

You may not use the FREE command when you specified automatic connection of workstations in the job control stream used to run the job.

PA**4.2.5. Suspending Job Processing (PAUSE Command)**

You use the PAUSE command to suspend processing of a job. You may enter the PAUSE command at any time, and processing of the specified job is immediately suspended. If the job is between job steps, PAUSE will take effect at the beginning of the next job step. You reactivate a job suspended by the PAUSE command by entering the GO command. *You may suspend only jobs executing under your user-id.*

Format:

PAUSE△jobname

Parameter:

jobname

Specifies the name of the job you wish to suspend.

Example:

PA MYJOB

In this example, the job MYJOB is suspended.

The following conditions may cause this command to be rejected:

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

■ No Job Name Specified

You must specify a job name with this command.

GO

4.2.6. Reactivating Suspended Jobs (GO Command)

The GO command is used to reactivate jobs that were suspended by using the PAUSE (4.2.5) command, or by job control operations.

Format:

```
GO△jobname
```

Parameter:

jobname

Is the name of the job you want to reactivate.

Example:

```
GO MYJOB
```

In this example, the job named MYJOB is reactivated after having been suspended.

The following conditions may cause this command to be rejected:

- Incorrect Syntax

The command is misspelled or ambiguous, or the command or parameters were improperly entered.

- No Job Name Specified

You must specify a job name with this command.

- Job Not Suspended

The job you are trying to reactivate was not suspended.

RES**4.2.7. Resuming Subsystem Execution (RESUME Command)**

The RESUME command enables you to resume execution of the general editor or the BASIC programming language, which was suspended when the workstation entered SYSTEM mode.

Format:

RESUME

There are no parameters associated with this command. Entering anything as a parameter for this command causes it to be rejected.

The following conditions may cause this command to be rejected:

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command was entered with parameters.

■ No Program to Resume

No subsystem was suspended, so none can be resumed.

SCR

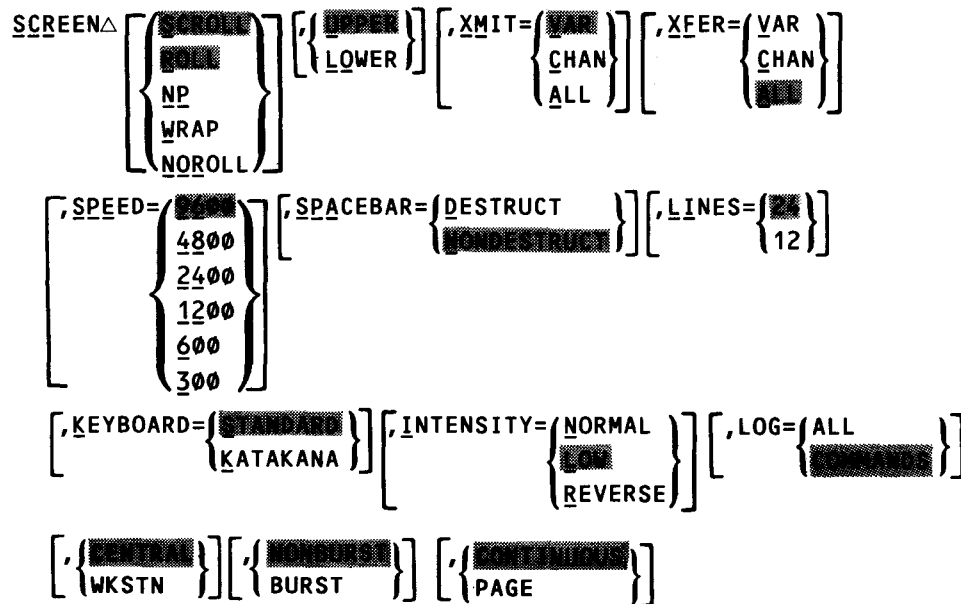
4.2.8. Altering Display Characteristics (SCREEN Command)

The SCREEN command enables you to alter some operational characteristics of the workstation or terminal with which you are working. The SCREEN command sends you a message informing you when it has completed processing. The SCREEN command permits you to alter screen control and display characteristics and input translation.

You can issue the SCREEN command at any time during a workstation session. During a session, it affects batch jobs as well as interactive facilities like the general editor, BASIC, and DDP. Moreover, each workstation user can issue a SCREEN command. If several workstations are connected to one job, each workstation operator can control his workstation or terminal independent of other users.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the last one.

Format:



Parameters:

NP/WRAP/NOROLL

Specifies screen control characteristics. Normally, the cursor moves from the last character position on the last line of the screen to the first character position on the top line of the screen. This is termed *wrapping*. You may change the screen so that, when the cursor reaches the last character position of the bottom line, it moves to the first character position of the top line of

the screen and the contents of the screen are erased. This is the *NP* option. Or, you can change the screen so that, when the cursor reaches the last character position of the bottom line, it will cause the bottom line to move up one and bump the top line off the screen. The cursor then moves to the first character position of the new, blank, bottom line. This is termed *scrolling*. To change from wrapping to scrolling, specify SCROLL or ROLL. If you have changed from wrapping to scrolling and wish to return to wrapping, specify WRAP or NOROLL. When applied to a System 80 workstation or System 80 console workstation, this parameter affects only WORKSTATION mode.

UPPER/LOWER

Specifies whether lowercase letters should be translated to uppercase upon input or remain lowercase. Specify UPPER if you want lowercase letters translated to uppercase. Specify LOWER if you want the lowercase letters to remain lowercase. When applied to a System 80 workstation or console workstation, this parameter affects only WORKSTATION mode.

XMIT=VAR/CHAN/ALL

Specifies what portion of the data entered on the screen is transmitted to the system. VAR specifies that only the unprotected data on the screen be transmitted. This is the most widely used mode and is required by screen format services. CHAN specifies transmitting only those fields changed by the workstation user. ALL specifies that all data, protected and unprotected, be transmitted. *This parameter is valid only when specified for a System 80 workstation, a console workstation, or a UTS 400 terminal.*

XFER=VAR/CHAN/ALL

Specifies what portion of the data entered on the screen is to be transferred to a peripheral device connected to the workstation or terminal. The three options, VAR, CHAN, and ALL, are the same as for the XMIT parameter. *This parameter is valid only when specified for a System 80 workstation, a console workstation, or a UTS 400 terminal.*

SPEED=~~9600~~/4800/2400/1200/600/300

Specifies the speed of the line connecting the workstation and any peripherals attached to the workstation.

The speeds are given in bits per second. The default value of 9600 bits per second is for direct-connected workstations. *This parameter is valid only when specified for a System 80 workstation or console workstation.*

SPACEBAR=DESTRUCT/~~NONDESTRUCT~~

Specifies whether the space bar destroys the character beneath the cursor as it advances or merely advances the cursor. DESTRUCT specifies that characters beneath the cursor are replaced with blank spaces as the cursor advances. NONDESTRUCT specifies that the spacebar merely advances the cursor without destroying the characters beneath it. *This parameter is valid only when specified for a System 80 workstation or console workstation.*

LINES=24/12

Specifies the number of lines on the workstation display screen. The indicator line present on workstation screens is not included in this count. The lines option does not apply when your workstation is in data mode (that is, if you are using the general editor or any other interactive facility). *This parameter is valid only when specified for a System 80 workstation or console workstation.*

KEYBOARD=STANDARD/KATAKANA

Specifies whether the keyboard generates standard English alphabetic characters or Katakana (Japanese) characters. *This parameter is valid only when specified for a System 80 workstation or console workstation having the necessary hardware to produce either Katakana or English characters.*

INTENSITY=NORMAL/LOW/REVERSE

Specifies the type of video available for protected fields of data displayed by programs. LOW specifies that low-intensity video is produced. REVERSE specifies that reverse video is produced. NORMAL specifies that the protected fields appear the same as the unprotected fields at normal intensity. *This parameter is valid only when specified for a System 80 workstation or console workstation.*

LOG=ALL/COMMANDS

Specifies which commands your workstation log will include. LOG=ALL specifies that your logs show all commands you enter – both in system mode and workstation mode. LOG=COMMANDS, the default, specifies that your logs show only commands entered in system mode. For example, if you specify LOG=ALL and activate the general editor, your log will show the system command EDT that activated the general editor, along with all of the editor commands you enter in workstation mode during your EDT session. However, if you do not specify the LOG parameter or specify LOG=COMMANDS, your log shows only the system command EDT, without any of your EDT commands.

CENTRAL/WKSTN

Specifies where your system prints your print files. Specifying CENTRAL, or omitting the parameter, directs all your print files to the central site printer. When it prints them depends on your specification for the NONBURST/BURST parameter of the SCREEN command. Specifying WKSTN directs all print files to the workstation auxiliary printer as soon as they complete, regardless of your specification for the NONBURST/BURST parameter. WKSTN applies only to print files for interactive products, specifically, EDT, ESCORT language, BASIC, or the interactive services PRINT command. (For print files from your own jobs to be printed at an auxiliary printer, their job control must specify auxiliary printer output. See spooling and job accounting concepts and facilities.) If you choose WKSTN, make sure the auxiliary printer is available and ready for use.

NOTE:

If you specify WKSTN at a workstation that is not connected to an auxiliary printer, your system holds your print files until you:

1. *log off that workstation;*
2. *log on to the workstation that is connected to an auxiliary printer under the same user-id; and*
3. *issue an RP spooling command.*

Then, your system prints your files at the auxiliary printer.

The SCREEN command issued with either of these parameters does not affect the printing of any file that had already begun before you issued the SCREEN command.

NONBURST /BURST

Specifies when your system prints all print files that you've sent to the central site printer through the CENTRAL parameter of the SCREEN command. Specifying NONBURST, or omitting the parameter, tells your system to print all print files that you sent to the central site printer according to your specification for the SPOOLBURST parameter during SYSGEN. In other words, if SPOOLBURST=YES at SYSGEN, then your system prints all print files as soon as they complete. If SPOOLBURST=NO, it holds all print files until you log off. Then it prints them all together, along with your workstation log file. Specifying BURST tells your system to print all print files that you've directed to the central site printer as soon as they complete, regardless of what you have specified for the SPOOLBURST parameter at SYSGEN.

The SCREEN command issued with either of these options does not affect the printing of any file that had already begun before you issued the SCREEN command.

NOTE:

In some cases, system programs override NONBURST or BURST. For instance, the EDT LIST command includes a parameter that lets you print files immediately, without waiting until you log off. If you were to specify this parameter, your system would print your files immediately, even if you had previously specified SCREEN NONBURST during your current workstation session and had specified SPOOLBURST=NO during SYSGEN.

CONTINUOUS /PAGE

Specifies the printing mode for any 0791 auxiliary printer connected to the workstation. CONTINUOUS causes output to be printed in continuous mode. That is, output prints continually as in normal printer operation. This option is the default. PAGE causes output to be printed in page mode. That is, the system requests the operator to insert the next page into the printer prior to the printing of each page. Both options are ignored if no 0791 printer is attached to the workstation.



↓

You can enter a SCREEN command that specifies PAGE or CONTINUOUS any time before the print file starts printing. The page mode remains in effect unless changed by another SCREEN command or unless the system IPL procedure is performed again. In the second case, the continuous mode becomes effective. If you alternate between the two modes, you can enter the command before you respond to the forms mount message. If the SCREEN command is entered after a file begins printing, the command is not effective until the next file is printed. Logging off the workstation has no effect on the page mode setting.

↑

Example:

```
SCR ROLL,LOWER
```

In this example, screen control is set for scrolling. Lowercase letters remain lowercase and are not translated to uppercase.

The following condition may cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

ST

4.2.9. Terminating a Job at the End of a Job Step (STOP Command)

The STOP command enables you to terminate a job at the completion of the currently executing job step. The STOP command differs from the CANCEL command in that the CANCEL command immediately halts the job, without allowing completion of the currently executing job step. *You may perform a STOP only for jobs executing under your user-id.*

Format:

```
ST $\Delta$ jobname
```

Parameters:

jobname

Specifies the name of the job you wish to bring to an orderly termination. The job name may be up to eight characters long.

Example:

```
ST MYJOB
```

In this example, the job named MYJOB is brought to an orderly termination.

The following conditions may cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were entered improperly.

- No Job Name Entered

You must enter a jobname to execute this command.

TEL

4.2.10. Sending Messages to the System Operator (TELL Command)

The TELL command enables you to send messages not requiring a response to other users or the system operator.

Format:

```
TELLΔ {user-id}, 'text'  
      {ALL }
```

Parameters:

user-id/ALL

Specifies the user-id of the user to whom you wish to send the message. If you do not specify a user-id, the message will be sent to the system operator. The console, however, has a user-id that you may use in communicating with the system operator; it is `$$CON`. If you wish to send a message to a user who has initiated an ENTER file, enter `$$MAS` as the user-id. If you specify ALL, the message will be sent to all users on the system, as well as the system operator.

text

Is the text of the message to be sent to the operator. A comma must separate the text from the user-id. If there is no user-id, you do not need to enter a comma. The text may be a maximum of 48 characters long. Longer messages are rejected. *The text must be enclosed in apostrophes and, therefore, can't contain apostrophes itself. So, be sure not to use contractions in your text. For example, enter 'WHERE IS DOOABCD?' instead of 'WHERE'S DOOABCD?'*

Example:

```
TEL 'PRINTOUT FROM PAYJOB IS NO GOOD'
```

In this example, the user is telling the system console operator that the printout of the specified job (PAYJOB) is unacceptable.

The following conditions may cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, apostrophes were used incorrectly, or the user-ID entered was longer than six characters.

- **Messages Too Long**

The message text was longer than 48 characters. The message is rejected.

4.3. COMMANDS TO CONTROL SPOOLING

4.3.1. General

These commands allow you to control the process of *spooling* (simultaneous peripheral operations online). Spooling permits both the high-speed components of your system (such as the central processor) and the low-speed components (such as the printer) to operate at their optimum speed. Spooling acts as the buffer to ensure that high-speed components are not bound by the slow-speed components. In the operation of spooling, input from a low-speed device such as a card reader is collected in the *spool file* on disk for high-speed transfer to the central processor. The high-speed output from the central processor is transferred back to the spool file, from which it can be output through a low-speed device such as a printer or card punch. This is by no means a complete explanation of the spooling process; it would be inappropriate to undertake a detailed explanation in this manual. For complete information on spooling, refer to the spooling and job accounting concepts and facilities manual.

NOTE:

You may access only those spooled files created by jobs initiated or running under your user-id.

4.3.2. Spooling Command Directories and Modifiers

When you enter a command to act upon a spooled file, you also enter information to define the file to the system. The information you enter to define the file is of two types:

1. The spool file directory
2. The spool file modifiers

4.3.2.1. Spool File Directories

Whether low-speed input or output devices interface directly with the central processor or are buffered by the spool file is transparent to your program. It always considers the input to be coming directly from the low-speed device. Therefore *spooled* files are identified in the *spool* file according to the low-speed device with which they are associated. For example, a file coming originally from a card reader would be identified as being in the card reader (RDR) directory. Note, however, that *diskette input* files are in the RDR directory, while *diskette output* files are in the PUNCH directory. Note also that there is a directory called the LOG directory. This directory contains job log information on jobs running in the system. Job log information includes the job control used to run the job, messages sent to the workstation or console controlling the job, and job accounting information. Table 4-1 lists the spool file directories you may access.

Table 4-1. Spool File Directories

| Directory | File Function |
|-----------|---|
| ALL | Makes all directories accessible to the command in which it is specified. |
| LOG | Indicates that the file referenced is a job log file. |
| PUNCH | Indicates that the file is to be output to either a card punch or a diskette. |
| PRINT | Indicates that the file is to be output to a printer. |
| RDR | Indicates that the file was originally input from a card reader or diskette. |

4.3.2.2. Spooled File Modifiers

You may further define the characteristics of the file you wish to reference in your command. The spooled file modifiers serve this purpose.

- ACCT=acctno

Specifies a 1- to 4-character number that is the account number of the job creating or using the file.

- ■ CART=cartridge-name

Specifies a 1- to 8-character identification of the print cartridge to be used in the printer specified to print the file referenced.

- DEV=nnnn

Specifies a 3- or 4-character number designating the type of device used for input or output of the file referenced. The device numbers you are permitted to enter, and the hardware with which they are used, are found in Table 4-2.

- FILE=filename

Specifies the name of the file. It may be from one to eight characters long.

- FORM=formname

Specifies the name of the form to be used when printing the file. It may be from one to eight characters long.

- JOB=jobname

Specifies the name of the job creating or using the file. It may be from one to eight characters long.

■ STEP=stepno

Specifies the number of the job step that created or uses the file. It must be three characters long and left-justified with zeros.

Table 4-2. Print Spooling Device Numbers

| Number | Device | Hardware Use |
|--------|--|-------------------------|
| 768 | 0768 Printer | Series 90 only |
| 770 | 0770 Printer | Series 90 only |
| 773 | 0773 Printer | Series 90 only |
| 776 | 0776 Printer | Series 90 and System 80 |
| 778 | 0778 Printer | Series 90 |
| 789 | 0789 Printer | System 80 only |
| 9300 | 9300 System running as a Series 90 peripheral device | Series 90 only |

4.3.3. The Output Writer

An output writer is a spooling facility that is responsible for printing and punching spooled output from a job. Under normal processing conditions, an output writer loads automatically and is transparent to a user. When it processes output, it does so according to specifications set at SYSGEN time.

You can, however, manually load an output writer from your workstation. By manually loading an output writer, you can control its mode of operation and its processing criteria. The following function codes are used with the PR/PU and RP spooling commands to control the output writer. See the formats for these commands (4.3.4 and 4.3.5) to learn which function codes are valid under which circumstances.

4.3.3.1. Function Codes for the Output Writer

{ BURST }
{ BX }

Places the output writer in burst mode. In burst mode, output files are available for processing before a job has terminated. The output writer writes an output data file as soon as the job step that created the file is completed. If BX is entered with file modifiers (4.3.2.2), the output writer terminates after processing all files that satisfy the modifiers. If BU is entered, the output writer requests another function when more files exist that do not satisfy the modifiers.


BYPASS

Terminates processing of the current file. The current file is closed and the output writer continues processing the next file. Bypassed files can be restarted later.

COPIES,nnn

Sets the number of copies the output writer is to produce for each file it processes. From 1 to 255 copies (nnn) may be specified. If no number is specified, one copy is assumed. The file is closed when processing is completed.

DELETE

Deletes the file being processed and proceeds with the next file to be processed.

DISPLAY

Displays the status of the current file on the workstation screen. The information displayed is:


- File name
- Job name
- Current page (card) number
- Total pages (cards) in file
- Program name
- Job step number
- Number of remaining copies
- Existence of a breakpoint

HALT

Terminates the output writer after the current file (if any) is processed. If the file being processed has multiple copies, the remaining copies are produced when the output writer is reloaded.

HOLD

Places the current file in a hold state and begins processing the next file. Files in a hold state are not available for processing until released by the BEGIN command.



NBURST

Places the output writer in nonburst mode. If specified while the output writer is processing a file, the function does not take effect until file processing is completed.

RETAIN

Retains the currently active file in a HOLD state in the spool file after it is processed. The retained file is unavailable for additional processing until released via the BEGIN command. (Otherwise, delete the retained file via the DELETE command.)

RESTART { ,nnn
{ ,PAGE,nnnn }
{ ,CARD,nnnn }

Restarts processing of the currently active file from a number of pages or cards. If no number is specified, the output writer restarts processing from the beginning of the file. If only nnn is entered, file processing is restarted nnn pages or cards back from the current position of file. If PA or CA is entered with nnnn, the file is positioned back to the page or card identified by nnnn.

SKIP { ,nnnn
{ ,PAGE,nnnn }
{ ,CARD,nnnn }

Directs the output writer to skip forward a specific number (nnnn) of pages or cards, or to skip forward to a specific page number (PAGE,nnnn) or card number (CARD,nnnn). After positioning, a request is made for another function.

SIOP[,PAGE]

Directs the output writer to stop processing. If PAGE is omitted, the output writer terminates immediately. If PAGE is included, the output writer terminates after printing the complete current page. The file being processed is closed but not deleted. When accessed by another output writer, the file is processed from the point at which it was closed.

PR/PU

4.3.4. Manually Loading an Output Writer (PR/PU)

The PR/PU spooling command allows you to manually load an output writer to print (PR) or punch (PU) spooled files associated with your job. Normally, the output writer is loaded automatically to process the output files from your job. However, under certain circumstances, you might choose to load an output writer yourself. In doing so, you can also control output processing by specifying function codes.

PR/PU is useful when, for example, your output writer normally operates in nonburst mode and you need an output file printed or punched before your job has terminated. In nonburst mode (a SYSGEN option), the output writer won't write an output data file to a device until the job that created the file has terminated and the job's header and log have been written.

You can also use PR/PU to print or punch spooled files that were recovered after a "warm" start. It is important to note, however, that if a device hardware error occurs while your files are being printed or punched, it is up to you to resume processing. When a hardware error occurs, the following message is routed to your workstation:

```

0n{PR}(did) UNRECOVERABLE OUTPUT ERROR,ENTER I OR FUNCTION.
   {PU}
  
```

The did is the device address of the printer or punch where the error occurred. To ignore the error, key in I as a solicited response. Otherwise, enter an output writer function code (4.3.3.1) like STOP or RESTART.

Format:

```

{PR}△[function-code][,ACCT=acctno][,CART=cartridge-name][,FILE=filename]
{PU} [,FORM=formname][,JOB=jobname]
  
```

Parameters:

function-code

Specifies the output writer's mode of operation and processing criteria. If a function code is omitted, the output writer is loaded in the mode (burst/nonburst) indicated at SYSGEN. See 4.3.3.1 for a complete list of output writer function codes. The BU and BX function codes allow you to change the output writer's mode of operation. The first time you issue the PR or PU spooling commands during one workstation session (from logon to logoff), BU and BX are the only valid function codes you can choose. Once the output writer starts printing or punching, you can enter the PR or PU command with any other function code to change the output writer's processing criteria.

NOTE:

Function code keyins cannot exceed 28 characters in length, including commas.

Examples:

PR BX,FO=MYFORM

The output writer is loaded in burst mode to group all spooled output files associated with the form name specified and then print the groups on a first-in, first-out basis. The output writer terminates when all form name file groups have been printed.

PU BU,JO=MYJOB

The output writer is loaded to process, in burst mode, all punch files created by MYJOB.

RP

4.3.5. Manually Loading an Output Writer to an Auxiliary Printer (RP)

The RP spooling command allows you to manually load an output writer to print to an auxiliary printer. The auxiliary printer can be connected to either a local or remote workstation.

As we explain in the spooling and job accounting concepts and facilities manual, in many cases it's not necessary for you to use the RP command to load an output writer to an auxiliary printer. Normally, your system loads it automatically, as long as you've generated your system to use auxiliary printers and directed your print files to an auxiliary printer prior to the time your system prints your print files.

You direct your print files to an auxiliary printer in one of two ways, depending on what type of program or job you're working with. If you're running an interactive system program, such as EDT, direct your print files to an auxiliary printer through the interactive services SCREEN WKSTN command. If you're running your own jobs, direct your print files to an auxiliary printer through job control. See spooling and job accounting concepts and facilities for details on directing output to an auxiliary printer.

In some cases, however, manually loading an output writer to an auxiliary printer using the RP command is not only desirable but necessary.

One case where the RP is necessary is when you've directed any of the output files from your jobs to another user's workstation auxiliary printer. In other words, you included another user's identification on either the // OPTION OUT or // ROUTE statement in the job control for your job. Here, the other user must issue an RP command to start the printing of your files. Printing in this case is not automatic. (Similarly, if you are not the initiator of a job, but output is directed to your auxiliary printer, you must issue an RP command to start printing. To find out whether any output files have been routed to your user-id, use the DISPLAY SPL command.)

Other cases when you use the RP command are similar to when you use the PR/PU command. Specifically, those cases might be:

- Your output writer normally operates in nonburst mode, and you need an output file printed before your job terminates. In this case, you could manually load an output writer to your auxiliary printer.
- You want to recover output files after a warm start or to resume printing after an auxiliary printer hardware error.
- You can also use RP to control an output writer's processing criteria by specifying function codes with the command after the output writer has begun printing.

- You need the RP command when you issue an interactive services SCREEN WKSTN command at a workstation that is not connected to an auxiliary printer and then issue either a PRINT command or an EDT LIST command. In this case, you must:
 1. log off that workstation;
 2. log on under the same user-id at the workstation that is connected to an auxiliary printer; and
 3. issue an RP command.

These actions cause your system to print the print files from your interactive system program sessions at your auxiliary printer. Only by performing all three actions can you get your print files printed at the auxiliary printer.

In all cases, remember that the RP command does not work independently; to use the RP command, you must have generated your system to use auxiliary printers and directed your print files to an auxiliary printer prior to the time your system prints your print files.

NOTES:

1. *The // ROUTE and // OPTION OUT statements have additional DDP parameters not discussed here. You cannot use RP to direct printing to a DDP site. RP applies only to auxiliary printers connected to local or remote workstations.*
2. *Output files that you direct to an auxiliary printer can be secured or unsecured. If a file is secured, the user the output is directed to must be logged on for printing to begin. Secure a file via the // SPL statement or jproc. See the job control user guide for details.*

Format:

```
RP△[function-code][,A_CCT=acctno][,CART=cartridge-name][,FILE=filename]
    [,FORM=formname][,JOB=jobname]
```

Parameters:

function code

Specifies the output writer's mode of operation and processing criteria. If a function code is omitted, the output writer is loaded in the mode (burst/nonburst) indicated at SYSGEN. See 4.3.3.1 for a complete list of output writer function codes. The BX function code lets you change the output writer's mode of operation to nonburst mode and is the only valid function code the first time you issue the RP spooling command during one workstation session (from logon to logoff). Once the output writer starts printing, you can enter the RP command with any other function code to change the output writer's processing criteria.

 Examples:

```
RP BX,JO=MYJOB
```

In this example, an output writer is loaded to print to an auxiliary printer in burst mode to process the output data files from the job MYJOB.

The following conditions could cause this command to be rejected:

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

■ Output not Destined for an Auxiliary Printer

You cannot direct output to an auxiliary printer unless output was destined for it via job control.

DI ACT/DI SPL**4.3.6. Obtaining Spooled File Information (DISPLAY ACT and DISPLAY SPL Commands)**

These commands enable you to obtain information about either spooled files being created by the program (DISPLAY ACT) or completed spooled files (DISPLAY SPL). The two commands function in a similar manner and produce similar displays. They enable you to obtain information, such as the name of the job creating the file or files, number of pages or cards created, and the number of copies of the file to be produced upon output to the printer or punch.

DISPLAY ACT Format:

```

DISPLAY△ACT [ , (ALL) ] [ , ACCT=acctno ] [ , CART=cartridge-name ] [ , DEV=( 768
              { PRINT }
              { PUNCH }
              )
              ] [ , FILE=filename ] [ , FORM=formname ] [ , JOB=jobname ] [ , STEP=stepno ]

```

DISPLAY SPL Format:

```

DISPLAY△SPL [ , (ALL) ] [ , ACCT=acctno ] [ , CART=cartridge-name ] [ , DEV=( 768
              { LOG
              { PRINT
              { PUNCH
              { RDR
              }
              )
              ] [ , FILE=filename ] [ , FORM=formname ] [ , JOB=jobname ] [ , STEP=stepno ]

```

NOTE:

If you do not enter a directory, all directories will be displayed.

The DISPLAY SPL and DISPLAY ACT commands display information in a series of single lines. The first display line gives you a list of the number of files the system has found meeting your requirements, the number of pages produced, and the number of card images produced. For DISPLAY ACT, these totals represent files currently being created; for DISPLAY SPL, they represent the completed files. The first display is as follows:

```
DI01 status FILES=ffff PAGES=ppppp CARDS=cccc
```

where:

status

Specifies whether the line is displaying a list of files *QUEUED*, in the *HOLD* state, or *IN-PROGRESS* being processed by the output writer. If RDR is specified as the spool queue, *IN-PROGRESS* specifies that files are being processed by programs.

fffff

Represents the number of files.

ppppp

Represents the total number of pages. *Log files are not included in the page count.*

cccc

Represents the number of card images.

This display will be produced for each of the three statuses containing one or more files. If there are more than one, the system will display them to you *one at a time*. You must press the *XMIT* key to view the remaining displays.

After you have seen these displays, the system will ask you if you want to see further details of the spooled files requested in your *DI ACT* or *DI SPL* command. If you entered a *DI ACT* command, the following line will be displayed:

```
DI02 SPOOL FILE DETAILS? ***Y,N,***
```

If you entered a *DI SPL* command, the following line will be displayed:

```
DI02 SPOOL FILE details? ***Y,N,Q,H,I,S,SQ,SH,SI***
```

where:

Y

Specifies that the system is to display all spooled file details.

N

Specifies that the display should not be continued; no spooled file details should be displayed.

- Q Specifies to display all queued files.
- H Specifies to display all files being held.
- I Specifies that all files currently being processed by the output writer be displayed. If RDR is specified as the spool queue, I specifies that all files currently being processed by programs be displayed.
- S Specifies that you wish to see an abbreviated display of all files.
- SQ Specifies that you wish to see an abbreviated display of all queued files.
- SH Specifies that you wish to see an abbreviated display of all the files being held.
- SI Specifies that you wish to see an abbreviated display of all files in-progress.

You respond by keying in one of these nine choices.

If you respond with N, the display and command terminate. If you respond with Y, Q, H, or I, and a queue other than RDR was specified, the system displays the following:

```

DI04 JOBNAME  jobname  FILE  filename  STATUS  file  status
DI05 TOTAL  { PAGES } nnnnn  REMOTE-ID  xxxxxx  COPIES  nnn
           { CARDS }
           { LINES }
DI06 STEP-NUMBER  nnn  DEVICE-TYPE  xxxxx  BREAKPOINT  { Y }
                                           { N }
DI07 BAND-NAME  xxxxxxxx  FORM-name  xxxxxxxx  ACCT  xxxx
DI08 PROGRAM-NAME  xxxxxxxx  CONTINUE?  ***Y,N***

```

If you respond with Y, Q, H, or I and had specified the RDR queue, you will receive the following display:

```

DI09 DEVICE-TYPE RDR TOTAL-CARDS nnnnn VOL xxxxxx
DI10 LABEL XXXXXXXXXXXXXXXXXXXX CONTINUE? ***Y,N***

```

NOTE:

The VOL display appears only when the RDR file was spooled in from a diskette.

where:

JOBNAME jobname

Is the name of the job that created the spooled files.

FILE filename

Is the name of the file whose characteristics are being displayed.

STATUS

Is the status of the file whose characteristics are being displayed (QUEUED, HOLD, IN-PROGRESS).

TOTAL { PAGES } nnnnn
 { CARDS }
 { LINES }

Is the total number of pages, cards, or lines.

REMOTE-ID

Is the user-id of the workstation from which the job producing the spooled output was initiated.

COPIES nnn

Is the number of copies of the file to be made when it is transferred to the output device (printer, punch, etc.)

STEP-NUMBER nnn

Is the number of the job step within the job specified by the job name.

DEVICE-TYPE xxxxx

Is the type of output device the file will be sent to (printer, punch, etc.)

BREAKPOINT

Informs you whether or not the file has been breakpointed.

BAND-NAME xxxxxxxx

Is a name given to the print band to be used by the printer that will print the file. For example, a band name of 48-BUS. would indicate the printer is to use a 48-character standard business print band. Band names are initially specified at SYSGEN time. This name is the same as the cartridge name that you specified.

FORM-NAME

Is the name of the form to be used on the printer.

ACCT xxxxx

Is the account number of the job that created the file.

VOL

Is the volume serial number of the diskette read from.

LABEL

Is the label of the diskette read from.

These lines of information are displayed one at a time, and you must press the XMIT key after each line is displayed, to view the next line. If you respond to CONTINUE? on line 108 with Y, the displays will continue with other files. If you respond with N, the display terminates.

If you responded to line DI02 (SPOOL FILE DETAILS.....) with S, SQ, SH, or SI, the following display will be produced:

```
DI11  JOB=jobname  PROG=programe  FO=formname  (PA)=nnnnn  ST=nn
                                         {CA}
                                         {LI}
```

NOTE:

ST=nn indicates the step number of the job that created the file.

The system produces five DI11 lines on the screen (or however many, up to five, needed to display information for files that fit your criteria) and then asks you if you wish to continue:

```
DI12  CONTINUE SUMMARY?  ***Y,N***
```

Your entry of Y continues the display. An entry of N terminates the display.

If no files exist that fit the criteria you entered with the command, the system will display the following:

```
DI03  SPOOL FILE EMPTY
```

Example:

```
DI SPL PR,FILE=JOBCOST
```

This example causes the display of information about a completed file named JOBCOST, residing on the printer spool queue.

The following condition may cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

HO SPL**4.3.7. Holding Spooled Files (HOLD SPL Command)**

The HOLD SPL command enables you to place spooled files in a "hold" state. In the hold state, the files are unavailable for processing. To make them available, they must be released by using the BEGIN command (4.3.8). *You may place only completed spooled files in the hold state from the workstation, and you may hold only those files created by jobs running under your user-ID.*

Format:

```

HOLD△SPL [ , ALL
           { PRINT
           { PUNCH
           { LOG
           { RDR } ] [ , ACCT=acctno ] [ , CART=cartridge-name ] [ , DEV=
           { 768
           { 770
           { 773
           { 776
           { 778
           { 789
           { 9300 } ]
           [ , FILE=filename ] [ , FORM=formname ] [ , JOB=jobname ] [ , STEP=stepno ]

```

When you enter the HOLD SPL command, the following message will be displayed to you:

```
HO01 xxx.SPOOL FILES HELD
```

where:

xxx

Is the number of spooled files held by your command.

Example:

```
HO SPL,PRINT,CART=48-SCI
```

In this example, all print spooled files that require the use of a 48-character scientific print cartridge are held.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

BE SPL

4.3.8. Releasing Held Spooled Files (BEGIN SPL Command)

The BEGIN SPL command enables you to release spooled files being held by a HOLD SPL command. This command can release completed spooled files and spooled files held by the //SPL job control statement. This command causes the output writer to be loaded and placed in burst mode. However, an output writer called in burst mode from a workstation does not take over the printer. It waits its turn for the printer and then prints the files specified. *You may release only those files created by jobs running under your user-ID.*

Format:

```

→ BEGIN△SPL, [ { LOG } ] [ ,ACCT=acctno ] [ ,CART=cartridge-name ] [ ,DEV= { 768 } ]
                { PRINT }
                { PUNCH }
                { RDR }
                [ ,FILE=filename ] [ ,FORM=formname ] [ ,JOB=jobname ] [ ,STEP=stepno ]

```

When you enter the BEGIN SPL command, the following message will be displayed to you:

```

BE01   xxx SPOOL FILES RELEASED

```

where:

xxx

Is the number of spooled files released by your BEGIN SPL command.

Example:

```

BE SPL,PRINT,CART=48-SCI

```

In this example, print files requiring a 48-character scientific print cartridge, which had been held (see the example of the HO SPL command in 4.3.7), are released.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

DE SPL

4.3.9. Deleting Spooled Files (DELETE SPL Command)

The DELETE SPL command enables you to delete spooled files from spooled file directories. You may delete only completed files, that is, those that are not being processed by an output writer or user program. *You may delete only the files created by jobs running under your user-id.*

Format:

```
DELETE△SPL, { ALL
              LOG
              PRINT
              PUNCH
              RDR } [,ACCT=acctno][,CART=cartridge-name] [,DEV={ 768
                                                                770
                                                                773
                                                                776
                                                                778
                                                                789
                                                                9300 } ]
              [,FILE=filename][,FORM=formname][,JOB=jobname][,STEP=stepno]
```

NOTE:

There is no default value for spool directories in this command. You must enter a directory.

When you enter the DELETE SPL command, the following message is displayed to you:

```
DE01 xxx SPOOL FILES DELETED
```

where:

xxx

Is the number of spooled files deleted by your DELETE SPL command.

Example:

```
DE SPL,PRINT,CART=48-SCI
```

In this example, all print spooled files requiring the use of a 48-character scientific print cartridge are deleted.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

BR

4.3.10. Breakpointing Spooled Files (BRKPT Command)

The BRKPT command enables you to close one or more files and make them available to an output writer for printing or punching. The remainder of the file created after you enter the BRKPT command is placed in a new file. *You may breakpoint only the files created by jobs running under your user-id. Files which are to be placed on a diskette may not be breakpointed.*

Format:

→ `BRKPT△{P}, {PR}, _JOB=jobname[, _ACCT=acctno][, _CART=cartridge-name]`
`{I} {PU}`
`[, _DEV=(768`
`770`
`773`
`776`
`778`
`789`
`9300)] [, _FILE=filename][, _FORM=formname]`

Parameters:

P/I

Specifies whether you want the file breakpointed immediately (I) or at the end of the currently completing page (P). If, through the next parameter, you specify that you wish to breakpoint a file for *punching*, the file will be breakpointed immediately, whether you specify P or I.

PR/PU

Specifies the type of file you wish to breakpoint. PR indicates you wish to breakpoint a printer file; PU indicates you wish to breakpoint a punch file.

When you enter the BRKPT command, one of the following messages will be sent to you to inform you of the status of your command:

FILE filename FOR JOB jobname HAS BEEN BREAKPOINTED

This message indicates that your breakpoint has been successful.

FILE filename FOR JOB jobname UNABLE TO BE BREAKPOINTED

This message indicates that an I/O error has prevented successful breakpointing of the job.

JOB NAME NOT SPECIFIED FOR BREAKPOINT

You did not enter a job name as part of the BRKPT command. Breakpoint cannot be done without a job name.

BREAKPOINT REQUEST INVALID

You attempted to breakpoint a LOG, or RDR file, or a file created by a job running under another user-id.

FILE NOT AVAILABLE FOR BREAKPOINT

The system is not able to find the file specified to be breakpointed.

BREAKPOINT ALREADY IN PROGRESS FOR JOB jobname.

A previously entered breakpointing operation is in progress involving the job you wish to breakpoint. Your breakpointing operation cannot begin until the previous operation has completed.

JOB NOT AVAILABLE FOR BREAKPOINT

Either the job you wish to breakpoint is not in the system or is running under Data Base Systems, a data base system developed by Sperry Univac for European users.

Example:

```
BR I,PU,JOB=MYJOB
```

In this example, a punch file currently being created by the job named MYJOB is to be breakpointed.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

4.4. COMMANDS USED TO PERFORM UTILITY ROUTINES

The commands contained in this section enable you to interactively perform utility routines.

Utility routines perform the "housekeeping" chores that need to be done on every data processing system. These commands simplify the housekeeping chores by removing the need to create job control streams to run the utility routines. Utility routines perform such functions as erasing files to make more storage space available, allocating files for future use, making printed and punch card copies of files, and obtaining information about the status of files and listings of the files currently stored in the system.

The utility routines described in this section are not the only such routines available to you. Also available from the workstation are the DATA utilities, which enable you to work with the contents of data files on the system. See 5.2 for more information on the DATA utilities.

4.4.1. File Parameters

If you wish to use a command that acts upon a file (allocating, copying, erasing), you must define the file to be acted upon. You do this with a set of file parameters, entered with the command. The file parameters are entered as keywords. The following is an example of a keyword parameter:

```
VSN=volume
```

The keyword, VSN, defines the parameter to the system as the volume serial number of the file being referenced. The equal sign (=) tells the system that the next group of characters is the actual parameter. After the equal sign, you enter the actual volume serial number of the file being referenced. Thus, to enter this parameter for a volume with the volume serial number PAY346, you would key in VSN=PAY346.

In many cases, the first two or three letters of a keyword are underlined. This means that you need to enter only the underlined letters, not the entire keyword. If the keyword is not underlined, you must enter the entire keyword. You may always enter any whole keyword, and the system will accept it. Each keyword parameter must be separated from the one preceding it by a comma. The first keyword parameter must be separated from the command by a single space.

Many command formats include not only file parameters but other parameters, appearing after the keyword parameters, at the end of the parameter string. These other parameters are termed *command options*, and serve to modify the activity of the command with which they are associated. They must be separated from the keyword parameters by a single space, and from each other by a comma. They are defined with the commands they modify.

The keyword parameters are defined in the following section. They are listed alphabetically. Refer to the format of each command to find out which keyword parameters are needed for that particular command.

NOTE:

Some of the parameters may also be entered in a positional format. For more information on using the positional format, refer to Appendix D.

4.4.2. Keyword Parameters

ACCT=acct

Specifies the account number of the job that created the spool file you wish to access. The account number may be from one to four alphanumeric characters long.

ALL=

| |
|-----|
| YES |
| NO |

Specifies whether or not all spool files with the specified parameters are to be processed. If you specify YES, all the spool subfiles with the specified parameters will be processed. If you specify NO, only the first spool file found with the specified parameters will be processed.

↓

BESZ=n

For disk file:

Specifies the minimum I/O buffer size for the MIRAM file you want to access. Specifying a larger-than-minimum buffer size could reduce the number of disk I/O calls, thus causing your command to run faster. Not specifying a buffer size will cause this parameter to default to a buffer size appropriate for the file you are accessing.

For tape files:

Specifies the block size within a tape file.

↑

BKNO= { YES }
 { NO }

When you specify this parameter for a tape output file, it causes block numbers to be created on the tape as the file is being created. When you specify this parameter for a tape *input* file, it causes the block numbers of the file to be checked.

CONTIG= { YES }
 { NO }

Specifies whether the file to be created should be allocated contiguous or non-contiguous space. YES causes the file to be allocated contiguous space; NO causes the file to be allocated noncontiguous space. File space on data-set-label (DSL) diskettes must be contiguous; therefore, take the default YES when allocating space on DSL diskettes. *This parameter is specified only when creating a new file.*

COPIES= { n }
 { 1 }

Specifies the number of copies you wish made of a spool file. This parameter applies only to spool files created by commands such as the PRINT command. This parameter takes effect when the spool file is processed by an output writer and sent to the peripheral device for output. For example, specifying COPIES=2 causes two copies of the file to be made.

You may make up to 255 copies of a file.

DEVICE= { did
 DISKETTE
 DISK
 PRINT
 PUNCH
 RDR
 TAPE }
 }

Specifies the type of device you wish to use to read from, or write to, as a sequential file. did specifies the device address (did) of the device you wish to use. The first digit is the channel number; the second and third, the hardware

address. The other choices, DISKETTE...TAPE, specify the type of device to be used, but not a specific device. If you do not specify a device, the parameter will default to disk. If you are using a disk or diskette and specify its volume serial number, you do not need to enter the DEVICE= parameter. If the volume you want to use is mounted, the system will find it.

EXTEND= $\left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right\}$

Specifies whether the contents of a file are to be either overwritten or merged with new data. If you take the default YES, the existing file remains intact and the new data is added to the end. If you specify NO, the data is overwritten, but the file characteristics are preserved. To overwrite a file using new file characteristics, use the INIT=YES parameter instead of EXTEND. (The default values for EXTEND and INIT specify the same condition. The two parameters differ in how a file is overwritten with new data.) The EXTEND parameter is ignored if you specify INIT=YES. EXTEND applies only to MIRAM data files and tape files.

FILENAME= $\left\{ \begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right\}$

Specifies the name of the file being referenced. In all cases except two, the file name must be that of an existing file. The exceptions are file names entered with the ALLOCATE command and file names entered for files you are creating through the COPY command. In both cases, the names entered must be names you create. Physical file names may be from 1 to 44 alphanumeric characters long. Logical file names (for spool files) may be from 1 to 17 characters long. Tape file names may also be from 1 to 17 characters long. *The file name must be enclosed in either quotation marks or apostrophes if there are spaces, commas, or parentheses embedded within the file name.*

HOLD= $\left\{ \begin{array}{l} \text{L} \\ \text{N} \\ \text{Y} \end{array} \right\}$

Specifies whether or not a spooled file has been placed in a HOLD state. Files are held in two ways. The first is through the use of a HOLD command, from the workstation or system console, or the inclusion of a HOLD job control parameter. The second is by specifying the file to be retained after processing. A file placed in a HOLD state by the first method has not been processed. A file placed in a HOLD state by the second method has been processed once. If the file you are referencing is a log file, and is in a HOLD state, enter L. If the file you are referencing is a type other than a log file and is in a HOLD state, enter Y. If the file you are referencing, log or other type, is *not* in a HOLD state, enter N. If you are accessing a file on the reader queue (RDR), the HOLD option is forced to NO. If you omit the HOLD option for any other type of spool file, all HOLD options will be tried in order: HOLD=NO, HOLD=YES, HOLD=LOG. Then, any file meeting the specified file characteristics, including the HOLD option, is located. Retained input files are not placed on HOLD.

$$\underline{INC} = \left\{ \begin{array}{l} n \\ \text{NO} \end{array} \right\}$$

Specifies the number of cylinders of disk storage space to be added to a file when it fills its original allocation of space and requires further space. *This parameter does not apply to files on data-set-label diskettes. DSL diskettes cannot be extended.*

$$\underline{INIT} = \left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right\}$$

Specifies whether the contents of a file are to be overwritten with new data being added. If you specify NO (the default), the old information remains intact and the new data is added to the end of the file. If you specify YES, the original file contents are overwritten with the new data, but the original file characteristics are not preserved. (The only exception would be if the original file was created by using the default values for all the file parameters.) INIT assumes the default values for all file parameters. To overwrite a file using the old file characteristics, use EXTEND=NO, not INIT. Otherwise, use INIT and specify your new file parameters. Init applies only to MIRAM data files and tape files.

JOB=jobname

Specifies the name of the job that produced the spool file you wish to access. The job name may be from one to eight alphanumeric characters long.

$$\underline{KEY}i = \left\{ \begin{array}{l} n:m \\ \left(\left(n:m, \left\{ \begin{array}{l} \text{DUP} \\ \text{NDUP} \end{array} \right\}, \left\{ \begin{array}{l} \text{CHG} \\ \text{NCHG} \end{array} \right\} \right) \right) \end{array} \right\}$$

Specifies the starting and ending column positions of a key ($i=1-5$). You may repeat this parameter for as many of the five keys you have available to you when creating a file. If you use this parameter with an existing file, your specifications must match those of the present file. DUP/NDUP specifies whether or not duplicate keys are allowed. CHG/NCHG specifies whether or not keys may be changed.

$$\underline{KEYNO} = \left\{ \begin{array}{l} n \\ \text{NO} \end{array} \right\}$$

Specifies the number of the key to be used to access a MIRAM file. n may equal 1 to 5.

MODULE=module name

Specifies the name of the module being referenced. It may be from one to eight alphanumeric characters long.

$$\underline{QUEUE} = \left\{ \begin{array}{l} \text{LOG} \\ \text{PRINT} \\ \text{PUNCH} \\ \text{RDR} \end{array} \right\}$$

Specifies the spool file directory of the spool file you are referencing. This parameter is valid only when referencing a spool file. You may enter LOG, PRINT, PUNCH, or RDR. LOG indicates that the file is in the log spool directory.

PRINT indicates that the file is in the printer spool directory. PUNCH indicates that the file is in the card punch spool directory. RDR indicates that the file is in the card reader spool directory.

RCB= $\left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right\}$

Specifies whether or not a MIRAM file should be created with a record control byte.

RCFM= $\left\{ \begin{array}{l} \text{FIX} \\ \text{VAR} \\ \text{FIXUNB} \\ \text{FIXBLK} \\ \text{VARUNB} \\ \text{VARBLK} \\ \text{UNDEF} \end{array} \right\}$

Specifies the record format of a file. For a MIRAM file or a unit record file, you may specify either FIX or VAR. For a tape file, you may specify any of the others. This parameter applies only to tape, unit record, and MIRAM files. If you enter this parameter for an existing file, it must match the present record format of the file. For more information on record formats, refer to the consolidated data management concepts and facilities.

RCSZ=n

Specifies the record size of the file or module being accessed. The default value of this parameter is based on the type of file or module you want to access. The following table gives the default values:

| | | |
|-----------------|--|---|
| Library modules | 128 bytes | |
| MIRAM files | 256 bytes (or record size of existing file) | |
| Tape files | 256 bytes (or record size of existing file) | |
| Card files | 80 bytes | |
| Printer files | 132 bytes | ← |
| Diskette files | Same as MIRAM files if recorded in format-label mode; 128 bytes if recorded in data-set-label mode. | ← |

RDPASS=password

Specifies a password used to control read access to the file being referenced. A password is required if the file is listed with a read password in the catalog file. If the file referenced is to be cataloged, you must specify passwords if you want the file to be only accessible through the passwords. Passwords may be from one to six alphanumeric characters long. You may specify that a file have both read and write passwords, or only one or the other, or no passwords at all.

SAT= {YES }
 {NO }

Specifies whether the file being created should be a SAT (system access technique) or a MIRAM library file. If you are creating a library file, you must specify YES if you want the system to allocate a SAT file. Otherwise, the system will allocate a MIRAM library file. You should specify a SAT file when the program to be placed in the file is to be compiled, linked, or in some way accessed by the system.

SCSZ={n }
 {256 }

Specifies the sector size of the MIRAM file. This parameter is specified only when accessing a file on a volume mounted on a *selector channel* device. The following are selector channel devices:

8414 disk drive

8425 disk drive

8430 disk drive

8433 disk drive

NOTES:

1. *These devices are available on Series 90 hardware only.*
2. *This keyword defines the sector size of the disk pack when creating a file; it must match the file for an existing file.*

SIZE=n

Specifies the size of a new file to be allocated. The value of n specifies number of cylinders for disk files and number of blocks for diskette files. This parameter is specified only when creating a new file.

SKIP={n }
 {0 }

Specifies the number of spool files, with parameter values identical to those specified, that are to be skipped before accessing a spool file for processing. The default value of 0 indicates that the first file found matching the specified parameters is accessed.

TYPE={module-type }
 {S }

Specifies the type of module being referenced. For SAT files, the types permitted are source (enter S), macro (enter M), procedure (enter P), proc name (PN) load (enter L), or object (enter O).

NOTE:

A specification of L means both blocked and unblocked load modules.

For MIRAM files, you may specify format (enter F), saved job control stream (enter J), screen format (enter FC), menu (enter MENU), help screen (enter HELP), or any other type available on your system. You may create your own MIRAM module types, identifying them with a 1- to 4-character type. The MIRAM module types can serve as qualifiers for the modules they are associated with. ←

VSN=volume

Specifies the volume serial number of the volume on which the file you are referencing is located. The volume serial number is required if the file is *not* cataloged. The volume serial number may be from one to six alphanumeric characters long. The VSN for SYSRES is RES.

NOTE:

You may reference system files (those with file names of \$Y\$XXX) without specifying a volume serial number. The system assumes the files are on the SYSRES disk volume.

WRPASS=password

Specifies a password needed to control write access to the file being referenced. A password is required if the file is listed with a password in the catalog file. If the file referenced is to be cataloged, you must specify passwords if you want the file to be accessible only through the passwords. Passwords may be from one to six alphanumeric characters long. You may specify that a file have both read and write passwords, or only one or the other, or no passwords at all.

AL

4.4.3. Allocating Files (ALLOCATE Command)

The ALLOCATE command enables you to allocate files interactively without using job control statements. The ALLOCATE command reserves space for a file on a disk or diskette of your choosing, and identifies the reserved space for use by the file you create. The parameters entered with the command permit the system to reserve enough space for your file and to provide for increasing the space reserved. This increase is performed if you wish to add more data to the file after the original allocation of space is used up. You may allocate different types of files, depending on the hardware you use. *You may not allocate spooled files or tape files with this command. You may only allocate disk or diskette files.*

If you are operating Series 90 hardware, you can allocate MIRAM, ISAM, IRAM, DAM, SAT, SAM, and nonindexed files. *However, interactive services commands work only with MIRAM and SAT files.* For example, you can use the ALLOCATE command to allocate an ISAM file, but after allocation you cannot use an interactive command like COPY or PRINT on that file. All diskettes on Series 90 hardware are recorded in data-set-label (DSL) mode. With this mode, the system will always allocate a sequential, card-image file. Enter MI as the file type in this case. For DSL diskettes, give the INC= and SIZE= parameters in blocks.

For System 80 hardware, you can allocate only MIRAM and SAT files. Data files, menu, HELP, saved, and screen format modules are allocated as MIRAM files, and program library files are allocated as SAT files. If you are using diskettes on a System 80, they can be recorded in either DSL or format label (FL) mode. For DSL diskettes, enter MI as the file type; give the SIZE= parameter in blocks. *DSL diskette files cannot be extended; therefore, any INC= specification is ignored.* FL diskettes are accessed in the same way as disks. Therefore, they can be allocated as either MIRAM (MI, ML) or SAT (ST) files. The INC= and SIZE= parameters are given in cylinders.

NOTE:

To allocate DSL diskettes as basic data exchange (BDE), or to allocate a file with an expiration date, use the // EXT job control statement, not the ALLOCATE command.

The ALLOCATE command issues a message to you, informing you when it has completed processing. You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish: just place a dash at the end of every line except the line on which the parameter string ends.

Format:

```

ALLOCATE△(ST),FILENAME={filename} [,RDPASS=password][,WRPASS=password]
           {IR}
           {IS}
           {DA}
           {SQ}
           {NI}
           {MI}
           ,VSN=volume [ ,CONTIG={YES} ] [ ,INC={n} ],SIZE=n
                        {NO}
  
```

Command Option:

```

(ST)
(IR)
(IS)
(DA)
(SQ)
(NI)
(MI)
  
```

Specifies the type of file you wish to allocate. Enter ST for SAT library files, IR for IRAM files, IS for ISAM files, DA for DAM files, SQ for sequential files, NI for nonindexed files, and MI for MIRAM data files.

Example:

```
AL MI,FILENAME=MIRAMFILE,VSN=PACK01,INC=5,SIZE=10
```

In this example, a MIRAM file named MIRAMFILE is to be allocated. The file is to be allocated on the volume that has a volume serial number of PACK01. Initially, 10 cylinders of disk space are allocated to the file; as more space is required, it is allocated in increments of 5 cylinders.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- File Already Allocated

A file with the same name as the file you are attempting to allocate has already been allocated.

- No Room for File

There is not enough space left on the volume you specified for the size file you wish to allocate.

- Invalid Password

The password you entered is not valid.

COM**4.4.4. Adding and Modifying Library Module Comments (COMMENT Command)**

The COMMENT command enables you to add a comment to a library module header or to replace an existing comment. You must allow a single space between the last file parameter and the beginning of your comment text. Anything following that space is considered part of the comment by the system. Comments may be up to 30 characters long. Longer comment texts are truncated to 30 characters. The COMMENT command issues a message to you, informing you when it has completed processing.

To find out what, if any, comments have been added to a module, use the FSTATUS command with the LONG parameter (4.4.11) or use the COP librarian statement (see the system service programs user guide).

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:

```
COMMENT△MODULE=modulename [ ,TYPE={module-type} ],FILENAME={filename
                             {s}                               'filename'
                             "filename"
                             }
                             [,RDPASS=password][,WRPASS=password],VSN=volume△text
```

Command Options:

text

Specifies the text of the comment you wish to insert, or the text you wish to insert in place of the existing comment. It may be up to 30 characters long. *A single space must separate the text from the file parameters.*

Example:

```
COM MODULE=PARTIME,FILENAME=PAYFIL PART TIME WORKERS
```

In this example, the comment PART TIME WORKERS is being added to the module named PARTIME, located in the file named PAYFIL. Since no module type is specified, the module is a source module. No password is needed to access this file, and since no volume serial number is included among the file parameters, it may be assumed to be included in the file catalog entry for the file containing the module commented.

The following conditions could cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were entered improperly.

- **File, Module, or Volume Not Available**

The file, module, or volume you requested is not available for your access.

- **Invalid Password**

The password you entered is not valid.

COP**4.4.5. Copying Files (COPY Command)**

The COPY command copies modules or files. You can also use it to alter the format of a file. You can copy one type of file to another type. For example, you can make a spooled file copy of a MIRAM file or a tape file copy of a spooled file. You can use the COPY command with these types of files: MIRAM, library module, spool, tape, and unit record files. The different parameters required for each type of file are shown in the following formats.

To copy one type of file to another type, use the appropriate input and output parameter strings. For example, to copy a MIRAM file to a spool file, use the input parameter string in format 2 (for MIRAM files) and the output parameter string from format 3 (for spool files). (The input parameter string is everything preceding the word TO; the output parameter string is everything following it.) You must place the delimiter TO between the input and output parameter strings. The word TO must be separated from the input and output parameter strings by single spaces. The command will be rejected if the spaces are omitted.

The COPY command runs as a background task. You can enter other commands while the COPY command is executing. You will receive a message when the COPY command is finished.

You can use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You can use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

The COPY command is divided into the following five formats:

Format 1, Copying SAT or MIRAM Library Modules:

```

COPY△MODULE=modulename [ ,TYPE={module-type} ] ,FILENAME={ filename
                                     { 'filename'
                                     { "filename"
[ ,RDPASS=password ], VSN=volume△TO△MODULE=modulename [ ,TYPE={module-type}
                                     { 'filename'
                                     { "filename"
, FILENAME={ filename } [ ,RCSZ=n ] [ ,WRPASS=password ], VSN=volume
                                     { 'filename'
                                     { "filename"
[ ,CONTIG={ YES } ] [ ,INC={ n } ] [ ,SIZE=n ] [ ,SAT={ YES } ] \[ NUMBER ] [ ,HEX ] [ ,WAIT ]
                                     { NO } ] [ ,INC={ n } ] [ ,SIZE=n ] [ ,SAT={ NO } ]

```

NOTE:

There are two modules for every screen format: an F type module and an FC type module. To copy a screen format, you must copy both modules in two separate COPY operations. Remember to specify the file type by using the TYPE= parameter. The rest of the file parameters remain the same for both modules.

Command Options:**NUMBER**

Specifies that each record is to be prefaced with a record number in the copy made of the file.

NOTE:

This option makes each record 10 characters longer.

HEX

Specifies that the copy made of the file will have its records translated into their printable hexadecimal equivalent.

NOTE:

This option doubles the size of each record.

WAIT

Specifies that control is not returned to WORKSTATION mode until the COPY command has completed execution. Normally, when you issue a COPY command, a background task will be created, and the COPY command will execute as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the COPY operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the COPY command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 2, Copying MIRAM Data Files:

```

COPY△FILENAME={filename } [,RDPASS=password],VSN=volume [,KEYNO={n}]
                {'filename'}
                {"filename"}

[ ,DEVICE={did
            {DISK
            DISKETTE}}]
△TO△FILENAME={filename } [,WRPASS=password],VSN=volume
                {'filename'}
                {"filename"}

[ ,CONTIG={YES} ] [ ,INC={n} ] [ ,KEYNO={n} ] [ ,KEYi={n:m
            (n:m, {DUP}, {CHG})
            {NDUP} {NCHG}} ]
[ ,SIZE=n ] [ ,INIT={YES} ] [ ,EXTEND={YES} ]
            {NO} {NO}
[ ,RCB={YES} ] [ ,RCFM={FIX} ] [ ,RCSZ=n ] [ ,SCSZ={n} ] [ ,DEVICE={did
            {NO} {VAR} {256} {DISK
            DISKETTE}}]

[ ,BFSZ=n ] △[NUMBER] [ ,HEX ] [ ,WAIT ]

```

Command Options:

NUMBER

Specifies that each record is to be prefaced with a record number in the copy made of the file.

NOTE:

This option makes each record 10 characters longer.

HEX

Specifies that the copy made of the file will have its records translated into their printable hexadecimal equivalent.

NOTE:

This option doubles the size of each record.

WAIT

Specifies that control is not returned to WORKSTATION mode until the COPY command has completed execution. Normally, when you issue a COPY command, a background task will be created, and the COPY command will execute as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the COPY operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the COPY command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 3, Copying Spool Files:

```

COPYΔ[JOB=jobname] [ ,HOLD={ ! } ] [ ,FILENAME={ filename } ] [ ,ACCT=acct ] ,QUEUE={ LOG
→                                     { PRINT }
→                                     { PUNCH }
→                                     { RDR } ]
[ ,ALL={ YES } ] [ ,SKIP={ n } ] ΔTOΔ[JOB=jobname] [ ,HOLD={ N } ] [ ,FILENAME={ filename
→                                     { 'filename' }
→                                     { "filename" } ]
,QUEUE={ PRINT } Δ[NUMBER][ ,HEX ][ ,WAIT ]
→                                     { PUNCH }
→                                     { RDR }

```

Command Options:**NUMBER**

Specifies that each record is to be prefaced with a record number in the copy made of the file.

NOTE:

This option makes each record 10 characters longer.

HEX

Specifies that the copy made of the file will have its records translated into their printable hexadecimal equivalent.

NOTE:

This option doubles the size of each record.

WAIT

Specifies that control is not returned to WORKSTATION mode until the COPY command has completed execution. Normally, when you issue a COPY command, a background task will be created, and the COPY command will execute as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the COPY operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the COPY command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 4, Copying Tape Files:

```

COPY△ [ FILENAME={filename } ] [ , RDPASS=password ] , VSN=volume , DEVICE={did }
      [ 'filename' ]
      [ "filename" ]
      [ , BKNO={YES } ] △TO△
      [ NO ]
      [ FILENAME={filename } ] [ , WRPASS=password ] , VSN=volume , DEVICE={did }
      [ 'filename' ]
      [ "filename" ]
      [ , INIT={YES } ] [ , EXTEND={YES } ]
      [ NO ] [ NO ]
      [ , BFSZ=n ] [ , BKNO={YES } ] [ , RCFM={FIXUNB } ] [ , RCSZ=n ] △[NUMBER] [ , HEX ] [ , WAIT ]
      [ NO ]
      [ FIXBLK ]
      [ VARUNB ]
      [ VARBLK ]
      [ UNDEF ]

```

Command Options:**NUMBER**

Specifies that each record is to be prefaced with a record number in the copy made of the file.

NOTE:

This option makes each record 10 characters longer.

HEX

Specifies that the copy made of the file will have its records translated into their printable hexadecimal equivalent.

NOTE:

This option doubles the size of each record.

WAIT

Specifies that control is not returned to WORKSTATION mode until the COPY command has completed execution. Normally, when you issue a COPY command, a background task will be created, and the COPY command will execute as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the COPY operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the COPY command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 5, Copying Unit Record Files:

```

COPY△DEVICE={did          },FILENAME={filename  },VSN=volume
             {DISKETTE    }           {'filename' }
             {RDR         }           {"filename" }

△TO△DEVICE={did          }[,RCFM={FIX }][,RCSZ=n]
            {DISKETTE    }
            {PRINT       }
            {PUNCH       }

,FILENAME={filename      },VSN=volume△[NUMBER][,HEX][,WAIT]
          {'filename'    }
          {"filename"   }

```

NOTE:

The FILENAME= and VSN= parameters are required only when using a diskette.

Command Options:**NUMBER**

Specifies that each record is to be prefaced with a record number in the copy made of the file.

NOTE:

This option makes each record 10 characters longer.

HEX

Specifies that the copy made of the file will have its records translated into their printable hexadecimal equivalent.

NOTE:

This option doubles the size of each record.

WAIT

Specifies that control is not returned to WORKSTATION mode until the COPY command has completed execution. Normally, when you issue a COPY command, a background task will be created, and the COPY command will execute as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the COPY operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the COPY command causes WORKSTATION mode to be stopped until the command has completed execution.

Example:

```
COP MODULE=SYSDUMP,FILENAME=$Y$JCS TO FILENAME=JCL,QUEUE=RDR,HOLD=N
```

In this example, the module named SYSDUMP, located in the file named \$Y\$JCS, is to be copied to a spool file name JCL, located on the RDR (reader) spool queue. The file is *not* specified to be retained after processing.

The following conditions could cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- **File, Module, or Volume Not Available**

The file, module, or volume you wish to copy is unavailable for your use.

- **Invalid Password**

The password or passwords you entered are not valid.

DEFKEY

4.4.6. Assigning Commands to Function Keys (DEFKEY Command)

The DEFKEY command allows you to assign any interactive command to a function key or to the MESSAGE WAITING key. (See Appendix A for the location of the 22 function keys and the MESSAGE WAITING key on your keyboard.) Once you've assigned a key, you simply have to press it to execute the command associated with it. Your workstation can be in data or system mode when the function key is pressed.

You can include function key assignments in your execution profile. You should note, however, that when you assign a function key, it is only valid during your workstation session and only for your user-id. Another user with a different ID does not get the same results by pressing the function key you've assigned a command to. To assign function keys for other users, you can include the function key assignments in their execution profiles.

To free a function key from a command string, see the DEFKEY (delete) command (4.4.7). To simply assign a new command string to an already assigned function key, enter the DEFKEY command with the new command string. This overrides the previous assignment for that key.

NOTE:

You must take care not to redefine keys used by interactive services or any of the interactive facilities. If you do reassign them, you will lose the functions they provide. These function keys are listed in Appendix E.

For SCREEN FORMAT SERVICES USERS: Be aware that screen format services (SFS) also allows you to assign function keys that are used by a program. If you assign a function key already used by your SFS program, the DEFKEY assignment takes precedence and you lose the SFS function key capability.

Format:

```
DEFKEY△ {F#nn} , { 'command string' }
        {MW}  { "command string" }
```

where:

nn

Is the 1- or 2-digit number of the function key you're assigning a command to.

MW

Stands for the MESSAGE WAITING key.

{ 'command string' }
{ "command string" }

Is the actual interactive command string. It must be enclosed in either single or double quotes and can be up to 60 characters long.

NOTE:

The DEFKEY command does not check the syntax of your command string. If you make an error in the command string, it won't show up until you actually use the function key associated with it. At that time, you will get an error message that an illegal command has been entered.

Examples:

```
DEFKEY F#2, 'BR LO'
```

In this example, the breakpoint log command is assigned to function key 2. When you press this key, the workstation log is closed and is made available to the output writer for printing.

```
DEFKEY MW, "LOGOFF"
```

In this example, the LOGOFF command is assigned to the MESSAGE WAITING key. When you press this key, the LOGOFF command automatically executes.

The following conditions could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were entered improperly.

DEFKEY (delete)

4.4.7. Deleting Function Key Assignments (DEFKEY Delete Command)

To free a function key or the MESSAGE WAITING key from a command assignment, you use the delete form of the DEFKEY command. The delete DEFKEY command is exactly like the DEFKEY command, minus the command string. Once you delete a command assignment from a function key you can reassign a new command to it. (To view a list of the function keys assigned under your user-id, use the DEFKEY DISPLAY command 4.4.8.)

The format of the delete DEFKEY command is:

```
DEFKEY△{F#nn}
      {MW }
```

where:

nn

Is the number of the function key (from 1 to 22) that you previously assigned a command to.

MW

Is the MESSAGE WAITING key.

Example:

```
DEFKEY F#5
```

In this example, function key 5 is no longer associated with the command string previously assigned to it.

```
DEFKEY MW
```

In this example, the MESSAGE WAITING key is freed from the command previously assigned to it.

The following conditions could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were entered incorrectly.

DEFKEY DISPLAY

4.4.8. Displaying Function Key Assignments (DEFKEY DISPLAY Command)

The DEFKEY DISPLAY command displays all the function keys and commands assigned under your user-id during a workstation session. (See the DEFKEY command, 4.4.6.) If function key assignments were made a part of your execution profile, this is a very useful means of finding out what commands the function keys will execute.

Format:

```
DEFKEY△DISPLAY
```

The format of the function key display that appears on your workstation screen is:

```
F#nn=command string
```

where:

nn

Is the function key number (from 1 to 22).

command string

Is the actual interactive command assigned to that key.

A sample function key display is shown below:

```
DEFKEY DISPLAY
0A F#06=BR LO
0B F#07=EDT
0C F#08=LOGOFF
0D F#09=STATUS
```

The following conditions could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were entered improperly.

To Run a Command Stream from a Spooled File:

$$\text{ENTER}\Delta \left[\text{HOLD}=\begin{cases} \text{N} \\ \text{Y} \end{cases} \right], \text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases}, \text{QUEUE}=\text{RDR}$$

To Run a Command Stream from a DSL Diskette:

$$\text{ENTER}\Delta \left[\text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases} \right] [,\text{RDPASS}=\text{password}], \text{VSN}=\text{volume}$$

$$\left[,\text{DEVICE}=\begin{cases} \text{did} \\ \text{DISKETTE} \end{cases} \right]$$

To Run a Command Stream from a Tape:

$$\text{ENTER}\Delta \left[\text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases} \right] [,\text{RDPASS}=\text{password}], \text{VSN}=\text{volume} \left[,\text{DEVICE}=\begin{cases} \text{did} \\ \text{TAPE} \end{cases} \right]$$

To Run a Command Stream from a Card Reader:

$$\text{ENTER}\Delta \text{DEVICE}=\begin{cases} \text{did} \\ \text{RDR} \end{cases}$$

Example:

```
ENT MODULE=INJOB,TYPE=S,FILENAME=JOBFILE,RDPASS=SEEK,VSN=REL070
```

In this example, a stream of workstation commands is to be read from a library module and executed. The module name is INJOB, and it is a source module. It is located in the library file named JOBFILE. To access JOBFILE, a password, SEEK, is entered. JOBFILE is located on a disk volume with the volume serial number REL070. The following is a typical stream of workstation commands. Note that it invokes the general editor and performs four editing commands.

```
LOGON JOB6,A263
EDT
@READ MYFILE,DISK02
@CHANGE 'OPWN' TO 'OPEN'
@WRITE
@HALT
PRINT MYFILE,DISK02 NUMBER
LOGOFF
```

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- First Command Not LOGON

The first command of any stream using the ENTER command *must* be the LOGON command. If you do not have a LOGON command as the first card of your enter stream, the system will reject your ENTER command.

- Data Cards Interpreted as System Commands

This could happen if, for example, you have a LOGON card, an EDT card, and then a series of EDT commands. If the EDT card is not read for some reason, the system will attempt to interpret the EDT commands as system commands and reject each one it reads and misinterprets.

- Cards in the Stream after the LOGOFF Command

Cards in the stream after a LOGOFF command are not read. The system stops reading immediately upon reading the LOGOFF command.

ER

4.4.10. Deleting Library Files and Modules and MIRAM Files (ERASE Command)

The ERASE command permits you to delete library and data files, and library file modules. The ERASE command deletes library modules by marking them as deleted. The ERASE command deletes files by removing them from the volume table of contents (VTOC) for the volume on which they reside. When you enter the command to erase a whole file, the system returns the following message to you before executing the command:

```
IS51 ERASING ENTIRE FILE, PROCEED? (Y,N)
```

This helps prevent unintentional deletion of important files. You must enter Y before the system can go ahead and delete the file. The ERASE command issues a message to you when it has completed execution.

Since the ERASE command only deletes files or modules *logically* by removing them from the volume table of contents (VTOC), they still physically exist. In the event you accidentally erase a source, procedure, or macro module in a SAT library, use the RECOVER command (4.4.17) to recover the module. The RECOVER command is effective up to the time a LIBS PAC operation is performed. Once a LIBS PAC is completed, the file or module is permanently lost, logically and physically. MIRAM files cannot be recovered.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:

To Erase a Single Library Module:

```
ERASE△MODULE=modulename [ ,TYPE={module-type} ] ,FILENAME={filename }
                                     { 'filename' }
                                     { "filename" }
                                     [ ,WRPASS=password ], VSN=volume
```

To Erase Library and MIRAM Files:

```
ERASE△FILENAME={filename } [ ,WRPASS=password ], VSN=volume
                 { 'filename' }
                 { "filename" }
```

NOTE:

To erase a spool file, you must use the DELETE SPL command, found in 4.3.9.

Example:

```
ER MODULE=PAYMOD,FILENAME=$Y$JCS,VSN=RES
```

In this example, the module named PAYMOD, which is part of the file \$Y\$JCS, is to be erased. The file \$Y\$JCS resides on the volume with the volume serial number RES and is a source module.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- File or Module Not Found

The system is unable to locate the file or module you requested.

- Volume Not Available

The volume containing the file you requested is not available.

- Invalid Password

The password you entered is invalid.

FS

4.4.11. Obtaining File Status Information (FSTATUS Command)

You use the FSTATUS command to obtain information about files and their contents. You may obtain information about the modules contained in a library file. When you reference a library file using this command, you may select either a short or long display of module information. The short display gives a list of active modules and what type of module each is. The long display lists each module with its name, type, comment (if one is present), and the date and time the module was created.

NOTE:

Only one FSTATUS command may be processed at a time. You may not enter another FSTATUS until the command currently being processed is completed.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:**For Library Files:**

```
FSTATUS△[MODULE=modulename] [ ,TYPE={module-type} ] ,FILENAME={ filename
                                     {s}
                                     'filename'
                                     "filename"
                                     }
[ ,RDPASS=password ], VSN=volume△[LONG]
```

Command Options:**NOTE:**

With the FSTATUS command you can use the module name and module type as qualifiers to produce displays of groups of modules.

To use the module name parameter as a qualifier, enter the characters you wish to use to qualify the modules to be displayed, and enter a period (.) as the last character of the module name parameter. The command will then produce a display of all modules whose names begin with the characters you specified. For example, entering ABC. will cause all modules whose names begin with the letters ABC to be displayed. Entering the letters ABC without a period, however, will cause the display of only those modules named ABC.

To use the module type parameter as a qualifier, enter the character or characters you wish to use to qualify the modules to be displayed, and enter a period (.) as the last character of the module type parameter. The command will then produce a display of all modules whose type begins with the character or characters you specified. For example, if you enter F. as the module type, you will receive a display of all modules of type F, or whose type begins with an F. Entering only F with no period will cause only type F modules to be displayed.

LONG

Specifies that you wish to see the long version of the FSTATUS display for library files. If you do not enter LONG, the short version will be displayed by default.

Example:

If you wanted a short display of the modules contained in the library file \$Y\$JCS, residing on volume RELO60, you would enter:

```
FS FILENAME=$Y$JCS,VSN=REL060
```

You would then see the following display:

```
FSTATUS      $Y$JCS,REL060
P-J$CPYLBO   P-J$COBOL   P-J$FORT     P-J$FOR
P-J$LINK     P-UTSLIN    S-SYSDUMP    S-COR#V
P-CORRUN     S-SU$PAT
```

Entering the LONG parameter produces the following display:

```
FSTATUS J, $Y$JCS,PACK75  LONG
P-J$CPYLBO  EMUL AID 8410  DISC COPY 11/28/79 09:27
P-J$COBOL   COBOL CANNED  JPROC      11/28/79 09:28
P-J$FORT    FORTRAN CANNED JPROC      11/28/79 09:28
```

The following conditions could cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- File or Volume Not Available


The file or volume you wish to access is not available.

- Invalid Password

The password you entered to read the file is not valid. You are denied access to the file.

- Attempt to Reference a Spool File

The FSTATUS command cannot be used to obtain information about a spool file. Only library files may be referenced.



HE

4.4.12. Obtaining Command and Message Assistance (HELP Command)

You use the HELP command to obtain information on how to use other workstation commands and how to handle error messages sent to you by the system. When you enter HELP for a command, you receive a screen display explaining the command and its use. When you enter HELP for a message, you receive a screen display explaining the message and how to respond to it.

Format:

→ `HELPΔ { command
 message-no
 keyword-parameter }`

Parameters:

command

Specifies the command you need help with. You must enter the command in its entirety; you may not abbreviate it.

message-no

Specifies the prefix of the message you need help with. The message prefix can contain both letters and numbers, as in the case of messages generated by interactive services. Interactive services messages are prefixed with ISnn, where nn is a 2-digit number. All message numbers appear in the message manual.

NOTE:

System error codes contain only numbers. To get help for a system error code, enter HELP EError-code, where error-code must be three digits long, with a leading zero if necessary. Thus, to get help for error code 051, you enter HELP EC051.

keyword-parameter

Specifies a keyword parameter used in a command. If you need help with a keyword parameter, you must enter the keyword, for example, HELP FILENAME, HELP MODULE, or HELP VSN. You must enter the parameter in its entirety; you may not abbreviate it.

NOTE:

You can receive help for many commands, messages, and keyword parameters; if, however, you enter a command, message, or keyword parameter for which help is not yet available, you receive the following message:

↑ IS94 THE HELP YOU REQUESTED IS NOT AVAILABLE, SORRY.

Examples:

Help for a message:

```

help is34
IS34 ILLEGAL COMMAND HAS BEEN ENTERED, IGNORED
YOU ENTERED A BAD COMMAND; PERMITTED COMMANDS ARE:
ALLOCATE CONNECT FILE LOGOFF RECOVER SDU
ASK COPY FREE MENU REMARK SMU
BASIC DDP FSTATUS OCL RESUME STATUS
BEGIN DEFKEY GO OV RUN STOP
BRKPT DELETE HELP PAUSE RV TELL
CANCEL DISPLAY HOLD PRINT SCREEN VTOC
CHANGE EDT IN PUNCH SCHED $$$ON
COMMENT ENTER LOGON REBUILD SI $$$OF
ERASE
IS90 HELP COMMAND TERMINATED NORMALLY

```

The top of the screen shows the HELP command entered for message IS34. The next 10 lines of the screen are the HELP display, explaining the message and giving you a list of permitted commands. The final line of the display shows that the HELP command has completed. Note that the command is entered in lowercase letters; as with all commands, you may enter it in either uppercase or lowercase.

Help for a command:

```

HELP ALLOCATE
THE ALLOCATE COMMAND IS USED TO ADD A NEW FILE TO A DISK OR
DISKETTE. ITS FORMAT IS:
ALLOCATE EXT-TYPE,FILENAME,VSN,SIZE=N,INC=M
WHERE: EXT-TYPE IS A TWO CHARACTER EXTENT TYPE. VALUES ARE
MI, ST, ML, IS, DA, SQ, NI
FILENAME IS THE 1 TO 44 CHARACTER NAME TO BE GIVEN
TO THE NEW FILE.
VSN IS THE VOLUME SEQUENCE NUMBER OF THE DISK ON
WHICH THE NEW FILE IS TO RESIDE.
SIZE IS THE NUMBER OF DISK CYLINDERS OR DISKETTE
BLOCKS WHICH ARE TO BE ALLOCATED TO THE
FILE.
INC IS THE NUMBER OF DISK CYLINDERS OR DISKETTE
BLOCKS TO BE ALLOCATED IF THE FILE MUST BE
EXTENDED.
IS90 HELP COMMAND TERMINATED NORMALLY

```

The first line of the screen display shows the HELP command entered for the ALLOCATE command. The next 15 lines are the HELP display, explaining the function of the command, its format, and the parameters entered with it. The final line of the display shows that the HELP command has completed.

↓
Help for a keyword parameter:

```
help filename
FILENAME= FILENAME/'FILENAME'/'FILENAME''
SPECIFIES THE NAME OF THE FILE BEING REFERENCED. IN ALL
CASES EXCEPT TWO, THE FILE NAME MUST BE THAT OF AN EXISTING
FILE. THE EXCEPTIONS ARE FILE NAMES ENTERED WITH THE
ALLOCATE COMMAND AND FILE NAMES ENTERED FOR FILES YOU ARE
CREATING THROUGH THE COPY COMMAND. IN BOTH CASES THE NAMES
MUST BE NAMES YOU CREATE. PHYSICAL FILE NAMES MAY BE FROM
1 TO 44 ALPHANUMERIC CHARACTERS LONG. LOGICAL FILE NAMES
(FOR SPOOL FILES) MAY BE FROM 1 TO 17 CHARACTERS LONG.
TAPE FILE NAMES MAY ALSO BE FROM 1 TO 17 CHARACTERS LONG.
YOU MUST ENCLOSE THE FILE NAME IN EITHER QUOTATION MARKS OR
APOSTROPHES IF THERE ARE SPACES, COMMAS, OR PARENTHESES
EMBEDDED WITHIN THE FILE NAME.
IS90 HELP          COMMAND TERMINATED NORMALLY
```

↑
The top of the screen shows the HELP command entered for the file name parameter. The next 14 lines are the HELP display, explaining the keyword parameter and showing its format. The final line of the display shows that the HELP command has completed.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- Invalid Parameters

You entered the command with no parameters, or with something other than a command or message prefix. You may only enter either a message prefix or a command with the HELP command.

MENU ↓**4.4.13. Calling Menus (MENU Command)**

Menus are an OS/3 feature designed to simplify certain everyday operator tasks. A menu is a list of items from which a workstation operator chooses. Usually these represent workstation commands, programs, or functions within programs. Menu items are numbered so that when the system displays a menu on your workstation, you select an item by entering its item number in the blanks provided on the menu screen and pressing the transmit key. There are two types of menus: those supplied by Sperry Univac, called system menus, and those you create, called user-created menus. You create menus using menu services; Section 5 briefly outlines how menu services work.

Format:

```
MENU△ [menu-name] [ , { filename, vsn } ] [ , { SYSFMT [ , { RES } ] } ] ]
```

Parameters:

menu-name

Is the name of a user-created menu you want to call. If you don't specify a menu name, menu services automatically call a standard system menu (included with your system) that is designed to acquaint new operators with OS/3. For more information on system-supplied menus, see 2.4.

file-name

Is the name of the MIRAM file in which your menu resides. The default is the system format file, \$Y\$FMT.

vsn

Is the volume serial number of the volume containing your menu file. The default value, the system resident volume, RES, applies only if the filename parameter is \$Y\$FMT. For any other filename, you must specify this parameter even if it is RES.

Examples:

MENU



You simply select a function, enter the number associated with that function on line 8, and transmit the screen. If you need help with any function, enter a '?' before the function number. Or, to get help with the entire screen, simply enter a question mark. Menu services then display a help screen associated with that function or menu.

The following conditions could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

PRI

4.4.14. Making Printed Copies of Files (PRINT Command)

The PRINT command is used to make a printed copy of a SAT or MIRAM library module, a MIRAM data file, a spool file, tape file, or unit record file. You may direct the output of this command to either an actual printer or a spool printer. The printed copy produced by this command includes a header page containing identification information.

The PRINT command executes as a background job; that is, it is executed in such a way as to permit you to continue entering commands while it is executing. The PRINT command issues you a message when it has completed execution.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

The PRINT command is divided into the following five formats:

Format 1, Printing SAT and MIRAM Library Modules:

```
PRINT△MODULE=modulename [ , IYPE={ module-type } ] , FILENAME= { filename
                        { 'filename'
                        "filename" }
[ , RDPASS=password ] , VSN=volume [ , COPIES={ n } ] △ [ DIRECT ] [ , NUMBER ] [ , HEX ] [ , WAIT ]
```

Command Options:

DIRECT

Specifies that an actual printer be assigned to print the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file and printed when a printer becomes available. When DIRECT is specified, the file is not spooled and is printed immediately if a printer is available. If DIRECT is specified and no printer is available, you will receive an error message at the workstation. The default value is that the file be spooled.

NUMBER

Specifies that line numbers are to be included with the printout.

HEX

Specifies that the file be translated into its printable hexadecimal equivalent before being printed.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PRINT command has completed execution. Normally, when you issue a PRINT command, a background task is created, and the PRINT command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PRINT operation has been completed successfully before going on with your program. In this case, entering the WAIT keyword with the PRINT command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 2, Printing MIRAM Data Files:

```
PRINT△FILENAME= { filename } [ ,RDPASS=password ] ,VSN=volume [ ,KEYNO={ n } ]
                  { 'filename' }
                  { "filename" }
[ ,COPIES={ n } ] [ ,DEVICE={ did } ] △[DIRECT][ ,NUMBER ][ ,HEX ][ ,WAIT ]
                  { DISKETTE }
                  { DISK }
```

Command Options:

DIRECT

Specifies that an actual printer be assigned to print the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file and printed when a printer becomes available. When DIRECT is specified, the file is not spooled and is printed immediately if a printer is available. If DIRECT is specified and no printer is available, you will receive an error message at the workstation. The default value is that the file be spooled.

NUMBER

Specifies that line numbers are to be included with the printout.

HEX

Specifies that the file be translated into its printable hexadecimal equivalent before being printed.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PRINT command has completed execution. Normally, when you issue a PRINT command, a background task is created, and the PRINT command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PRINT operation has been completed successfully before going on with your program. In this case, entering the WAIT keyword with the PRINT command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 3, Printing a Spool File:

\rightarrow `PRINT` Δ [`JOB`=jobname] [`HOLD`= $\left\{ \begin{array}{l} L \\ N \\ Y \end{array} \right\}$] [`FILENAME`= $\left\{ \begin{array}{l} \text{filename} \\ 'filename' \\ "filename" \end{array} \right\}$] [`ACCT`=acct]
 $\left\{ \begin{array}{l} \text{LOG} \\ \text{PRINT} \\ \text{PUNCH} \\ \text{RDR} \end{array} \right\}$ [`ALL`= $\left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right\}$] [`COPIES`= $\{n\}$] [`SKIP`= $\{n\}$] Δ [`DIRECT`] [, `NUMBER`]
 [, `HEX`] [, `WAIT`]

Command Options:

DIRECT

Specifies that an actual printer be assigned to print the file requested in the command. When **DIRECT** is not specified, the file requested will be copied to a spool file and printed when a printer becomes available. When **DIRECT** is specified, the file is not spooled and is printed immediately if a printer is available. If **DIRECT** is specified and no printer is available, you will receive an error message at the workstation. The default value is that the file be spooled.

NUMBER

Specifies that line numbers are to be included with the printout.

HEX

Specifies that the file be translated into its printable hexadecimal equivalent before being printed.

WAIT

Specifies that control is not returned to **WORKSTATION** mode until the **PRINT** command has completed execution. Normally, when you issue a **PRINT** command, a background task is created, and the **PRINT** command executes as a background job, allowing you to continue to use **WORKSTATION** mode for user or system programs. However, there may be times when you want to be sure the **PRINT** operation has been completed successfully before going on with your program. In this case, entering the **WAIT** keyword with the **PRINT** command causes **WORKSTATION** mode to be stopped until the command has completed execution.

Format 4, Printing a Tape File:

`PRINT` Δ [`FILENAME`= $\left\{ \begin{array}{l} \text{filename} \\ 'filename' \\ "filename" \end{array} \right\}$] [, `RDPASS`=password], `VSN`=volume, `DEVICE`= $\left\{ \begin{array}{l} \text{did} \\ \text{TAPE} \end{array} \right\}$
 [, `BKNO`= $\left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right\}$] Δ [`DIRECT`] [, `NUMBER`] [, `HEX`] [, `WAIT`]

Command Options:

DIRECT

Specifies that an actual printer be assigned to print the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file and printed when a printer becomes available. When DIRECT is specified, the file is not spooled and is printed immediately if a printer is available. If DIRECT is specified and no printer is available, you will receive an error message at the workstation. The default value is that the file be spooled.

NUMBER

Specifies that line numbers are to be included with the printout.

HEX

Specifies that the file be translated into its printable hexadecimal equivalent before being printed.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PRINT command has completed execution. Normally, when you issue a PRINT command, a background task is created, and the PRINT command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PRINT operation has been completed successfully before going on with your program. In this case, entering the WAIT keyword with the PRINT command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 5, Printing a Unit Record File:

$$\text{PRINT}\Delta\text{DEVICE}=\left\{\begin{array}{l} \text{did} \\ \text{DISKETTE} \\ \text{RDR} \end{array}\right\}, \text{FILENAME}=\left\{\begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array}\right\}, \text{VSN}=\text{volume}$$

$\Delta[\text{DIRECT}][, \text{NUMBER}][, \text{HEX}][, \text{WAIT}]$

NOTE:

The FILENAME= and VSN= parameters are required only when using a diskette.

Command Options:

DIRECT

Specifies that an actual printer be assigned to print the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file and printed when a printer becomes available. When DIRECT is specified, the file is not spooled and is printed immediately if a printer is available. If DIRECT is specified and no printer is available, you will receive an error message at the workstation. The default value is that the file be spooled.

NUMBER

Specifies that line numbers are to be included with the printout.

HEX

Specifies that the file be translated into its printable hexadecimal equivalent before being printed.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PRINT command has completed execution. Normally, when you issue a PRINT command, a background task is created, and the PRINT command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PRINT operation has been completed successfully before going on with your program. In this case, entering the WAIT keyword with the PRINT command causes WORKSTATION mode to be stopped until the command has completed execution.

Examples:

```
PRI MODULE=PIB,FILENAME=$Y$SRC NUMBER
```

In this example, the source module PIB, located in the file named \$Y\$SRC is printed. The NUMBER command option is included, so the printed output will have line numbers added to it.

```
PRI FILENAME=LONGMIRAMFILE,VSN=PACK23
```

In this example, a file named LONGMIRAMFILE, located on volume PACK23, is printed.

The following conditions could cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- **File or Volume Not Available**

The file or volume you requested is not available for your use.

- Spooling Not Configured

The spooling process was not configured at the time your system was generated; therefore, no spool files exist to be accessed.

- Printer Not Available

If you entered the DIRECT option with this command and no actual printer was available to print the file you requested, the system will give you an error message and tell you to enter the command again.

- Invalid Password

The password you entered is invalid.

PUN

4.4.15. Making Punched Card Copies of a File (PUNCH Command)

The PUNCH command enables you to make punched card copies of library modules, MIRAM data files, and spooled files. Output from the PUNCH command may be directed to either an actual or spooled punch. The command is executed in such a way to permit you to continue entering commands while the PUNCH command is being executed. The PUNCH command issues you a message when it has completed execution.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

NOTE:

The PUNCH command punches only fixed-length, 80-character records. If your file has records with more than 80 characters and you use the PUNCH command, the records are truncated.

The PUNCH command is divided into the following five formats:

Format 1, Punching a Library Module:

```
PUNCH△MODULE=modulename [ ,TYPE={ module-type } ] [ ,FILENAME={ filename
                                     { 'filename' }
                                     "filename" } ]
[ ,RDPASS ] ,VSN=volume [ ,COPIES={ n } ] [ DIRECT ] [ ,WAIT ]
```


Command Options:

DIRECT

Specifies that an actual punch is to be assigned to punch the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file. Then, when a punch becomes available, the file will be punched. If DIRECT is specified and no punch is available, you will receive an error message at the workstation. The default value is to spool the file.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PUNCH command has completed execution. Normally, when you issue a PUNCH command, a background task is created, and the PUNCH command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PUNCH operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the PUNCH command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 2, Punching a MIRAM Data File:

$$\text{PUNCH}\Delta\text{FILENAME}=\left\{\begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array}\right\} [, \text{RDPASS}=\text{password}], \text{VSN}=\text{volume} \left[, \text{KEYNO}=\left\{\begin{array}{l} n \\ \text{[]} \end{array}\right\} \right]$$

$$\left[, \text{COPIES}=\left\{\begin{array}{l} n \\ \text{[]} \end{array}\right\} \right] \left[, \text{DEVICE}=\left\{\begin{array}{l} \text{did} \\ \text{DISKETTE} \\ \text{DISK} \end{array}\right\} \right] \Delta[\text{DIRECT}] [, \text{WAIT}]$$

Command Options:

DIRECT

Specifies that an actual punch is to be assigned to punch the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file. Then, when a punch becomes available, the file will be punched. If DIRECT is specified and no punch is available, you will receive an error message at the workstation. The default value is to spool the file.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PUNCH command has completed execution. Normally, when you issue a PUNCH command, a background task is created, and the PUNCH command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PUNCH operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the PUNCH command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 3, To Punch a Spool File:

\rightarrow `PUNCH` Δ [`JOB`=jobname] [`HOLD`= $\left\{ \begin{array}{l} L \\ N \\ Y \end{array} \right\}$] [`FILENAME`= $\left\{ \begin{array}{l} \text{filename} \\ 'filename' \\ "filename" \end{array} \right\}$] [`ACCT`=acct]
 $\left\{ \begin{array}{l} LOG \\ PRINT \\ PUNCH \\ RDR \end{array} \right\}$ [`ALL`= $\left\{ \begin{array}{l} YES \\ NO \end{array} \right\}$] [`COPIES`= $\left\{ \begin{array}{l} n \\ 1 \end{array} \right\}$] [`SKIP`= $\left\{ \begin{array}{l} n \\ 0 \end{array} \right\}$] Δ [`DIRECT`][`WAIT`]

Command Options:

DIRECT

Specifies that an actual punch is to be assigned to punch the file requested in the command. When **DIRECT** is not specified, the file requested will be copied to a spool file. Then, when a punch becomes available, the file will be punched. If **DIRECT** is specified and no punch is available, you will receive an error message at the workstation. The default value is to spool the file.

WAIT

Specifies that control is not returned to **WORKSTATION** mode until the **PUNCH** command has completed execution. Normally, when you issue a **PUNCH** command, a background task is created, and the **PUNCH** command executes as a background job, allowing you to continue to use **WORKSTATION** mode for user or system programs. However, there may be times when you want to be sure the **PUNCH** operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the **WAIT** keyword with the **PUNCH** command causes **WORKSTATION** mode to be stopped until the command has completed execution.

Format 4, Punching a Tape File:

`PUNCH` Δ [`FILENAME`= $\left\{ \begin{array}{l} \text{filename} \\ 'filename' \\ "filename" \end{array} \right\}$] [`RDPASS`=password],`VSN`=volume,`DEVICE`= $\left\{ \begin{array}{l} did \\ TAPE \end{array} \right\}$
 $\left\{ \begin{array}{l} YES \\ NO \end{array} \right\}$ Δ [`DIRECT`][`WAIT`]

Command Options:

DIRECT

Specifies that an actual punch is to be assigned to punch the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file. Then, when a punch becomes available, the file will be punched. If DIRECT is specified and no punch is available, you will receive an error message at the workstation. The default value is to spool the file.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PUNCH command has completed execution. Normally, when you issue a PUNCH command, a background task is created, and the PUNCH command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PUNCH operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the PUNCH command causes WORKSTATION mode to be stopped until the command has completed execution.

Format 5, Punching a Unit Record File:

$$\text{PUNCH}\Delta\text{DEVICE}=\left\{\begin{array}{l} \text{did} \\ \text{DISKETTE} \\ \text{RDR} \end{array}\right\}, \text{FILENAME}=\left\{\begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array}\right\}, \text{VSN}=\text{volume}$$

Δ [DIRECT][,WAIT]

NOTE:

The FILENAME= and VSN= parameters are required only when using a diskette.

Command Options:

DIRECT

Specifies that an actual punch is to be assigned to punch the file requested in the command. When DIRECT is not specified, the file requested will be copied to a spool file. Then, when a punch becomes available, the file will be punched. If DIRECT is specified and no punch is available, you will receive an error message at the workstation. The default value is to spool the file.

WAIT

Specifies that control is not returned to WORKSTATION mode until the PUNCH command has completed execution. Normally, when you issue a PUNCH command, a background task is created, and the PUNCH command executes as a background job, allowing you to continue to use WORKSTATION mode for user or system programs. However, there may be times when you want to be sure the PUNCH operation has been completed successfully before going on with your program, such as when you are making a backup copy of a file. In this case, entering the WAIT keyword with the PUNCH command causes WORKSTATION mode to be stopped until the command has completed execution.

Examples:

```
PUN FILENAME=PAYFILE,VSN=D00063
```

In this example, a punched card copy is made of a file name PAYFILE, located on volume D00063.

```
PUN FILENAME=PUNCHIT,QUEUE=PRINT,COPIES=5
```

In this example, a punched card copy is made of the spooled file PUNCHIT, located in the PRINT spool file directory. Five copies will be made of the file.

The following conditions could cause this command to be rejected:

- **Incorrect Syntax**

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- **File or Volume Not Available**

The file or volume you requested is not available for access.

- **Spooling Not Configured**

The spooling process was not configured at the time your system was generated. Therefore, no spool files are available to access.

- **Punch Not Available**

There is no punch available to punch the requested file or module. The file or module is copied to a spool file and punched when a punch becomes available. Spool files remain in queue and are punched when a punch becomes available.

- **Invalid Password**

The password or passwords you entered are invalid.

RECALL

4.4.16. Displaying All or Part of the Workstation Log File (RECALL Command)

The RECALL command lets you display all or part of your workstation log file at your workstation screen. You can view selected portions of the log file by specifying a particular time span (for instance, from 9:00 to 10:00). Or, you can indicate the number of messages, prior to the most current one, that you'd like to see.

If your workstation is free (that is, not allocated to a job), the RECALL command uses the entire screen to display the log file. If the screen fills up before the display is finished, the following system message appears:

```
0n LR05 CONTINUE? (Y OR N)
```

To continue the display, key in the message prefix (0n) and Y. To end it, key in the message prefix (0n) and N.

If your workstation is allocated to another job when you issue the RECALL command, the log file appears one record at a time on the second line of the system messages area of the screen. You must hit the XMIT key after each record is displayed in order to see the next one.

If you are specifying a time period and you accidentally enter an invalid starting or ending time, the RECALL command displays the message:

```
0n LR02 LIMIT INVALID. CONTINUE? (Y OR N)
```

If you key in Y, the RECALL command substitutes a default time in place of the invalid time you entered. If you key in N, you cancel the command and you're free to reenter it with the correct times. If you accidentally enter an invalid number of messages, the RECALL command ignores your entry and starts at the first message in the log file.

Format:

```
RECALL△ { LAST△nn  
          { hh:mm:ss-hh:mm:ss } }
```

Command Parameters:

LAST△nn

Indicates the number of messages in the log file that you wish to see. The keyword LAST must be separated from the number of messages (nn) by a space.

↓

hh:mm:ss-hh:mm:ss

Indicates that you wish to see the contents of the log file for a particular time period. The minutes (mm) and seconds (ss) are optional. You must, however, separate the beginning time from the ending time with a dash (-). All time must be specified in military time.

You can specify a beginning time without specifying an ending time. In this case, the log display starts at the time you specify and ends at the end of the log file. Similarly, you can specify an ending time without specifying a beginning time. In this case, the log display starts at the beginning of the file and ends at the time you specified. When keying in an ending time by itself, you must prefix it with a dash (-).

Examples:

RECALL LAST 24

In this example, only the last 24 log file entries are displayed.

RECALL 15:30-16

In this example, all the log file entries from 15:30:00 (3:30 p.m.) to 16:00:00 (4:00 p.m.) are displayed.

RECALL 8:30

In this example, the log file display starts at 8:30:00 (8:30 a.m.) and continues to the end of the file.

RECALL -17

In this example, the log file display starts at the beginning of the file and continues till the ending time is reached (17:00:00 or 5:00 p.m.).

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- Logging Not Active

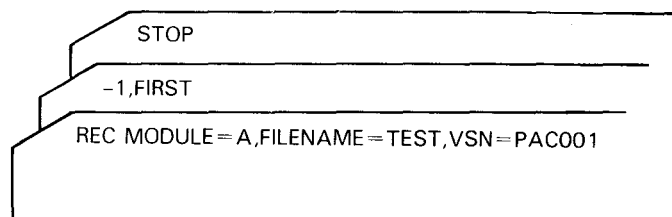
Workstation logging is not active; therefore, no log file is created. Workstation logging is not active if you did not specify it at SYSGEN time with the CONSOLOG parameter or later with the operator's spooling command, SET SPL, CNSLG.

↑

REC**4.4.17. Recovering Deleted Modules (RECOVER Command)**

The RECOVER command enables you to recover deleted source, procedure, or macro modules in the SAT library. The command functions by listing, for the file specified, all deleted modules of the module type specified in the parameter you enter with the RECOVER command. Each module displayed is numbered, as well as being listed with its header comment, date, and time it was created. Modules that are not deleted but have the same module name as that specified in the command parameters are also displayed. These modules are displayed with an indicator to show that they are not deleted. When you have reviewed the complete display, the system allows you to choose one of the modules displayed and give it a new name. Renaming the module "undeletes" it. *This command may also be used to rename modules not deleted.* The new name you give the module must be unique; duplicates are not permitted. The RECOVER command can recover only the modules since the last LIBS PAC operation was performed. The LIBS PAC routine that reorganizes file space destroys all deleted modules. You must choose the module to be recovered carefully because the RECOVER display could show you several very similar modules.

You may use the RECOVER command in an ENTER stream by observing certain special procedures. The RECOVER command requires three cards to execute in an ENTER stream. The first card contains the RECOVER command and its associated file parameters. The second card indicates which copy of the module specified by the file parameters you want recovered and the new name you want to give the recovered module. Entering a zero indicates that you want the most recent copy of the module recovered; a minus 1 (-1) indicates you want the next most recent copy; minus 2 (-2), the next most recent; etc. The third card simply says STOP to end the RECOVER command. The following is an example of the use of the RECOVER command in an ENTER stream:

**NOTE:**

Only one RECOVER command may be processed at a time. You may not enter another RECOVER until the command currently being processed is completed.

The RECOVER command issues a message to you, informing you it has completed execution.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:

```
RECOVER△MODULE=modulename [ ,TYPE={module-type} ] ,FILENAME={filename }
                                     { 'filename' }
                                     { "filename" }
[ ,RDPASS=password ][ ,WRPASS=password ],VSN=volume
```

Example:

```
REC MODULE=A,FILENAME=TEST,VSN=PAC001
```

In this example, the user is seeking to recover source modules named A. The deleted modules reside in a file named TEST, on the PAC001 volume. Entering this produces the following displays:

```
1.  01 A          COPY ONE          80/02/15  16:49
2.  02 A          COPY TWO          80/02/15  16:49
3.  03*A         COPY THREE         80/02/15  16:49
4.  IS56 ENTER  ''ELEMENT-NUMBER, NEW-NAME'' OR ''STOP''
5.  1,FIRST
6.  01 A          COPY TWO          80/02/15  16:49
7.  02*A         COPY THREE         80/02/15  16:49
8.  IS56 ENTER  ''ELEMENT-NUMBER, NEW-NAME'' OR ''STOP''
9.  STOP
10. IS90 RECOVER  COMMAND TERMINATED NORMALLY
```

The above display is explained as follows:

Lines 1, 2, 3, and 4 are displayed after you enter the RECOVER command. They show all the modules that match the specifications you entered with the command. Line 4 is message IS56, asking you to enter either the number of, and a new name for, the module you wish to recover, or STOP, to terminate the RECOVER command. Line 5 shows that the user has entered 1,FIRST. Therefore, the first module shown in the display will be recovered, and renamed FIRST instead of A. In lines 6, 7, and 8, the RECOVER command continues, displaying the remaining modules matching the specifications entered with the command. On line 9, the user has entered STOP, terminating the RECOVER command. Line 10 shows the message indicating that the RECOVER command has terminated normally.

If the user enters an FSTATUS command to display the modules in the file named TEST, he will see a display showing the recovered module now named FIRST:

```
FSTATUS  FILENAME=TEST,VSN=RES LONG
S-FIRST   COPY ONE           80/02/15 16:49
IS83 FSTATUS FINISHED, 00001 ELEMENTS WERE DISPLAYED
```

The following conditions could cause this command to be rejected:

■ Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

■ No Deleted Modules

There are no deleted modules in the file specified.

■ Attempt to Duplicate Existing Name

In renaming a module in order to undelete it, you specified a name already in use by a module. The name you use must not be in use by a module that has not been deleted.

■ File or Volume Not Available

The file or volume you requested is not available for access.

■ Invalid Password

The password or passwords you entered are not valid.

REM

4.4.18. Inserting a Comment in a Command Stream (REMARK Command)

The REMARK command allows you to enter a comment in a stream of commands. This command is different from the COMMENT command, which allows you to add a comment to a library module header. The principal use of the REMARK command is in card decks for batch processing, but it may be used in any situation to include a command in a comment stream. The command and text generate no output to the workstation.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:

REMARK△text

Command Option:

text

Is the text of the comment. The total length of the command and its associated text must be a maximum of 60 characters.

Example:

```
REM THIS DECK EXECUTES PAYJOB
```

In this example, the user is inserting a comment to indicate that the card deck referred to executes the program PAYJOB.

The following conditions could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous.

STA**4.4.19. Obtaining System Status Information (STATUS Command)**

The STATUS command enables you to obtain information about various aspects of your system, including usage of terminals and workstations, number of jobs running on the system, system storage resources in use and available, information on jobs running under your user-id, and interactive commands and facilities being performed under your user-id. STATUS can also give you a listing of the disk, tape, and diskette volumes currently mounted on your system.

Format:

```
STATUS△ { TERMINALS
          { RESOURCES
          { JOBS
          { FUNCTIONS
          { VOLUMES }
```

The output of the STATUS command depends on which, if any, of the command options you enter.

Example:

STA

Entering the STATUS command with no command option produces a display of your user-id, the current date and time, and the job name of the job, if any, to which your workstation is connected.

```
jobname    user-id    mm/dd/yy    hh:mm
```

Example:

STA T

Entering the STATUS command with the command option TERMINALS produces a display similar to the following:

```

TRM          USERID
014          USERA
016          BILLPR
IS90 STATUS  COMMAND TERMINATED NORMALLY

```

where:

TRM

Is the physical device address of local workstations. It is the logical name of remote workstations and terminals assigned during ICAM generation.

USERID

Is the user-id of the person currently logged on at the workstation or terminal.

Example:

STA R

Entering the STATUS command with the command option RESOURCES produces a display similar to the following:

```

BUFFERS      INT  ENT  BCK  JOBS      SYS-SIZE
014/003640   001  001  000  00/07     0491520

```

where:

BUFFERS

Shows the number of dynamic storage buffers allocated to interactive users, together with the total size (in bytes) of those buffers. The first number is the number of buffers; the second, the total size.

INT

Shows the number of interactive tasks running on the system.

ENT

Shows the number of ENTER command streams being run on the system.

BCK

Shows the number of background (batch) tasks running on the system.

JOBS

Shows, first, the number of system job slots in use; second, the number of job slots configured on the system.

SYS-SIZE

Shows the amount of main storage present in the system. This figure does not change unless more main storage is physically added to your system.

Example:

STA J

Entering the STATUS command with the command option JOBS produces a display similar to the following:

| JOBNAME | SIZE | CPU TIME | STEP | EXEC | JOB NO | MASTER |
|---------------|---------|----------|------|----------------|---------|--------|
| ASMBLY | 040960 | 12.1 | 01 | ASM000 | 12 | USERA |
| COBOLJOB | 073862 | 125.6 | 02 | COBOL0 | 15 | USERA |
| RPGLINK | 016384 | 5.4 | 01 | LNKEDT | 16 | USERA |
| UNUSED MEMORY | 0388352 | | | LARGEST REGION | 148,624 | |

where:

JOBNAME

Is the name of the job running on the system.

SIZE

Shows the amount of main storage, in bytes, being used by the job.

CPU TIME

Shows the amount of time the job has used the central processing unit.

STEP

Shows the number of the job step currently in execution.

EXEC

Is the name of the program (within the job) currently being executed.

JOB NO

Gives the job accounting number assigned to the job by the system.

MASTER

Shows the user-id of the master workstation controlling the job. Note that all the jobs have the same user-id. *You are permitted to obtain only the status information about jobs running under your user-id.*



UNUSED MEMORY

Shows the total free main storage available.

LARGEST REGION

Shows the size of the largest free region of main storage.



Example:

STA F

Entering the STATUS command with the command option FUNCTIONS produces a display similar to the following:

| USERID | COMMAND | MODE | ID |
|--------|---------|------|-----|
| USERA | EDT | I | 001 |
| USERA | PRINT | B | 002 |

where:

USERID

Is your user-id. *Only commands running under your user-id are displayed.*

COMMAND

Lists by name the commands being executed.

MODE

Shows in which mode (interactive or background) the command is being executed. Some commands, such as PRINT, can be executed as batch jobs to avoid tying up WORKSTATION mode for long periods of time while the command is executing.

ID

Is the internal task number assigned to the command by the system.

NOTE:

Only commands executing under your user-id are displayed in the STATUS FUNCTIONS display.

Example:

STA V

Entering the STATUS command with the command option VOLUMES produces a display similar to the following:

```
D-REL070      D-REL052      T-BACKUP      F-WORK20
```

where:

- D Prefacing the volume serial number with a D (as in D-REL070 in the sample screen) indicates that the volume is a disk volume.
- T Prefacing the volume serial number indicates that the volume is a tape volume.
- F Prefacing the volume serial number indicates that the volume is a diskette volume.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

VT

4.4.20. Listing the Contents of a VTOC (VTOC Command)

↓
The VTOC command enables you to produce a listing of the files present on a disk or diskette volume. The listing may contain all the files on the volume, or just those whose names match a prefix you specify with the command. For disk and format-label diskettes, the listing for each file gives the file name, file size and type, and extent information. After the last file has been listed, a count of free cylinders left on the disk or diskette is given. For data-set-label diskettes, the listing for each file gives the file name, the starting and ending addresses of each file, and the file's block size and record size.

↑
NOTE:

Only one VTOC command may be executed at a time. You must wait until the currently executing VTOC command finishes before entering another.

You may use more than one line of the workstation screen to enter this command and its associated parameters. To do this, place a dash (-) as the last character in the first line of the command. The system recognizes the dash as a continuation character and asks you (in a message) for further input. You may use as many lines of the screen as you wish; just place a dash at the end of every line except the line on which the parameter string ends.

Format:

```
VTOCΔ['file-prefix',]VSN=volumeΔ[FREE]
```

Command Options:

file-prefix

Specifies a full (such as JOBDUMP) or a partial (such as JOB) file name. If you enter a file prefix, the listing produced will show only those files whose file names match the prefix specified. *This parameter must be enclosed in apostrophes.*

↓
FREE

Specifies that you also want a listing of each free extent available on the volume you are referencing. For disk and format-label diskette volumes, this parameter gives the size of each free extent in cylinders and tracks. For data-set-label diskettes, this parameter gives you the beginning and ending addresses, and size (in sectors) of each free extent.

↑
Examples:

1. For a disk volume:

```
VT RES FREE
```


In this example, the VTOC command is used to produce a listing of the files present on the disk volume named RES (the system RESident disk pack). Since we specified the FREE option, we will receive a listing of the size of each free extent. The following display is produced:

| FILENAME | TYPE | EXT | CYL | %DIR | %DATA | %THIRD | AVAIL. |
|----------|-------|-----|-----|------|-------|--------|--------|
| \$VTOC | MIRAM | 001 | 001 | | | | |
| SCSCOR | SAT | 001 | 001 | 0 | 0 | 0 | 80 |
| SYSACAT | SAT | 001 | 001 | 4 | 0 | 0 | 0 |
| SYSDUMP | SAT | 002 | 015 | 100 | 0 | 0 | 0 |
| SYSJCS | SAT | 004 | 021 | 19 | 81 | 0 | 2920 |
| SYSOBJ | SAT | 004 | 009 | 52 | 35 | 0 | 1200 |
| SYSMAC | SAT | 007 | 046 | 22 | 97 | 0 | 0 |
| SYSSRC | SAT | 001 | 009 | 5 | 91 | 0 | 920 |
| SYSTRAN | SAT | 001 | 012 | 89 | 97 | 0 | 0 |
| SYSTRANA | SAT | 001 | 012 | 89 | 97 | 0 | 0 |

On the display:

- FILENAME gives the name of the file.
- TYPE gives the type of file, that is, the access technique used to create the file.
- EXT gives the number of extents that, together, make up the file.
- CYL gives the size of the file in total number of cylinders composing the file.
- %DIR expresses in percentage how much of the directory portion of the file has been used up.
- %DATA expresses in percentage how much of the data portion of the file has been used up.
- %THIRD expresses in percentage how much of the third portion of the file (the portion containing block load modules) is used up.
- AVAIL gives the amount of space (expressed in number of blocks) and how much space of the file is unused *and* unassigned to the directory, data, or block load module portions of the file.

If the VTOC listing requires more than 24 lines to display, the listing will continue to roll up from the bottom of the screen, and some of the initial listings of the VTOC will be rolled off the screen.

| FILENAME | TYPE | EXT | CYL | %DIR | %DATA | %THIRD | AVAIL. |
|---------------------------------------|-------|-----|-----|------|-------|--------|--------|
| \$Y\$ELOG | SAT | 001 | 005 | 0 | 0 | 0 | 0 |
| \$Y\$ESUM | SAT | 001 | 001 | 0 | 0 | 0 | 0 |
| \$Y\$\$SHR | SAT | 001 | 001 | 0 | 0 | 0 | 0 |
| \$Y\$\$SYSTEMTABLES | SAT | 001 | 001 | 0 | 0 | 0 | 0 |
| \$Y\$MIC | SAT | 001 | 002 | 0 | 0 | 0 | 0 |
| \$Y\$\$CLOUD | SAT | 002 | 003 | 30 | 96 | 0 | 0 |
| \$Y\$FMT | MIRAM | 001 | 002 | | | | |
| SG\$OBJ | SAT | 001 | 016 | 30 | 86 | 0 | 1920 |
| SG\$MAC | SAT | 002 | 021 | 6 | 98 | 0 | 2080 |
| FREE SPACE: CYLINDERS 001, TRACKS 000 | | | | | | | |
| FREE SPACE: CYLINDERS 415, TRACKS 000 | | | | | | | |
| TOTAL FREE: CYLINDERS 416, TRACKS 000 | | | | | | | |

This display concludes the disk volume RES VTOC listing. The count of free cylinders is visible at the bottom of the display.

2. For data-set-label diskettes:

VT MYVOL1 FREE

In this example, the VTOC command is used to produce a listing of the files present on the data-set-label diskette volume named MYVOL1. Since the command includes the FREE option, the listing shows the size of each free extent, along with the starting and ending addresses of each extent:

| FILENAME | START OF FILE | END OF FILE | END OF DATA | BLOCK SIZE | RECORD SIZE |
|--------------|-----------------------------|----------------|-------------------|---------------|----------------|
| FIL1 | 01 0 01 | 02 1 22 | 01 0 01 | 128 | 128 |
| FIL2 | 02 1 23 | 07 1 12 | 02 1 23 | 256 | 256 |
| FIL3 | 07 1 13 | 13 0 26 | 07 1 13 | 256 | 256 |
| FIL4 | 13 1 01 | 14 0 24 | 13 1 01 | 256 | 256 |
| AAA | 44 0 08 | 63 0 19 | 44 0 08 | 128 | 128 |
| FIL6 | 15 0 16 | 24 1 21 | 15 0 16 | 256 | 256 |
| TST | 63 0 20 | 65 0 15 | 63 0 20 | 128 | 128 |
| FREE EXTENTS | START | END | SIZE (IN SECTORS) | | |
| | 14 0 25 | 15 0 15 | 43 | | |
| | 24 1 22 | 44 0 07 | 1000 | | |
| | 65 0 16 | 74 1 26 | 505 | | |
| IS90 VTOC | COMMAND TERMINATED NORMALLY | | | | |



On the display:

- FILENAME gives the name of the file.
- START OF FILE gives the diskette address where the file begins. The format is track-side-sector.
- END OF FILE gives the ending diskette address for the file.
- END OF DATA gives the diskette address of the last sector to be used for data within the file.
- BLOCK SIZE gives the size of the block written to the file.
- RECORD SIZE gives the size of the records which make up the file.

The FREE EXTENTS information at the bottom of the display is listed because the FREE option was specified. This display shows the following:

- START gives the diskette address where the free extent begins. The format is track-side-sector.
- END gives the ending diskette address for the extent.
- SIZE gives the number of sectors available in this extent.

The following conditions could cause this command to be rejected:

- Incorrect Syntax

The command was misspelled or ambiguous, or the command or parameters were improperly entered.

- Volume Not Available

The volume you requested is not available.



BR LO

4.5. COMMANDS TO CONTROL WORKSTATION LOGGING

→
↓
↑
Use the following commands to control the process of *workstation logging*. Workstation logging automatically creates a record of all the system commands issued from your workstation, system messages sent to your workstation, and the responses you make to those messages. If you want it to include commands entered in workstation mode also, you can specify that with the LOG parameter of the SCREEN command (4.2.8). The record created is a *workstation log file*; it begins when you log on the workstation and ends when you either log off the workstation or *breakpoint* the log file for printing. Workstation logging is available if *console logging* (CONSOLOG parameter) is configured when your system is generated (SYSGEN). Workstation logging is available to both locally connected workstations and remote terminals acting as workstations. When you log on, use the LOG parameter of the LOGON command to specify whether the workstation log file is to be printed. See 2.3.2 for complete information on the LOGON command.

4.5.1. Breakpointing Workstation Log Files (BRKPT LOG Command)

The BRKPT LOG command enables you to close the workstation log file and make it available to the output writer for printing *before* you log off the workstation. A new workstation log file is started with the first message or command after the BRKPT LOG command is entered. The new workstation log file continues until it too is breakpointed, or you log off the workstation.

Format:

BRKPT△LOG

There are no parameters associated with this command. If your BRKPT LOG command is accepted, you receive the following message:

BREAKPOINT TAKEN FOR WORKSTATION LOG yy/mm/dd USER=xxxxxx

where:

xxxxxx

Is your user-id.

If the BRKPT LOG command is *not* accepted, you receive the following message:

WORKSTATION LOGGING NOT ACTIVE

This indicates that you did not specify that the workstation log was to be printed when you logged on, or that console logging was not specified at SYSGEN.

If an error occurs while the breakpoint is being taken, you receive the following message:

BREAKPOINT ERR WORKSTATION LOGGING SUSPENDED

Contact your system administrator if you receive this message.

The following is a sample printout of a breakpointed workstation log:

```

LOGON                                     L 14:05:44 311
0A IS19 LOGON ACCEPTED AT 14:05:44 ON 81/02/17, REV 07. 1.0 W 14:05:45 311
0B IS27 TODAYS BULLETIN IS:              W 14:05:49 311
0C      -- TO TYPE IN COMMANDS,  DEPRESS 'FUNCTION' AND -- W 14:05:49 311
0D      -- 'SYSTEM-MODE' KEYS SIMULTANEOUSLY, THEN TYPE -- W 14:05:50 311
0E      -- THE COMMAND AND DEPRESS TRANSMIT.           -- W 14:05:51 311
0F      -- ON UNISCOPES DEPRESS 'MESSAGE WAITING' KEY. -- W 14:05:51 311
VT RES                                    W 14:06:43 311
0G  FILENAME          TYPE  EXT CYL %DIR %DATA %THIRD AVAIL. W  4:06:44 311
0H  $VTOC              SEQ.  001 001                                W 14:06:45 311
0J  $IMPL              SEQ.  001 000                                W 14:06:45 311
0K  $IPL               SEQ.  001 001                                W 14:06:46 311
0L  $Y$STRANA          SAT   001 005   99   99     0   360 W 14:06:46 311
0M  $Y$MAC             SAT   001 010   51   99     0   360 W 14:06:46 311
0N  $Y$OBJ            SAT   001 003   80   97     0   840 W 14:06:47 311
0P  $Y$JCS            SAT   001 003   50   70     0     0 W 14:06:47 311
0Q  $Y$CAT            SAT   001 001     1     0     0     0 W 14:06:48 311
0A  $Y$STRAN          SAT   001 005   99   99     0   360 W 14:06:48 311
0B  $Y$L0D            SAT   005 071     2   86    99     0 W 14:06:48 311
0C  $Y$SHR            SAT   001 001     0     0     0     0 W 14:06:49 311
0D  $Y$SYSTEMTABLES   SAT   001 001     0     0     0     0 W 14:06:49 311
0E  $Y$MIC            SAT   001 002     5   93     0   780 W 14:06:50 311
0F  $Y$SCLOD          SAT   001 005   51   77     0   540 W 14:06:50 311
0G  $Y$FMT            MIRAM 004 007                                W 14:06:51 311
0H  $Y$SAVE           MIRAM 001 002                                W 14:06:51 311
0J  $Y$DIALOG         MIRAM 001 005                                W 14:06:51 311
0K  $Y$SDF            MIRAM 001 002                                W 14:06:52 311
0L  $Y$HELP           MIRAM 001 003                                W 14:06:52 311
0M  SG$XXX            SAT   001 001     0     0     0     0 W 14:06:53 311
0N  $Y$ELOG           MIRAM 001 001                                W 14:06:53 311
0P  $Y$ESUM           MIRAM 002 002                                W 14:06:53 311
0Q  $C$COR            SAT   001 001     0     0     0     0 W 14:06:54 311
0A  SG$OBJ            SAT   001 005   80   99     0   780 W 14:06:54 311
0B  SG$L0D            SAT   001 002     6   95    95   540 W 14:06:55 311
0C  SG$JCS            SAT   001 002   18   99     0   600 W 14:06:55 311
0D  SG$MAC            SAT   001 005   48   99     0   600 W 14:06:56 311
0E  IVPLIB            SAT   001 005   61   98     0  1140 W 14:06:56 311
0F  $Y$DUMP           MIRAM 001 005                                W 14:06:56 311
0G  $Y$SMCLOG         MIRAM 001 003                                W 14:06:57 311
0H  $Y$SRC            SAT   001 003   13   74     0     0 W 14:06:57 311
0J  SY$POOL           SAT   001 050  100     0     0     0 W 14:06:58 311
0K  ALTFILE           SAT   001 010     1     6     0  4140 W 14:06:58 311
0L  TOTAL FREE: CYLINDERS 322, TRACKS;001 W 14:06:58 311
FS  SG$L0D,RES       W 14:09:53 311
0M  IS09 DOPEN: VOLUME NAME NOT SPECIFIED, FILE NOT IN CATALOG W 14:09:53 311
FS  ,SG$L0D,RES     W 14:11:30 311
0N      L-SG$SZE00      L-SG$DSL00      L-SY$CDI00      L-SGCNFG00 W 14:11:34 311
0P  IS83 FSTATUS FINISHED, 00004 ELEMENTS WERE DISPLAYED W 14:11:36 311
BR  LO

```

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled, not entered correctly, or entered with parameters attached.

DI LO**4.5.2. Obtaining Workstation Log Information (DISPLAY LOG Command)**

The DISPLAY LOG command gives you a 1-line report on the status of your workstation log file. The display shows the number of workstation lines used since you logged on.

Format:

DISPLAYΔLOG

There are no parameters associated with this command. The following is an example of the display produced by the DISPLAY LOG command:

TOTAL WORKSTATION LINES 0000063

If the command is not accepted, you receive the same message, WORKSTATION LOGGING NOT ACTIVE, as when the BRKPT LOG command is not accepted.

The following condition could cause this command to be rejected:

Incorrect Syntax

The command was misspelled, not entered correctly, or entered with parameters.



**PART 3. A GUIDE TO OTHER INTERACTIVE
PRODUCTS**



5. Quick-Reference Guide to the Interactive Facilities, Distributed Data Processing, BASIC, and ESCORT

In this section, you will find information on the various interactive facilities, the distributed data processing (DDP) system, and the interactive programming languages BASIC and ESCORT, available with OS/3. The interactive facilities discussed here are:

- Screen format services
- Menu services
- OS/3 interactive data utilities
- Interactive dump/restore hardware utility
- General editor (EDT)

Included with the discussion of the general editor, are the three subeditors available:

- RPG II editor
- COBOL editor
- Error file processor

Another interactive facility, interactive job control stream preparation, has already been introduced to you in 3.2.

The information given herein is meant only for quick reference: to initialize the interactive facilities and DDP, or to refresh your memory concerning the commands and functions available through these facilities. For complete detailed information on the use of DDP, BASIC, ESCORT, and the interactive facilities, refer to the appropriate user guide. You will find a list of these in the preface.

5.1. SCREEN FORMAT SERVICES

The screen format services allow you to create, modify, and use formatted screen displays in your programs. Screen format services consist of two software products: the screen format generator and the screen format coordinator.

5.1.1. The Screen Format Generator

The screen format generator allows you to interactively create and modify formatted screen displays. You lay out the format at the workstation and specify information concerning each field of the format. To use the screen format generator, you first enter the RV command with the job name SFGEN:

→ RV△SFGEN

When the system has processed your command, the screen clears and the following display appears on the screen:

↓

```

1.  FUNCTION (1):    1 CREATE  2 CREATE-FROM  3 MODIFY  4 DELETE
2.                    5 SHOW   6 LIST      7 SPOOL   8 TERMINATE
3.  OLD FORMAT NAME (_____) IS ON THE FOLLOWING LIBRARY:
4.  FILE NAME: ($Y$FMT                ) VOLUME: (RES      )
5.
6.  NEW FORMAT NAME (_____) IS TO BE STORED ON THE FOLLOWING LIBRARY:
7.  FILE NAME: ($Y$FMT                ) VOLUME: (RES      )
8.  IF THIS FILE DOES NOT EXIST, ALLOCATE (002) CYLINDERS. INCREMENT IS (01) CYL.
9.
10. ** FUNCTION KEYS ARE:F1-GO TO HOME SCREEN, F5-BREAKPOINT SPOOL FILE,
11.    F13-HELP,F14-EXIT HELP, F20-RESTORE SCREEN
  
```

This screen is called the home screen. It's always the first display presented when you activate the screen format generator. On it, you choose a function and indicate either where an old format resides or where a format you're creating will ultimately reside. If necessary, you can allocate file space for newly created formats. Once you complete the home screen, the next screen presented is the characteristics screen:

↑

- ↓
1. GLOBAL CHARACTERISTICS FOR FORMAT *format-name*: *
 2. LOWER CASE TRANSLATION? (1) 1 YES 2 NO
 3. ALPHABET: (ENGLISH) SCREEN FORMAT IS (1): 1 ORIGINAL 2 OVERLAY
 4. ERASE/UNLOCK OPTION (1): 1 NONE 2 REPLENISH SCREEN 3 ERASE SCREEN
 5. 4 UNLOCK KEYBOARD 5 CONDITIONAL INDICATOR IN USER PROGRAM
 6. ERROR RETRY COUNT: (2) SPECIAL EDITING CHARACTERS? (1): 1 NO 2 YES
 7. SPECIAL DISPLAY CONTROL? (1): 1 NO 2 YES
 8. DO YOU WISH TO SPECIFY AN ERROR MESSAGE FIELD? (1): 1 NO 2 YES
 9. DISPLAY RETENTION FOR ALL FIELDS? (1) 1 NO 2 YES
 10. FUNCTION OR COMMAND KEYS TO BE DEFINED? (1): 1 NO 2 YES
 11. DOES THIS FORMAT HAVE A NON-DISPLAYED CONSTANT? (1): 1 NO 2 YES

On this screen, you provide general information about how your format will look and act. Certain selections, in turn, cause other screens, called optional screens, to appear. For example, if you entered (2) for SPECIAL EDITING CHARACTERS, the following display will result after you transmit the characteristics screen:

ENTER EDIT CHARACTERS IN PARENTHESES FOR THE FOLLOWING SYMBOLS

| | | |
|----|--------------------|--------|
| \$ | : CURRENCY SIGN | = (\$) |
| . | : DECIMAL POINT | = (.) |
| , | : COMMA | = (,) |
| * | : REPLACEMENT CHAR | = (*) |
| CR | : CREDIT | = (CR) |
| DB | : DEBIT | = (DB) |
| - | : REPLENISH CHAR | = (-) |

NOTE: MAKE SURE NO SYMBOL IS AMBIGUOUS WITH ANY VALID PICTURE STRING CHARACTER.

After you have supplied all the information requested by the characteristics screen and by any optional screens that were displayed, you are presented with a blank screen. This is the template screen, on which you actually lay out the format you want to create. The following is an example of a screen format after the first template screen:

↑

```

                                PERSONAL CREDIT REPORT

NAME: _____ /___/79

ADDR: _____

SOCIAL SECURITY: _____ STATE: ___ ZIP: _____

ACCOUNT NUMBER: _____

PAST DUE AMOUNT: _____

NEW BALANCE: _____ PAYMENT DUE DATA: ___/___/___

```

The screen format you have laid out is presented to you two more times to allow you to specify further information about each field within the format. If the screen format generator requires more information about a particular field one or more dialog screens are presented, requesting further information. The following is an example of a dialog screen for the "social security" field of this screen format:

```

SOCIAL SECURITY: 999!99!9999
(FLD00007) IS FOR FIELD USE OF (1):  1 OUTPUT  2 INPUT  3 BOTH
INTERNAL USAGE IS (1):  1 DISPLAY  2 PACKED  3 BINARY  6 ZONED
INTERNAL LENGTH IS (08)
PLEASE INDICATE WHETHER THE FOLLOWING ARE REQUIRED:  1 NO  2 YES
    CONDITIONAL DISPLAY (1)          SPECIAL DISPLAY PROPERTIES (1)
    CONDITIONAL RETENTION (1)       CONDITIONAL PROTECTION (1)
                                     RANGE CHECKING (1)

REPLACE !'s IN THE FOLLOWING LINE BY INSERTION CHARACTERS
999!99!9999

```

When you have finished creating the screen format, it is automatically stored in either a file maintained by the screen format generator (\$Y\$FMT) or an alternate file that you specified on the home screen.

If, while you are creating a screen format, you come to a step you do not understand, you can receive help in the form of screen displays describing each step of the format generation process.

To receive the appropriate HELP screen for the step in which you are working, press the FUNCTION key and, while holding it down, press the key marked F13 or its functional equivalent. When you have finished studying the HELP screen and you want to go back to the step on which you were working, press the FUNCTION key and F14 or its functional equivalent.

5.1.2. The Screen Format Coordinator

The screen format coordinator retrieves screens you want to use in a program from the \$Y\$FMT file or alternate file and manages their use in the program. To use a screen format in one of your programs, you must indicate in the program at what point the format is to be used and include a special statement in the job control stream of your program, alerting the system that you require screen format services in the program. Following is the job control statement to use screen formats in your program:

```
//[symbol] USE SFS [ , { [format-file-lfd-1]/[format-file-lfd-2] } ] [ , initial-screen ]
                    [ { $Y$FMT } ]
                    [ format-file-lfd ]
                    [ , { nnn } ] [ , screen-format-1=alias-1, ..., screen-format-12=alias-12 ]
```

Parameters:

```
[ , { [format-file-lfd-1]/[format-file-lfd-2] } ]
  [ { $Y$FMT } ]
  [ format-file-lfd ]
```

Specifies the names for up to two screen format files. If you omit a format-file-lfd name, it is assumed that all screen formats used reside in \$Y\$FMT.

initial-screen

Specifies the first format name to be used in behalf of the user program. Use of this parameter depends on the program's language.

```
{ nnn }
```

Specifies the number of formats that are to reside in main storage for use with a given file. You use this parameter if a job alternates between two or more formats. Including this parameter reduces I/O activity needed to retrieve the format. The default value is 1.

```
[ , screen-format-1=alias-1... [ , screen-format-12=alias-12 ] ]
```

Specifies that a name other than the real format name is to be used to identify a format. You may include a maximum of 12 aliases. If you supply more than 12 aliases, your job control stream will be rejected. Write each alias into the statement as follows:

Real Format Name=Program Format Name

NOTE:

For complete information on the use of the screen format generator and screen format coordinator, refer to the screen format services concepts and facilities manual.

5.2. MENU SERVICES

Menu services allow you to create and manage menus for use either with the MENU command or with a user program. There are two components in menu services: the menu generator and the menu processor.

5.2.1. The Menu Generator

The menu generator allows you to create your own menus interactively. You use it to create new menus and modify or display existing menus. A menu generator session begins when you enter the MENUGEN command:

MENUGEN

You then see this screen:

```

MG01
MENU GENERATOR HOME SCREEN MENU
1. CREATE A NEW MENU MODULE
2. MODIFY AN EXISTING MENU MODULE
3. DISPLAY AN EXISTING MENU MODULE
4. END MENU GENERATOR
ENTER SELECTION NUMBER _
```

If, for example, you want to create a menu, choose 1. The menu generator then leads you by means of menus and screens through these steps:

1. You identify the menu you're creating. If, for instance, you are creating a menu named PAYMENU for use in system mode, you'd later call it with this command:

MENU PAYMENU

2. You tell the system in what file you wish to store the completed menu. In this example, you stored the menu named PAYMENU in \$\$FMT in SYSRES.
3. You create the screen that appears when you call the menu.
4. You create the action table which tells, for each item in the menu, what action the system takes, such as running a program or leaving the menu.

5. You create the help screens that go along with the menu.

At this point in the session, you can modify menu PAYMENU, go on to create other menus, or end the session. The menu is then ready for use.

NOTE:

When creating menus, remember that ESCORT cannot be called from a menu. Otherwise, you will encounter errors when using the menu.

5.2.2. The Menu Processor

In terms of menu services, the menu processor is the partner of the menu generator. As important as the menu generator is to creating a menu, the menu processor plays an equally important role by actually executing the menu. Every time you call a menu, whether by the MENU command or from a user program, it is the menu processor that searches for the menu, displays it on your workstation screen, accepts your choice, and tells the system what action to take in response to your choice. In addition, it retrieves and displays whatever help screen you may want.

You call the menu processor automatically when you enter a MENU command in system mode. If the menu is to be linked to a user program, however, some job control statements are necessary, most importantly the USE statement:

```
//[symbol] USE MENU { menu-file-LFD-1/menu-file-LFD-2
                      $YSFMT/menu-file-LFD-2
                      menu-file-LFD-1/$YSFMT
                      menu-file-LFD-1
                      $YSFMT }
[,initial-menu] { [,nnn] [,menu-1=alias-1...,menu-12=alias-12] }
```

where:

MENU

Indicates that the program is to use a menu.

```
{ menu-file-LFD-1/menu-file-LFD-2
  $YSFMT/menu-file-LFD-2
  menu-file-LFD-1/$YSFMT
  menu-file-LFD-1
  $YSFMT }
```

Names up to two files to be searched for menus. Any name you use must match an LFD name specified in a previously defined device assignment set for a menu library file (always a MIRAM file). A menu-file-LFD is one to eight alphanumeric characters long. If you don't specify anything for this parameter, it is assumed that all menus reside in system format file \$Y\$FMT. When coding this parameter, remember the following:

- If you omit \$Y\$FMT from \$Y\$FMT/menu-file-LFD, then code ./menu-file-LFD.

- ↓
- If you omit `$$FMT` from `menu-file-LFD/$$FMT`, then code `menu-file-LFD/`.

initial-menu

Specifies the name of the first or only menu to be used by the program. Because user programs do not themselves specify which menu is to be called to satisfy an input request, it falls to this parameter to specify the menu.

nnn

Specifies the number of menus to be resident in main storage at one time, in the range 1 to 255. The default value is 1.

menu-n=alias-n

Allows you to equate a menu name specified in an application program (alias) to a menu with a different menu name (given when the menu was created). A maximum of 12 alias name sets may be specified. The menu and alias names must each be from one to eight alphanumeric characters in length.

NOTE:

For complete information on creating and using menus, see the menu services concepts and facilities manual.

↑

5.3. THE OS/3 INTERACTIVE DATA UTILITIES

The OS/3 interactive data utilities enable you to interactively maintain the data files of your system. Through the data utilities, you can edit files, transfer files between peripherals, and compare files. Data utilities offer you the following functions:

- Reblocking files
- Correcting fields within a record
- Rearranging fields within a record
- Editing files while transferring them between peripherals
- Filing transfers between peripherals without editing
- Comparing files, with printouts of comparison disagreements

The correction function provides the following correction options:

- Selecting records for output
- Deleting records
- Inserting records
- Replacing records

5.3.1. Initializing the Interactive Data Utilities

The interactive data utilities are run as a user program, occupying a job slot. To initiate the data utilities, you must enter the RV command with the following parameters:

$$RV\Delta I@ DATA[(new-name)][,MEM=nnnnn][,ACT=acct-no][,DBG=\left\{\begin{matrix} Y \\ N \end{matrix}\right\}]$$

Parameters:

I@DATA

Is the job name that runs the interactive data utilities.

(new-name)

Allows you to specify another job name for the interactive data utilities. Each job name must be unique. If a new-name is not specified, the RV command will append a 2-digit number to the end of I@DATA to make the name unique. This permits more than one user to run interactive data utilities concurrently.

MEM=nnnnn

Specifies, in hexadecimal notation, the amount of main storage required for the data utilities function you wish to run. The default value is 8000¹⁶, (32,767¹⁰). The default value is adequate for many of the data utilities functions, but certain functions, chiefly those involving disk or tape files, could require more. There is a formula you may apply to calculate the amount of main storage you need for each of the data utilities functions. You can find the formula in the OS/3 data utilities user guide/programmer reference. If the amount of main storage you specify or the default value is insufficient for the function you wish to perform, the amount required for the function is displayed on the screen, and the data utilities terminates. You must then reinitialize data utilities and enter the correct amount of main storage.

ACT=acct-no

Specifies a 1- to 4-character alphanumeric account number.

DBG= $\left\{\begin{matrix} Y \\ N \end{matrix}\right\}$

Specifies that the data utilities run in the debugging mode. This is used chiefly to provide documentation for software user reports (SURs).

5.3.2. Using the Interactive Data Utilities

The data utilities operate interactively by presenting you with a series of menu selection screens. These screens enable you to enter the information needed by the data utilities function you wish to perform. When you enter the RUN/RV command to initialize the data utilities, your workstation screen will clear, and the first menu selection screen will be displayed. The first screen asks you what you want to do:

```
SCREEN1          DUS01
DO YOU WISH TO
1. COPY OR PRINT A FILE
2. COMPARE TWO FILES
3. CONVERT OS/4 FILES TO OS/3 DISK FILES
4. CONVERT A S/32-34 $ COPY DISKETTE
5. HELP
ENTER (1 THRU 5)  1
```

On the ENTER line, the numeral 1 occupies the space in which you enter your choice of 1, 2, 3, 4, or 5. The default value is 1 on this first screen. If you wish to specify 2, 3, or 4, you simply overwrite the 1. To send your choice to the data utilities program, press the transmit key.

The next screen displayed to you will be determined by your response to the first screen. In this way, the data utilities lead you through the preparation of the information needed to execute a particular function. If you let the default value, 1, on the first screen stand, choosing to copy or print a file, the next screen displayed would ask you to:

```
INPUT SCREEN     DUS02
PLEASE ENTER THE TYPE OF YOUR PRIMARY
FILE:
1. CARD
2. TAPE
3. DISKETTE
4. DISK
ENTER (1 THRU 4)  1
```

The data utilities continue to direct you, through the menu selection screens, until you have entered all the information needed to perform the function you have selected. The data utilities then inform you that the session is finished and the function you selected is being executed.

5.3.3. The HELP Function

As you probably noticed on the first menu selection screen, there was a third choice for entry, termed HELP. This option is available to you on many of the menu selection screens throughout the interactive data utilities. If you choose the HELP option, a display will appear on the screen, explaining the terms used in the menu selection

screen and the other choices available to you. The HELP screens can help you when you reach a point, while entering information, when you do not understand the choices offered you. The HELP screens can also act as a quick refresher for a section of the data utilities with which you are not completely comfortable. The following is an example of a HELP screen:

```
*** TAPE HELP SCREEN DUH0411 ***  
  
''VSN'' SPECIFIES THE VOLUME NUMBER (VOL1 LABEL) ON THE TAPE VOLUME TO  
BE USED AS YOUR PRIMARY/SECONDARY FILE. THE VOLUME SERIAL NUMBER  
UNIQUELY IDENTIFIES THE TAPE REEL TO THE OPERATING SYSTEM. IT IS  
WRITTEN INTERNALLY (ON THE TAPE SURFACE) AND POSSIBLY EXTERNALLY  
(GENERALLY ON A GUMMED LABEL). THE VSN CANNOT BE MORE THAN SIX  
ALPHANUMERIC CHARACTERS AND THE FIRST CHARACTER MUST BE ALPHABETIC.  
IF THERE ARE LESS THAN SIX, TRAILING BLANKS WILL BE ADDED ON THE RIGHT.  
NEED MORE (IF ANY) HELP SCREENS? (Y/N=CONTINUE NORMAL PROCESSING)Y
```

Notice that this screen is TAPE HELP SCREEN DUH0411. Choosing the HELP option on the menu selection screen to which this HELP screen is coordinated gets you actually three HELP screens, each adding to the explanation of the choice on the menu selection screen. To view the next screen, simply press the XMIT key. On the line that asks NEED MORE (IF ANY) HELP SCREENS?, the character Y, which causes the next HELP screen to appear, is the default value. When you have seen the last of each series of HELP screens, enter either Y or N and press the XMIT key to return to the menu selection screen.

NOTE:

This explanation of the OS/3 data utilities is presented to get you started in using the data utilities interactively or to act as a reminder of the process of their execution. For the information you need to make more extensive use of the data utilities, refer to the OS/3 data utilities user guide/programmer reference.

5.4. THE INTERACTIVE DUMP/RESTORE HARDWARE UTILITY (HU) ↓

The dump/restore hardware utility (HU) enables you to interactively initiate and control the DMPRST routine from your workstation. The DMPRST routine creates backup copies of your program and data libraries on disk, tape, or diskette. To initiate the DMPRST hardware utility, key in the following command:

HU

There are no parameters associated with this command. Once the system accepts your command, the first in a series of screens appears at your workstation. This first screen is a menu where you select the function you want to perform: ↑

↓

HARDWARE UTILITIES HU00A

1. DUMP FILES FROM A DISK
2. RESORE FILES TO A DISK
3. COPY FILES FROM A DISK TO DISK
4. COPY AND/OR VERIFY 8419 DISK
5. NONE OF THESE

ENTER SELECTION: __

Depending on which number you select, an appropriate set of screens is displayed. You simply enter the requested information on each screen and DMPRST does the rest.

You can request help with any screen by pressing function key 13. If the help you need extends over more than one screen, press the XMIT key to display the next screen. When the last help screen is displayed, press function key 14 or the XMIT key to return to your current screen.

For detailed information on the operation and use of the HU facility, refer to the system service programs user guide.

↑

5.5. THE GENERAL EDITOR

The general editor allows you to interactively edit data and library files, as well as edit and write source programs. To initialize the general editor, you enter the following command from the workstation:

EDT△[initial command]

↓

When the system has processed your command, it clears the workstation screen and displays the message EDITOR VERSION XX.X READY on the bottom of the screen. In addition, it sets the current work-space line to 1.0000 and displays a start-of-entry symbol (▷) to accept EDT commands. If you entered an editor command with the EDT command, the editor will process that command, and then display the appropriate line number based on the action of that command. Table 5-1 lists the commands available through the general editor, their formats, and brief descriptions of their functions.

↑

Table 5-1. General Editor Commands (Part 1 of 8)

| Command | Format | Explanation |
|---------------------|---|--|
| EDT Commands | | |
| <u>@</u> | $\left. \begin{array}{l} @ \{ \text{line-number} \text{ [increment]} \} \\ \left. \begin{array}{l} + \\ - \end{array} \right\} \left[\begin{array}{l} \{ \text{data} \\ \{ \text{command} \} \end{array} \right] \end{array} \right\}$ | Sets the current line number and increment for data and command lines keyed in at the workstation |
| <u>C</u> HANGE | @C ['search-string'[*n]] TO 'change-string'[*n] | Replaces an existing string in the current work-space file with a new string |
| <u>C</u> OPY | @CO [line-range]['search-string'[*n]] TO destination | Copies lines in the current work-space file to new line locations without deleting the original lines |
| <u>D</u> ELETE | @D [line-range]['search-string'[*n]] | Erases specified lines from the current work-space file |
| <u>F</u> IND | @FIN 'search-string'[*n] | Locates the first occurrence of a string in the work-space file and assigns its corresponding line number to the variable ? and the column numbers of the first and last columns it occupies to [and] respectively |
| <u>F</u> STATUS | To specify file parameters for any file for which you want a list of modules, use this format: @FS[MODULE=module-name] [,TYPE={module-type}] ,FILENAME= { filename 'filename' "filename" } [,RDPASS=password] ,VSN=volume [,DEVICE={ did DISK DISKETTE }] | Creates in the work-space file a list of all modules contained in a specified program library |
| <u>I</u> NSERT | @I 'change-string'[*n] | Inserts a specified string into lines in the current work-space file |
| <u>L</u> IST | @L [line-range]['search-string'[*n]][IMMEDIATE] | Prints specified lines from the current work-space file on the printer |
| <u>M</u> OVE | @M [line-range]['search-string'[*n]] TO destination | Transfers specified lines to new line locations in the work-space file and deletes the original lines and line numbers |

Table 5-1. General Editor Commands (Part 2 of 8)

| Command | Format | Explanation |
|----------------------------|--|---|
| EDT Commands (cont) | | |
| <u>N</u> UMBER | @NU 'sequence-string'[*n] [BY increment] | Inserts sequence numbers into input lines |
| <u>P</u> RINT | @P [line-range]['search-string'[*n]] | Displays specified lines from the current work-space file on the workstation screen |
| <u>P</u> UNCH | @PU [line-range]['search-string'[*n]][IMMEDIATE] | Reproduces specified lines from the current work-space file on cards |
| <u>R</u> EAD | <p>To read a SAT or MIRAM library module from disk or format label diskette to the current work-space file, use this format:</p> <pre>@READ MODULE=module-name [,TYPE={module-type}] [,TRUNC={YES } ,FILENAME={ filename 'filename' "filename" }] [,RDPASS=password],VSN=volume [,DEVICE={ did DISK DISKETTE }] Δ [[KEY=start-col-no:end-col-no KKEY=start-col-no:end-col-no SHOWΔfirst-col-no:last-col-no]]</pre> <p>To read a MIRAM data file from disk or format label diskette to the current work-space file, use this format:</p> <pre>@READ FILENAME={ filename } [,RDPASS=password] 'filename' "filename" , VSN=volume [,KEYNO={ n }] [,DEVICE={ did DISK DISKETTE }] [,BFSZ=n] [,TRUNC={ YES } NO }</pre> <p>To read a unit record file from a data set label diskette or from the card reader, use this format:</p> <pre>@READ FILENAME={ filename },VSN=volume 'filename' "filename" ,DEVICE={ did DISKETTE } [,TRUNC={ YES } RDR NO }</pre> | <p>Reads a copy of a library module or program library into the work-space file</p> |

Table 5-1. General Editor Commands (Part 3 of 8)

| Command | Format | Explanation |
|-------------------------------|--|--|
| EDT Commands (cont) | | |
| <p>READ (cont)</p> | <p>Δ [{ KEY=start-col-no:end-col-no KKEY=start-col-no:end-col-no SHOWΔfirst-col-no:last-col-no }]</p> <p>To read a file from a tape, use this format:</p> <p>@READ FILENAME= { filename } [, RDPASS=password] { 'filename' "filename" }</p> <p>, VSN=volume, DEVICE= { did } [, BKNO={ YES } { TAPE } { NO }]</p> <p>[, TRUNC={ YES }] Δ [{ KEY=start-col-no:end-col-no KKEY=start-col-no:end-col-no SHOWΔfirst-col-no:last-col-no }] { NO }</p> <p>To read a file from the spool file to the current work-space file, use this format:</p> <p>@READ [JOB=jobname] [, HOLD= { L N Y }]</p> <p>[, FILENAME= { filename }] [, ACCT=acct-no] { 'filename' "filename" }</p> <p>, QUEUE= { LOG PRINT PUNCH RDR } [, ALL={ YES }] [, SKIP={ n } { NO } { 1 }]</p> <p>[, TRUNC={ YES }] Δ [{ KEY=start-col-no:end-col-no KKEY=start-col-no:end-col-no SHOWΔfirst-col-no:last-col-no }] { NO }</p> <p>To read the same module or file last accessed through a previous @READ or @WRITE command, use this format:</p> <p>@READ</p> <p>To read the same module or file last accessed through a previous @READ or @WRITE command but read now with a KEY, KKEY, or SHOW parameter or any valid EDT command specified, use this format:</p> <p>@READΔ; Δ [{ KEY=start-col-no:end-col-no KKEY=start-col-no:end-col-no SHOWΔfirst-col-no:last-col-no }] [valid EDT command]</p> | |
| <p>REMOVE</p> | <p>@REM 'search-string'[*n]</p> | <p>Deletes a specified string from lines in the work-space file</p> |
| <p>SEQUENCE</p> | <p>@SEQ { 'sequence-string'[*n] } BY increment *</p> | <p>Inserts sequence numbers into existing lines in the current work-space file</p> |

Table 5-1. General Editor Commands (Part 4 of 8)

| Command | Format | Explanation |
|----------------------------|--|--|
| EDT Commands (cont) | | |
| <u>UPDATE</u> | @U [line-range] ['search-string' [*n]] | Displays specified lines from the work-space file one at a time for you to edit or change |
| <u>WRITE</u> | <p>To write the current work-space file to a SAT or MIRAM library module on a disk or format label diskette, use this format:</p> <pre>@WRITE MODULE=module-name [,TYPE={module-type}] ,FILENAME={filename} [,WRPASS=password] "filename" [,DEVICE={did DISK DISKETTE}] ,VSN=volume [,CONTIG={YES} NO}] [,INC={n}] [,RCSZ=n] [,SIZE=n] [,SAT={YES} NO}]</pre> <p>To write the current work-space file to a MIRAM data file on a disk or format label diskette, use this format:</p> <pre>@WRITE FILENAME={filename} [,WRPASS=password] "filename" ,VSN=volume [,CONTIG={YES} NO}] [,DEVICE={did DISK DISKETTE}] [,INC={n}] [,INIT={YES} NO}] [,EXTEND={YES} NO}] [,KEYi={start-col-no:end-col-no (start-col-no:end-col-no, {DUP} , {CHG})}] ,SIZE=n [,RCB={YES} NO}] [,RCFM={FIX} VAR}] ,RCSZ=n [,SCSZ={n} 256}] [,BFSZ=n]</pre> | Writes a copy of the current work-space file to: a program library or data file on disk, diskette, or tape, or to the spool file |

Table 5-1. General Editor Commands (Part 5 of 8)

| Command | Format | Explanation |
|--------------------------------|---|-------------|
| EDT Commands (cont) | | |
| <p>WRITE (cont)</p> | <p>To write the current work-space file to a unit record file (i.e., to the printer, card punch, or to a data set label diskette), use this format:</p> <pre>@WRITE FILENAME={filename } ,VSN=volume {'filename' } {"filename" }</pre> <pre>,DEVICE={did } [,RCFM={FIX }] [,RCSZ=n] {DISKETTE } {PRINT } {PUNCH }</pre> <p>To write the current work-space file to a tape, use this format:</p> <pre>@WRITE FILENAME={filename } [,WRPASS=password] {'filename' } {"filename" }</pre> <pre>,VSN=volume,DEVICE={did } [,BFSZ=n] [,BKNO={YES }] {TAPE } {NO }</pre> <pre>[,RCFM={FIXUNB }] [,RCSZ=n] [,INIT={YES }] {FIXBLK } {VARUNB } {VARBLK } {UNDEF }</pre> <pre>[,EXTEND={YES }] {NO }</pre> <p>To write the current work-space file to the spool file, use this format:</p> <pre>@WRITE [JOB=jobname] [,HOLD={YES }] {NO }</pre> <pre>[,FILENAME={filename }] [,ACCT=acct-no] {'filename' } {"filename" }</pre> <pre>,QUEUE={PRINT } [,COPIES={n }] {PUNCH } {RDR }</pre> <p>To write to the same module or file last accessed through a previous @READ or @WRITE command, use this format:</p> <pre>@WRITE</pre> <p>To write to the same module or file last accessed through a previous @READ or @WRITE command, but written to now with any valid EDT command specified, use this format:</p> <pre>@WRITEΔ;Δvalid EDT command</pre> | |

Table 5-1. General Editor Commands (Part 6 of 8)

| Command | Format | Explanation |
|---|---|---|
| General Editor Variable Commands | | |
| <u>A</u> SSIGN | @AS Gn= { 'string' [*n] n(x:y) n[±m] Gm LEN(n) } | Assigns values to EDT variables |
| <u>D</u> ISPLAY | @DI { 'string' [*n] n(x:y) n[±m] Gm LEN(n) } | Displays a specified expression or the value of a specified expression from the work-space file on the workstation screen |
| IF | @IF.condition.command or @IF expression relation expression command | Permits an EDT command or EDT procedure file command to be executed based on some condition |
| General Editor Procedure File Commands | | |
| <u>D</u> O | @DO proc-number { PRINT NOPRINT REVERT } | Executes a procedure file |
| <u>E</u> ND | @E | Terminates procedure file definition |
| <u>G</u> OTO | @G {label} {line} | Permits branching within a procedure file |
| <u>I</u> NP <u>T</u> | @INP file-parameters { PRINT NOPRINT REVERT } | Loads and executes a procedure file |
| <u>N</u> OP | @NOP [comment] | Enters extra lines for branching or comments into a procedure file |
| <u>P</u> ROC | @PRO [proc-number] | Begins procedure file definition |
| <u>R</u> ET <u>R</u> URN | @RET | Terminates procedure file execution |
| General Editor Directives | | |
| <u>C</u> HECK | @CHE { ON } { OFF } | Determines if processed lines are to be displayed on the workstation screen |
| <u>C</u> OBOL | @COB | Activates the COBOL editor |
| <u>D</u> ROP | @DR | Deletes all lines in the entire EDT work-space file |

Table 5-1. General Editor Commands (Part 7 of 8)

| Command | Format | Explanation |
|----------------------------------|--|---|
| General Editor Directives (cont) | | |
| EFP | @EFP | Activates the error file processor |
| FORMAT | @FORMAT parameter string (for RPGEDT) @FORMAT (for COBEDT) | Used only in conjunction with either RPGEDT or COBEDT. See the appropriate subeditor manual for information on the @FORMAT directive. |
| HALT | @H | Terminates the EDT session |
| RPG | @RPG | Activates the RPG II editor |
| SET | @S [CHAR=tab-character, IABS={columns}] [, LINE=length] [, EXCLUDE={exclusion-character}] [, ATSIGN=command-trigger] [, COLON=range-separator] [, ENCOL=end-column] [, BUFFER={record-size}] [, WIDTH=device-size] [, CLEAR] [, STRIP={ON}] [, DISPLAY] [, SCRDSPLY={TRUNCATE}] [, ROLL={15 (if SCRDSPLY=TRUNCATE); 8 (if SCRDSPLY=FOLD); 1-15}] [, MODE={LINE}] [, LANGUAGE={FREEFORM}] [, RECENTRY={SINGLE}] [, SCRFORM={UNDERLINE}] [, SCREEN] [, COBOL] [, RPG] [, BLANK] | Defines various parameters to EDT that collectively make up your EDT environment |
| SYSTEM | @SY [workstation-command] | Permits workstation commands to be issued during an EDT session or temporarily returns you to system mode |

Table 5-1. General Editor Commands (Part 8 of 8)

| Command | Format | Explanation |
|----------------------------|--------------------------|---|
| EDT Screen Commands | | |
| <u>B</u> LOCK | @BL | Displays a freeform screen that allows you to switch to block mode for entering multiple commands or data |
| <u>H</u> ELP | @HE [error message code] | Displays help screens for any EDT error messages |
| <u>P</u> ARAMS | @PA | Displays a screen showing the parameters on the @SET directive (those that make up your EDT environment) |
| <u>P</u> PROMPT | @PROM [edt command] | Displays the EDT command menu screen or help screens for any of the EDT commands (meaning EDT commands, modifiers, directives, procedure file commands, variables, and screen commands) |
| <u>R</u> ESTORE | @RES | Returns you to the point in your EDT session where you originally entered a screen command |
| <u>R</u> OLL | @RO | Displays freeform screens, showing the EDT work-space file, where you can update lines or simply view them |



5.5.1. The Error File Processor

The error file processor, or EFP, is a subroutine of the general editor. It lets you see errors in your source code immediately after the language compiler has compiled your program. You don't have to wait for the printed listing from the compiler. As you see your errors, you can correct them right at your workstation.

EFP has its own command set. Once it is running, it will display an error message together with the line of source code where the error occurs. You use the EFP commands to control this display. To correct the source code, you use regular EDT commands.

NOTE:

Before you can use EFP, you must specify an error file in your job control stream. You do this by specifying the // PARAM ERRFIL= job control statement, the RPG II jprocs, or the auto report jprocs. For details, see the user guide or programmer reference for the language you're using.

If you're not already using the general editor, you activate EFP by keying in:

```
EDT@EFP
```

If you are already in the general editor, clear the EDT workspace with the @DELETE command and set the current line number and increment equal to 1 with the @ command. Then key in:

```
@EFP
```

Once activated, EFP displays the following message at your workstation:

```
EFP001  VERSION n.n
EFP002  ENTER ERROR-FILE MODULE-NAME,FILE-NAME,VSN
```



The cursor is positioned on the line below the EFP002 message. There you enter the module name, file name, and volume serial number of your error file. Once you transmit this information, the next screen display is:

```
EFP003  ERROR FILE=error-module-name,error-filename,vsn
        language-compiler,compiler-version,compilation-date,compilation-time
EFP004  SOURCE FILE=source-module-name,source-filename,vsn
EFP005  MODULE=source-module-name                nnnn ERRORS
        or
        program-unit-names (for FORTRAN users only)
```





Lines 1 through 3 show you your error file library, information about your compilation, and your source file library. Line 4 shows you the total count of compilation errors.

At this point, you can begin displaying each error message and related line of source code by issuing an EFP command.

Table 5-2 lists the EFP commands and their functions.

Table 5-2. Error File Processor (EFP) Commands

| Command | Format | Explanation |
|----------------|--|---|
| <u>EFP</u> | <p>@EF[X]Δ[program-unit-name] Δ [error-range] Δ ['search-string]</p> <p>where:</p> <p>X</p> <p>program-unit-name</p> <p>error-range</p> <p>search-string</p> | <p>Displays error messages along with the line of source code where the error occurs</p> <p>Specifies that any message and line of source code that is displayed once will not be displayed again during the current EFP session</p> <p>(Applies only to FORTRAN IV source modules containing multiple program units) Lets you see and correct errors for one program unit within your source module</p> <p>Specifies the error message numbers you wish to see. They can be a range of numbers, or specific numbers.</p> <p>Specifies specific types of error messages you wish to see. You can specify up to 50 characters.</p> |
| | <p>@EFP SOU source-module-name, source-file-name,vsn</p> <p>where:</p> <p>source-module-name,source-file-name, and vsn</p> | <p>To correct and display FORTRAN IV errors for compilations that process multiple source modules</p> <p>Refer to the library where a specific source module that you want to correct is in your FORTRAN IV compilation.</p> |
| <u>END</u> | @EF END | Terminates EFP |
| <u>SUMMARY</u> | @EF SUM | Displays a summary of your error file |

↓
The following screens illustrate a typical EFP session.

```
EDT@EFP
```

You activate EDT and issue the EFP directive.

```
EFP001 VERSION 8.0
EFP002 ENTER ERROR-FILE MODULE-NAME, FILE-NAME, VSN
▶ ERRMOD, MYERRFIL, RES
```

EFP asks you for and you provide the name of your error file and where it resides.

```
EFP003 ERROR-FILE=ERRMOD, MYERRFIL, RES
COBOL74, VERSION 8.00/xx, 81/09/04, 10:27:04
EFP004 SOURCE FILE=PAYROLL, MYFILE, MYVOL1
EFP005 PAYROLL 10 ERRORS
```

EFP displays your error file's name and location, the name and location of your source file, information about your compilation, and the number of compilation errors.

```
101.0000 ▶ EFP
```

You issue an EFP command to begin display of your error messages and their related source code. (101.0000 is the next EDT line number because the source code for PAYROLL occupies the first 100 lines of the EDT workspace.)

```
ERR-001 ASSIGN CLAUSE NOT SPECIFIED IN SELECT SENTENCE.
10.0000 SELECT PAY-REPORT-FILE
102.0000▶ @U 10
103.0000▶ SELECT PAY-REPORT FILE ASSIGN TO PRINTER-UNITPR-VC
```

↑
The first error message and the line of source code it refers to are displayed (lines 1 and 2). On line 3, you issue the EDT update command to change line 10 of your source code. On line 4, you make your correction.

5.5.2. The RPG Editor

The RPG editor is a subeditor of EDT. It enables you to interactively create programs in the RPG II programming language. The RPG editor offers three format types to accommodate all levels of programming experience.

To initialize the RPG editor, enter the EDT command with the operand †RPG as follows:

```
EDT@RPG
```

When your command is processed, the screen will clear and you will receive the following display:

```

                RPGEDT VER #
SELECT MODE (C)
C = CREATE    U = UPDATE

SELECT FORMAT TYPE: (1)
  1 = POSITIONAL    2 = FORMATTED    3 = FREE FORM

SPECIFICATION TYPE DISPLAY? (N)  Y = YES  N = NO

```

The three format types are geared to different levels of programming ability. The *formatted* type provides the most prompting and is easiest to use. The following is a formatted-type screen:

```

                                                                    LINE - 1.0000
1 SEQUENCE NUMBER: _____ 6 FORM TYPE H
7 COMPILATION MODE: _          8 ERROR DUMP: _
9 OPERATOR CONTROL: _         15 DEBUG: _
21 INVERTED PRINT: _          26 ALTSEQ: _
31 BINARY SEARCH: _           40 SIGN HANDLING: _
41 FORMS ALIGNMENT: _         42 INDICATOR INIT.: _
43 FILE TRANSLATION: _        70 CCA NAME: _____
74 SUBROUTINE: _              75 PROGRAM ID: _____
NEXT SPECIFICATION TYPE, ST, OR CMD: (___)
-----
-----

```

The *positional* format type is for more experienced programmers; it offers less prompting and hence requires a more thorough knowledge of RPG II.

The following is an example of a positional format:

```

LINE - 1.0000
      1 2 2 3 4 4 4 4 7   7 7
1    6 7 8 9 5 1 6 1 0 1 2 3 0   4 5
----- H -----
NEXT SPECIFICATION TYPE, ST, OR CMD: (___)
-----
-----

```

The third format type, the *free-form* format, offers no prompting and is geared for highly experienced programmers who wish to create programs quickly. The following is an example of the free-form format:

```

LINE - 1.0000
      1      2      3      4      5      6
123456789012345678901234567890123456789012345678901234
-----
      1      1      1
6 7      8      9      0      1      2
567890123456789012345678901234567890123456789012345678
-----
ENTER ST, OR CMD: (___)
-----
-----

```

The procedure for using the RPG editor to create a program is the same for all three format types. The RPG editor displays the forms necessary to create an RPG II program. After you have completed one form, the next will be presented until a complete program is written.

For detailed information on the operation and use of the general editor and the RPG editor, refer to the general editor user guide/programmer reference and the RPG editor user guide/programmer reference.

5.5.3. The COBOL Editor (COBEDT)

The COBOL editor, or COBEDT, is a subeditor of the general editor. It allows you to interactively create programs in the COBOL programming language.

The COBOL editor offers you two modes of operation: the ordered creation mode and the selective creation mode. The ordered creation mode is designed for the novice programmer; it prompts you through certain steps in the creation process. The selective creation mode is for the more experienced user, who doesn't need prompting. COBEDT also allows you to use keyword abbreviations, either ones you define and/or ones COBEDT assigns.

To activate the COBOL editor, enter the EDT directive with the COBOL command as follows:

```
EDT@COBOL
```

The first screen displayed is the option select screen, where you select your mode of operation, specify whether you will use abbreviations, and choose your continuation mode.

```

OS/3  EDT/COBOL                                COBOL EDITOR (V8.0/1)
.....
Select Creation Mode: (2)
  1=Create in COBOL Program Order
  2=Create Selected Portions of the COBOL Program
Abbreviations to be used: (1)
  1=None  2=COBOL keywords  3=user specified  4=Both (COBOL and user)
Display COBOL keyword or abbreviation file abbreviations (1) 1=No 2=Yes
Abbreviation file to be read and/or written (1)
  1=No  2=Read file  3=Write  4=Read and write file
Enter file name (.....)
Enter file vsn (.....)
Continuation Code (NRM)
  NRM=Normal continuation  CMD=Enter command mode

EDT Command:.....
.....
.....

```


As you transmit each line of source code, the COBOL editor checks your syntax and notifies you of any errors. In this way, you can eliminate syntax errors before you compile your program.

For detailed information on the operation and use of the general editor and the COBOL editor, refer to the general editor user guide/programmer reference and the COBOL editor user guide/programmer reference.

5.6. DISTRIBUTED DATA PROCESSING (DDP)

Distributed data processing (DDP) permits you to form a network of data processing systems in which all systems may, within the limits of proper security, access each other's files and run jobs on each other while directing program output back to the remote system that originated the job.

The distributed data processing functions of OS/3 are initiated by entering the following command:

DDP△command

To perform the DDP functions, you use a special set of commands, given here in Table 5-3. Each command must be preceded by the DDP directive. For more information on DDP and more detailed information on the DDP commands, refer to the OS/3 distributed data processing concepts and facilities manual.

Table 5-3. DDP Commands (Part 1 of 2)

| Command | Explanation |
|---|--|
| <p>CREATE△FILE= [{host-id} ::= file-id {local-host-id}]</p> <p>[△BLOCK_SIZE= {nnnnnnnn} 256 }]</p> <p>[△DENSITY= { 200 556 800 1600 6250 host-SYSGEN-op-ton }]</p> <p>[△DEVICE_CLASS= { DISK TAPE DISKETTE }] [△FILE_TYPE= { SEQUENTIAL INDEXED LIBRARY UNDEFINED }]</p> <p>[△INCREMENT_SIZE= {nnnnnnnn} 3 cyl }] [△INITIAL_SIZE= {nnnnnnnn} 3 cyl }]</p> <p>[△KEY [- {n}] = (size, location [△ { DUPLICATES NO DUPLICATES }]) [△ { CHANGE NO CHANGE }])]</p> <p>[△PARITY= { ODD EVEN }] [△RECORD_FORM= { FIXED VARIABLE UNDEFINED }]</p> <p>[△RECORD_SIZE= {nnnnn} 256 }] [△REGISTER= { VTOC CATALOG }]</p> | <p>The CREATE command:</p> <ul style="list-style-type: none"> ■ Establishes a file on a receiving host ■ Allocates space for the file ■ Catalogs the file in your online system catalog ■ Records the file in the volume table of contents (VTOC) of the volume at the remote host on which the file is created <p>NOTE:</p> <p>The default for INCREMENT SIZE and INITIAL SIZE is three cylinders. If more or less than three cylinders is needed, the size must be entered in number of blocks (nnnnnnnn).</p> |
| <p>COPY△FROM= [{source-host-id} ::= source-file-id {local-host-id}]</p> <p>△TO= [{destination-host-id} ::= destination-file-id {local-host-id}]</p> <p>[△ELEMENT_TYPE= { SYMBOLIC RELOCATABLE ABSOLUTE MACRO PROC COMPILED_JOB SCREEN_FORMAT }]</p> <p>[△KEY [- {n}] = (size, location [△ { DUPLICATES NO DUPLICATES }]) [△ { CHANGE NO CHANGE }])]</p> <p>[△MODE= { DIRECT WAIT INDIRECT }] [△POSITION= { EOF SOF }]</p> <p>[△TRANSLATE= { ASCII EBCDIC NONE }]</p> | <p>The COPY command permits you to copy a file or module from one system to another. You may copy a file or module from one remote system to another, from your local system to a remote system, or vice versa. You may also use the COPY command to copy a file within your local system.</p> |

Table 5-3. DDP Commands (Part 2 of 2)

| Command | Explanation |
|--|---|
| <p>PURGE△FILE= [{host-id } ::] file-id [local-host-id]</p> | <p>The PURGE command allows you to physically remove a file, and all references to it, from a host system.</p> |
| <p>SUBMIT△FILE= [{source-host-id } ::] file-id [local-host-id]</p> <p>[△ELEMENT_TYPE= {SYMBOLIC } {COMPILED_JOB}]</p> <p>[△HOST= {destination-host-id } [local-host-id]]</p> | <p>The SUBMIT command allows you to send a file of job control streams to a host system for execution. You can also use it to initiate a file of job control streams already at the host system or to bring a job control stream to your local system for execution.</p> |
| <p>CANCEL△JOB= [{host-id } ::] jobname [local-host-id]</p> <p>[△OUTPUT= {DISCARD }] △COMMAND= {host-id } [DELIVER } [local-host-id] work-order-number</p> | <p>The CANCEL command allows you to terminate a job either executing or scheduled for execution on a host system.</p> |
| <p>SUBMIT△REQUEST=statement [△HOST= {host-id } [local-host-id]]</p> | <p>The SUBMIT REQUEST command allows you to send a statement, such as an operator or interactive command, to a host system. However, the following statements (commands) cannot be used: DISPLAY, DELETE, BREAKPOINT, FILE, IN, SU, TU, PD.</p> |
| <p>STATUS△ { COMMAND=work-order-number FILE= [{HOST-id } ::] file-id [local-host-id] [keyword parameter] HOST=host-id JOB= [{host-id } ::] jobname [local-host-id] USER= [{host-id } ::] user-id [local-host-id] }</p> | <p>The STATUS command enables you to obtain information about:</p> <ul style="list-style-type: none"> ■ Commands entered ■ Host systems in your DDP system ■ Jobs you have submitted ■ Files in your DDP system ■ Other users on your DDP system |
| <p>TALK△MESSAGE='string'</p> <p>△USER= [{host-id } ::] { OPERATOR } [△WAIT] [local-host-id] [user-id]</p> | <p>The TALK command allows you to send a message to a remote operator or user.</p> |

5.7. BASIC

BASIC is a powerful, interactive programming language that you use through your workstation. BASIC programs may be written, executed, and modified from the workstation.

To use BASIC, enter the following command:

```
BASIC
```

There are no parameters associated with this command.

When BASIC is ready for your use, you receive the following message:

```
BA001 OS/3 BASIC READY(VER x.x) BEGIN
```

You may now begin to enter the BASIC statements that comprise your program. Each line is checked for correct syntax as it is entered.

NOTE:

For more detailed information on programming in BASIC, refer to the OS/3 BASIC programmer reference manual.

5.8. ESCORT

ESCORT is an interactive programming language that uses English statements to create a program. ESCORT allows you to generate reports and perform inquiry and update routines through the use of simple, sentence-like programs, entered through your workstation. To use ESCORT, log on and then enter the following command:

```
ESCORT
```

There are no parameters associated with this command. The following display now appears on your workstation screen:

```
Welcome to an ESCORT to computers  
Please select your entry point: _  
  
1. The PROGRAM mode  
2. The TUTORIAL mode  
3. HELP - a brief description of ESCORT  
4. RECOVER
```

For the inexperienced ESCORT user, we recommend the *tutorial* mode of operation, which makes ESCORT easy to learn right at the workstation. The following screen is an example of the ESCORT tutorial displays:

```
*** ESCORT TUTORIAL SESSION ***  
A TUTORIAL SESSION IS PRIMARILY A QUESTION AND ANSWER  
TYPE OF ENVIRONMENT IN WHICH YOU RESPOND BY SELECTING  
ONE OF SEVERAL AVAILABLE OPTIONS. OCCASIONALLY YOU ARE  
REQUIRED TO PROVIDE BRIEF TEXT ENTRIES.  
BEGIN BY IDENTIFYING THE PRIMARY PROGRAM FUNCTION:  
1. ENTER DATA FROM WORKSTATION  
2. CHANGE DATA  
3. RETRIEVE DATA (SELECT)  
4. DELETE DATA  
5. SORT DATA  
+6. HELP - EXPLANATION OF OPTIONS  
* ENTER SELECTION NUMBER..._
```

ESCORT also offers help screens to explain the various statements and conventions it requires. For more detailed information on the use of ESCORT, refer to the ESCORT user guide.



PART 4. APPENDIXES



Appendix A. Interactive Terminals

A.1. GENERAL

This appendix contains the specific information you need to use the System 80 workstation, the System 80 console workstation, or the UNISCOPE and UTS 400 terminals as workstations. The different procedures for each terminal are also presented in a table at the end of this appendix.

NOTE:

The system console of a Series 90 system cannot be used as a workstation.

A.2. THE SYSTEM 80 WORKSTATION

The System 80 workstation keyboard contains data keys and control keys. Data keys are the keys you use to enter alphabetic, numeric, and punctuation characters (the same characters you find on a regular typewriter). Control keys either control keyboard operation or they perform a system function. As you will see, some keys are used both as data keys and as control keys.

Figures A-1 and A-2 show you the 70-character TYPEWRITER keyboard and the 94-character EXPANDED TYPEWRITER keyboard, respectively. Both are available with the System 80 workstation. Table A-1 highlights the most frequently used control keys and describes what each one does. For more information on both keyboards, see the System 80 local workstation operator's guide, UP-8910 (current version).

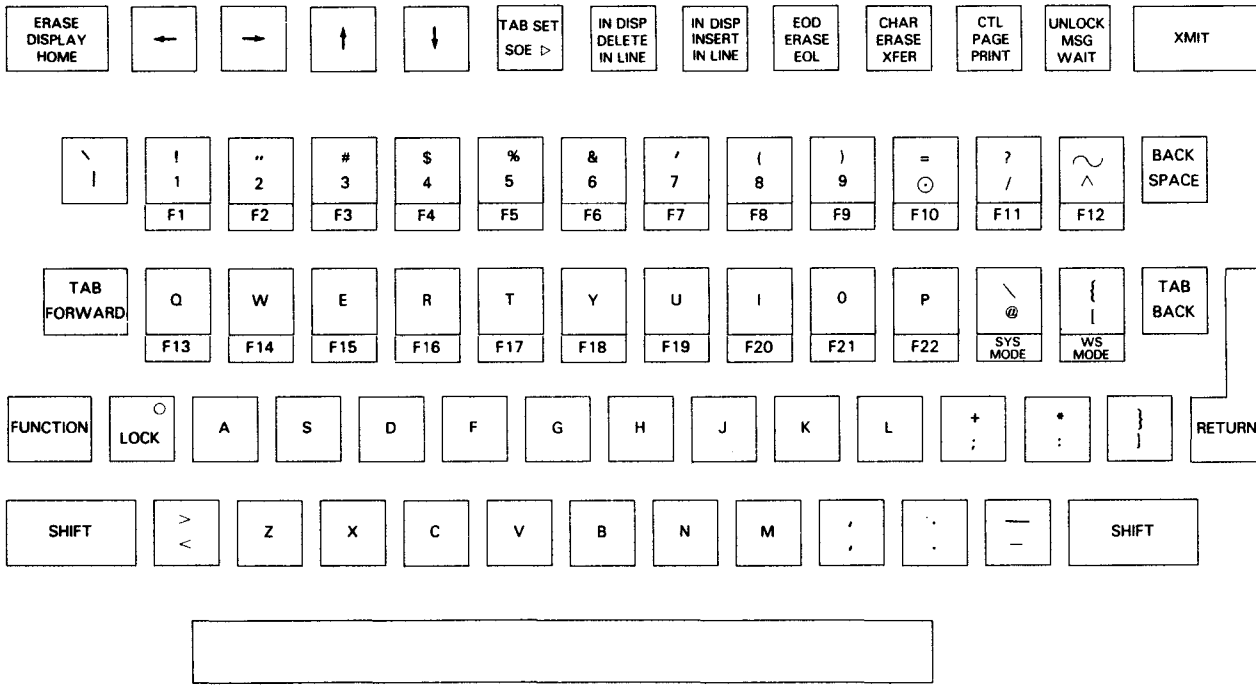


Figure A-1. Workstation TYPEWRITER Keyboard

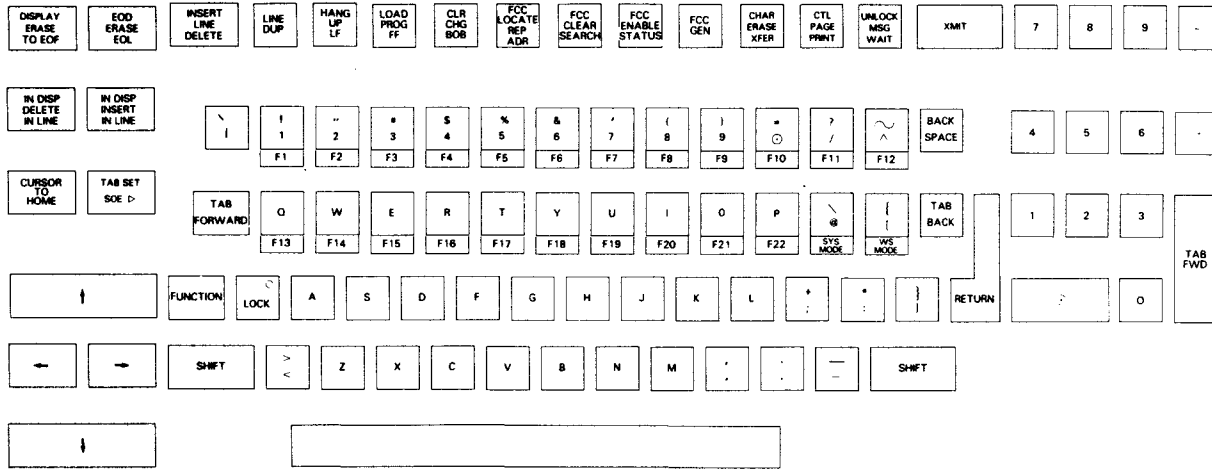


Figure A-2. Workstation EXPANDED TYPEWRITER Keyboard

Table A-1. System 80 Workstation Keyboard Functions (Part 1 of 4)

| Key | Function | | | | | | | | | | | | | | | | | | |
|-------------------------------------|------------------------|--------------------|----------|------------|---------------|------------------------|------------------------|------------------|-------------------|----------------|-----------------|----------------|-----------------|---------|------------|---|---|---|---------|
| TYPEWRITER KEYBOARD | | | | | | | | | | | | | | | | | | | |
| ERASE DISPLAY HOME | ← | → | ↑ | ↓ | TAB SET SOE ▷ | IN DISP DELETE IN LINE | IN DISP INSERT IN LINE | EOD ERASE EOL | CHAR ERASE XFER | CTL PAGE PRINT | UNLOCK MSG WAIT | XMIT | | | | | | | |
| ⌨ | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 | F11 | F12 | BACK SPACE | | | | | | |
| TAB FORWARD | Q | W | E | R | T | Y | U | I | O | P | SYS MODE | WS MODE | TAB BACK | | | | | | |
| FUNCTION | LOCK | A | S | D | F | G | H | J | K | L | * | * | RETURN | | | | | | |
| SHIFT | > | Z | X | C | V | B | N | M | , | - | - | - | SHIFT | | | | | | |
| [Empty Row] | | | | | | | | | | | | | | | | | | | |
| EXPANDED TYPEWRITER KEYBOARD | | | | | | | | | | | | | | | | | | | |
| DISPLAY ERASE TO EOF | EOD ERASE EOL | INSERT LINE DELETE | LINE DUP | HANG UP LF | LOAD PRG PF | CLR CHG ROB | TEK LOCATE REP ADM | FCC CLEAR SEARCH | FCC ENABLE STATUS | FCC GEN | CHAR ERASE XFER | CTL PAGE PRINT | UNLOCK MSG WAIT | XMIT | 7 | 8 | 9 | . | |
| IN DISP DELETE IN LINE | IN DISP INSERT IN LINE | ⌨ | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 | F11 | F12 | BACK SPACE | 4 | 5 | 6 | . |
| CURSOR TO HOME | TAB SET SOE ▷ | TAB FORWARD | Q | W | E | R | T | Y | U | I | O | P | SYS MODE | WS MODE | TAB BACK | 1 | 2 | 3 | TAB FWD |
| ↑ | FUNCTION | LOCK | A | S | D | F | G | H | J | K | L | * | * | RETURN | 0 | 0 | 0 | 0 | 0 |
| ← | → | SHIFT | > | Z | X | C | V | B | N | M | , | - | - | SHIFT | | | | | |
| ↓ | [Empty Row] | | | | | | | | | | | | | | | | | | |

Table A-1. System 80 Workstation Keyboard Functions (Part 2 of 4)

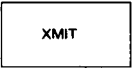











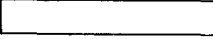
| Key | Function |
|--|---|
| Control Keys that Perform a System Function | |
|  | <p>SENDING DATA TO THE SYSTEM - Pressing the transmit key (XMIT) sends your entry to the system. The entry can be a command, a response, a screen, or any information pertaining to one of the interactive utilities, such as the general editor.</p> |
|  | <p>ACTIVATING THE UPPER OR FRONT-FACE OF A KEY - Certain control keys have dual functions, which are printed on their top, and certain data keys have a control function printed on their front. To activate the upper function of a control key or the front face of a data key, press the FUNCTION key and, while holding it down, press the desired key. (For an example, see the SYS MODE key.</p> |
|   | <p>ENTERING SYSTEM MODE - You may issue interactive commands only in SYSTEM mode. To enter SYSTEM mode, press the FUNCTION key and, while holding it down, press the SYS MODE key.</p> |
|   | <p>ENTERING WORKSTATION MODE - You complete the LOGON menu and fill in data only in WORKSTATION mode. To enter WORKSTATION mode, press the FUNCTION key and, while holding it down, press the WS MODE key.</p> |
|     | <p>USING FUNCTION KEYS - Function keys perform various system functions, depending on whether you've assigned an interactive command to them (via the DEFKEY command) or whether an interactive utility has assigned a function to them. Appendix E lists the function keys already assigned by the system. To use a function key, press the FUNCTION key and, while holding it down, press the desired function key.</p> |
| Control Keys that Control the Cursor | |
|  | <p>TAB FORWARD - Pressing the TAB FORWARD key advances the cursor one tab setting.</p> |
|  | <p>TAB BACK - Pressing the TAB BACK key moves the cursor back one tab setting.</p> |
|  | <p>SPACE BAR - Pressing the space bar moves the cursor forward one position.</p> |

Table A-1. System 80 Workstation Keyboard Functions (Part 3 of 4)
















| Key | Function |
|---|---|
| Control Keys that Control the Cursor (cont) | |
|  <p style="text-align: center;">OR</p>  | <p>HOME or CURSOR TO HOME keys - Pressing the HOME key (on the TYPEWRITER keyboard) or the CURSOR TO HOME key (on the EXPANDED TYPEWRITER keyboard) repositions the cursor to the upper left-hand corner of the screen.</p> |
|     | <p>CURSOR SCAN KEYS</p> <p>Pressing the forward arrow moves the cursor forward one position. If the cursor is at the end of a line, it will move to the beginning of the next line.</p> <p>Pressing the back arrow moves the cursor back one position. If the cursor is at the beginning of a line, it will move to the last character position of the previous line.</p> <p>Pressing the up arrow moves the cursor up one line without changing its horizontal position.</p> <p>Pressing the down arrow moves the cursor down one line without changing its horizontal position.</p> <p>NOTE:</p> <p style="padding-left: 40px;">Holding down any cursor scan key causes the cursor to move continuously until the key is released.</p> |
|  | <p>BACK SPACE KEY - Pressing the BACK SPACE key moves the cursor back one position.</p> |
|  | <p>RETURN KEY - Pressing the RETURN key moves the cursor to the beginning of the next line.</p> |
|  | <p>CHARACTER ERASE KEY - To erase a character (replace it with a blank), place the cursor on the character, press the FUNCTION key and, while holding it down, press the CHAR ERASE key. (You can also set the space bar so that it will erase a character and leave a blank in its place. For details, see the SCREEN command, 4.2.8.)</p> |
|  | <p>DELETING A CHARACTER IN A LINE - To delete a character in a line, place the cursor on the character to be deleted and press the DELETE IN LINE key. All characters on the same line to the right of the deleted character are moved left one position.</p> |
|  | <p>INSERTING A CHARACTER IN A LINE - To insert a character in a line, place the cursor on the character position immediately following the point of insertion. Press the INSERT IN LINE key. This moves all subsequent characters on the same line right one position. Press the key for the character you're inserting.</p> |

Table A-1. System 80 Workstation Keyboard Functions (Part 4 of 4)

| Key | Function |
|--|--|
| Control Keys that Control the Cursor (cont) | |
|  | ERASING CHARACTERS TO THE END OF A LINE - Pressing the erase-to-the-end-of-line key, ERASE EOL, erases all the characters you entered from the cursor position to the end of the line or the end of the field, whichever occurs first. If you are using a screen display that contains protected characters, the protected characters remain intact. |
|  <small>(EXPANDED TYPEWRITER KEYBOARD ONLY)</small> | DELETING A LINE - On the EXPANDED TYPEWRITER keyboard, pressing the DELETE LINE key removes the line in which the cursor is located. All lines following the deleted line are moved up one line, and a blank line is inserted at the bottom of the screen. The cursor remains in the same position after the DELETE LINE key is pressed. |
|  +  <small>(EXPANDED TYPEWRITER ONLY)</small> | INSERTING A LINE - On the EXPANDED TYPEWRITER keyboard, pressing the FUNCTION key and, while holding it down, pressing the INSERT LINE key, insert a blank line in front of the line where the cursor is located. All lines below the inserted blank line are moved down one line. The cursor remains in the same position as before the INSERT LINE key is pressed. |

NOTE:

The CTL PAGE and UNLOCK key functions present on the System 80 workstation keyboard are not necessary to the normal functioning of the workstation. Depressing these keys could cause unpredictable results in commands or programs being executed when they are depressed.

A.2.1. LOGON Procedure

After applying power to the workstation, the screen displays the results of the Power-On Confidence test (POC). When the POC is complete, the word LOADING will flash on the indicator line. Then, the display shown in Figure A-3 appears on your screen.

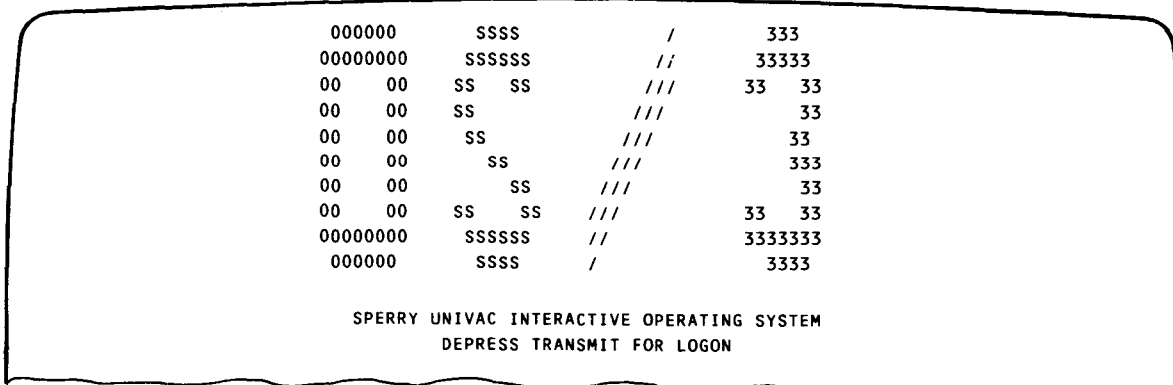


Figure A-3. OS/3 LOGON Request Screen

When this display appears, you may switch to SYSTEM mode and enter the LOGON command, or request the LOGON menu by pressing the TRANSMIT (XMIT) key.

A.2.2. LOGOFF Procedure

When you have completed your tasks using the workstation, enter the LOGOFF command as described in 2.7. After the LOGOFF ACCEPTED message is displayed, the OS/3 LOGON request screen described earlier is displayed. It remains displayed until the workstation screen is blanked, the workstation is turned off, or another user enters a LOGON command.

A.3. UNISCOPE 100 AND 200 TERMINALS

The UNISCOPE 100 and 200 terminals have identical keyboards, as shown in Figure A-4. Table A-2 highlights frequently used special function keys (control keys) and describes each one. For more information on the UNISCOPE terminal keyboard, see the UNISCOPE display terminal operator reference, UP-7788 (current version).

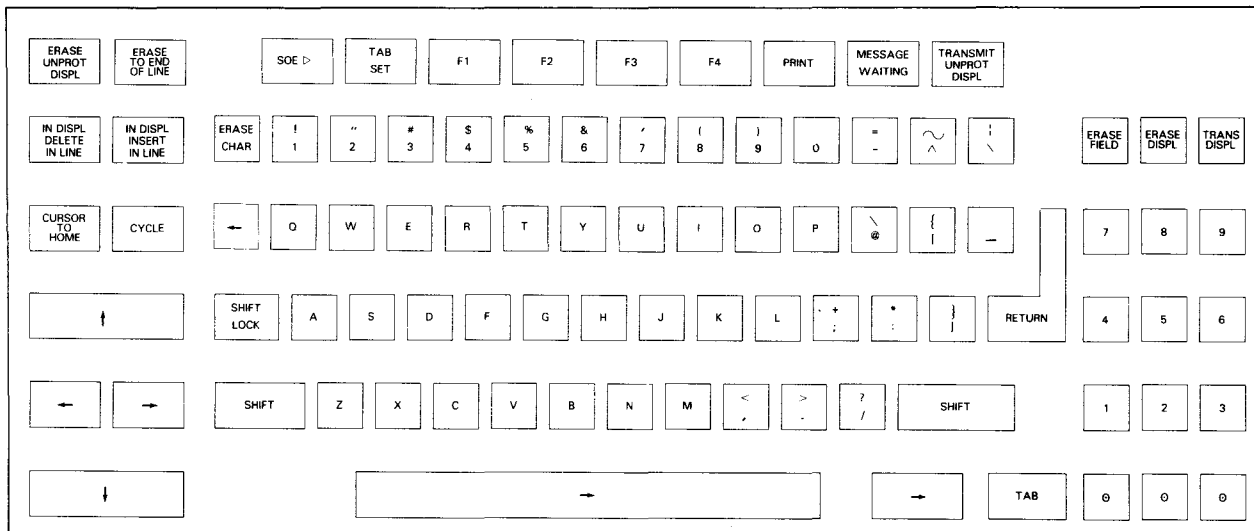


Figure A-4. UNISCOPE Terminal Keyboard

NOTE:

Be careful when you respond to a message; removing the two lines of data for SYSTEM mode could cause serious program errors when you return to WORKSTATION mode.

Table A-2. UNISCOPE 100 and 200 Terminal Keyboard Functions (Part 1 of 3)

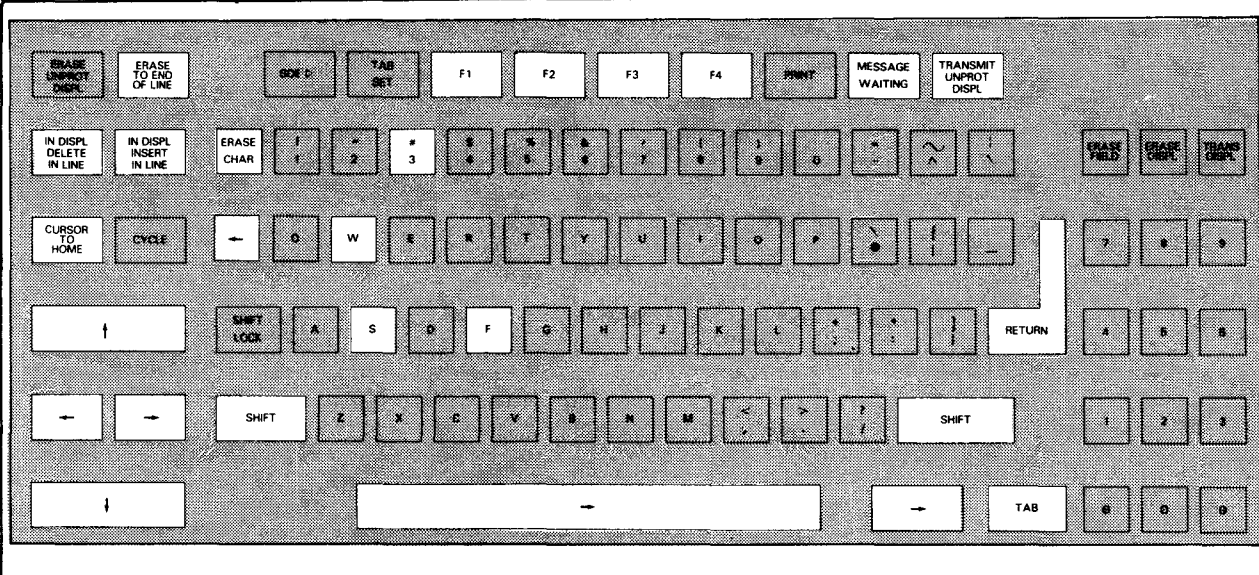

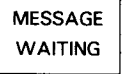
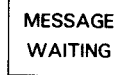


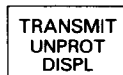
| Key | Function |
|---|---|
| UNISCOPE 100 and 200 Terminal Keyboard | |
|  <p>The diagram shows a terminal keyboard layout. At the top, there are function keys: ERASE UNPROT DISPL, ERASE TO END OF LINE, HOME, TAB SET, F1, F2, F3, F4, PRINT, MESSAGE WAITING, and TRANSMIT UNPROT DISPL. Below these are rows of alphanumeric keys with various symbols and characters. On the right side, there are keys for ERASE FIELD, ERASE CHAR, and ERASE DISPL. The bottom row includes a TAB key and numeric keys 6, 0, and 9.</p> | |
| Control Keys that Perform a System Function | |
|  | <p>SENDING DATA TO THE SYSTEM - Pressing the transmit key TRANSMIT UNPROT DISPL sends your entry to the system. The entry may be a command, a response, a screen, or any information pertaining to one of the interactive utilities, such as the general editor.</p> |
|  | <p>ENTERING SYSTEM MODE - You may issue workstation commands only in system mode. To enter system mode, press the MESSAGE WAITING key.</p> |
|     | <p>ENTERING WORKSTATION MODE - You may obtain the LOGON menu and fill in data only in workstation mode. To enter workstation mode, press the MESSAGE WAITING key, the W key, the S key, then the TRANSMIT key.</p> |

Table A-2. UNISCOPE 100 and 200 Terminal Keyboard Functions (Part 2 of 3)

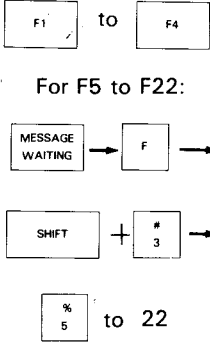



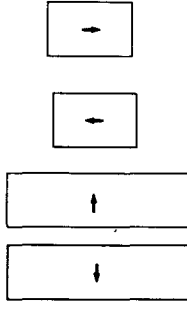
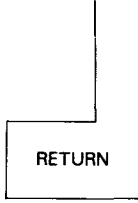
| Key | Function |
|--|--|
| Control Keys that Perform a System Function (cont) | |
|  <p>F1 to F4</p> <p>For F5 to F22:</p> <p>MESSAGE WAITING → F →</p> <p>SHIFT + # 3 →</p> <p>% 5 to 22</p> | <p>USING FUNCTION KEYS – UNISCOPE terminals have four function keys available on the keyboard: F1, F2, F3, and F4. To simulate function keys F5 through F22, press the MESSAGE WAITING key, the F key, the pound symbol (#), then the function key number you wish to use. To simulate the MESSAGE WAITING key, enter system mode by pressing the MESSAGE WAITING key, then type MSG.</p> |
| Control Keys that Affect Screen Display and Position the Cursor | |
|  | <p>TAB FORWARD – Pressing the TAB key advances the cursor one tab setting.</p> |
|  | <p>SPACE BAR – Pressing the space bar moves the cursor forward one position.</p> |
|  | <p>CURSOR TO HOME – Pressing the CURSOR TO HOME key repositions the cursor to the upper left-hand corner of the screen.</p> |
|  | <p>CURSOR SCAN KEYS</p> <p>Pressing the forward arrow moves the cursor forward one position. If the cursor is at the end of a line, it will move to the beginning of the next line.</p> <p>Pressing the back arrow moves the cursor back one position. If the cursor is at the beginning of a line, it will move to the last character position of the previous line.</p> <p>Pressing the up arrow moves the cursor up one line without changing its horizontal position.</p> <p>Pressing the down arrow moves the cursor down one line without changing its horizontal position.</p> <p>NOTE:</p> <p>Holding down any cursor scan key causes the cursor to move continuously until the key is released.</p> |
|  | <p>RETURN KEY – Pressing the RETURN key moves the cursor to the beginning of the next line.</p> |



Table A-2. UNISCOPE 100 and 200 Terminal Keyboard Functions (Part 3 of 3)

| Key | Function |
|---|--|
| Control Keys that Affect Screen Display and Position the Cursor (cont) | |
| ERASE CHAR | ERASING A CHARACTER - To erase a character (replace it with a blank), place the cursor on the character and press the ERASE CHAR key. |
| IN DISPL DELETE IN LINE | DELETING A CHARACTER IN A LINE - To delete a character in a line, place the cursor on the character to be deleted and then press the DELETE IN LINE key. All characters on the same line to the right of the deleted character are moved left one position. |
| IN DISPL INSERT IN LINE | INSERTING A CHARACTER IN A LINE - To insert a character in a line, place the cursor on the character position immediately following the point of insertion. Press the INSERT IN LINE key. This inserts a blank to the left of the cursor position and moves all subsequent characters on the same line right one position. Move the cursor left one position. Press the desired character key. |
| ERASE TO END OF LINE | ERASING CHARACTERS TO THE END OF A LINE - Pressing the ERASE TO END OF LINE key erases all unprotected characters (that is the data you entered) from the cursor position to the end of the line or the end of the field, whichever occurs first. |

A.3.1. Resuming Data Output

When your UNISCOPE terminal screen is full, the MESSAGE WAITING light turns on and the audible alarm sounds, just as when the system has a message for you. In fact, the system *does* have a message for you, informing you that the screen is full and asking whether you want to continue receiving data. Enter SYSTEM mode; the following message is displayed:

```
CONTINUE?
WORKSTATION DATA MODE FULL
```

As with other messages, this is displayed on the top two lines of the screen, causing the removal of any data present on those two lines. To respond, simply press the TRANSMIT key. The cursor is correctly positioned to the character position after the question mark on the top line of the screen.

A.3.2. LOGON Procedure

In order to log on to OS/3 interactive services from a UNISCOPE terminal, you must first sign on to ICAM through the standard terminal dialog of the ICAM terminal support facility. See A.5 for complete information on the standard terminal dialog. When your sign-on to ICAM has been accepted, you will receive the following message:

SESSION PATH OPEN

When you receive this message, you may either go into SYSTEM mode and enter the LOGON command, or press TRANSMIT to request the LOGON menu.

A.3.3. LOGOFF Procedure

When you have completed your tasks and wish to end your session of using the interactive services, enter the LOGOFF command as described in 2.7. After the LOGOFF ACCEPTED message is displayed, the OS/3 LOGON request screen is displayed. At this point, another user may log on to the system from your terminal. If you wish to terminate the communications connection of the terminal, you must perform the standard terminal sign-off command, \$\$\$OFF. See A.5 for complete information on the sign-off command.

A.4. UTS 400 TERMINAL

The UTS 400 terminal keyboard contains keys (special function keys) in addition to the keys found on a regular typewriter keyboard. Figure A-5 illustrates the UTS 400 terminal. Table A-3 highlights the control keys and describes what each one does. For more information on the UTS 400 terminal keyboard, see the Universal Terminal System 400 operator reference, UP-8358 (current version).

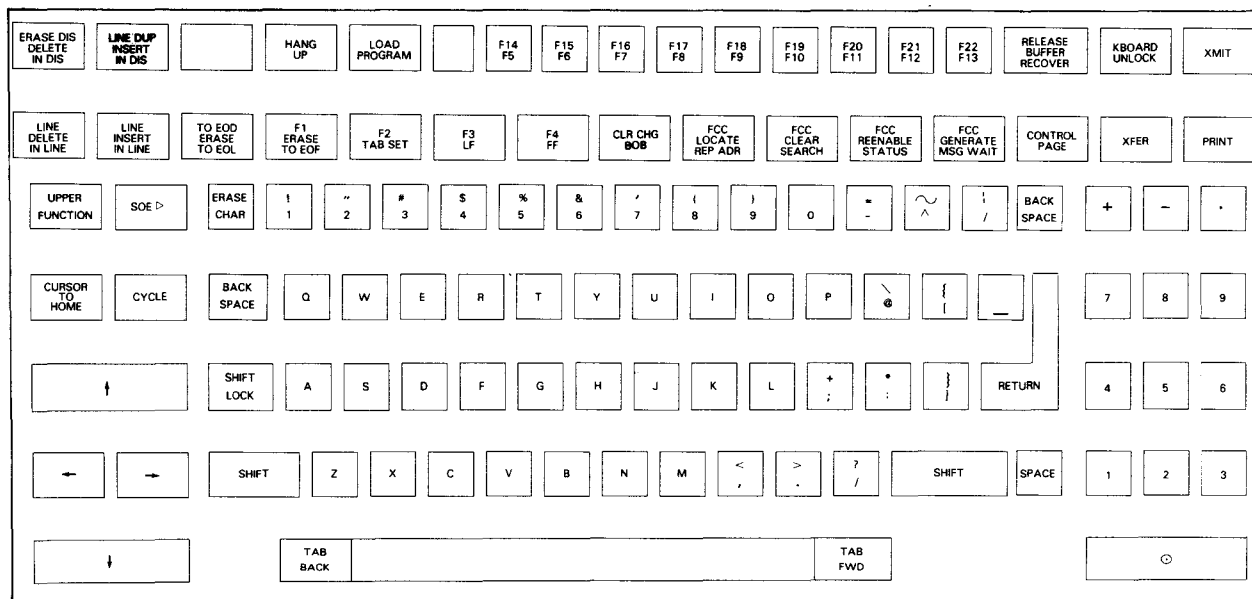


Figure A-5. UTS 400 Keyboard

Table A-3. UTS 400 Terminal Keyboard Functions (Part 1 of 3)

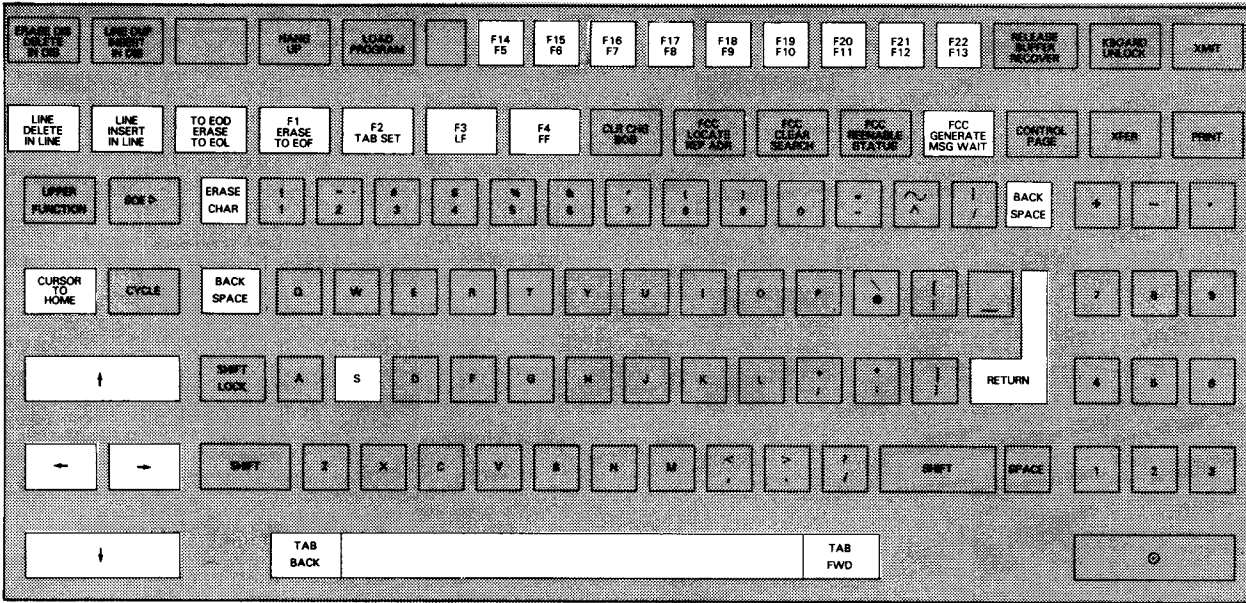

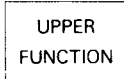
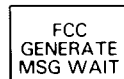
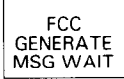


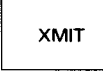
| Key | Function |
|---|---|
| UTS 400 Keyboard | |
|  <p>The diagram shows a keyboard layout with the following labels: ERASE OR DELETE IN CR, LINE SUPP W/OUT IN CR, HOLD UP, LOAD PROGRAM, F14 F5, F15 F6, F16 F7, F17 F8, F18 F9, F19 F10, F20 F11, F21 F12, F22 F13, RELEASE BUFFER RECOVER, ESCAPE/LOCK, XMIT, LINE DELETE IN LINE, LINE INSERT IN LINE, TO EOD ERASE TO EOL, F1 ERASE TO EOP, F2 TAB SET, F3 LF, F4 FF, CLR CR 900, FCC LOCATE W/ 404, FCC CLEAR SEARCH, FCC MESSAGE STATUS, FCC GENERATE MSG WAIT, CONTROL PAGE, XFER, PRINT, UPPER FUNCTION, ROE P, ERASE CHAR, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, BACK SPACE, CURSOR TO HOME, CYCLE, BACK SPACE, B, W, E, R, T, Y, U, I, O, P, SHIFT LOCK, A, S, D, F, G, H, J, K, L, RETURN, 4, 5, 6, SHIFT, Z, X, C, Y, B, N, M, SPACE, 1, 2, 3, TAB BACK, TAB FWD, and a power button.</p> | |
|  | <p>SENDING DATA TO THE SYSTEM - Pressing the transmit key, XMIT, sends your entry to the system. The entry can be a command, a response, a screen, or any information pertaining to one of the interactive utilities, such as the general editor.</p> |
|  | <p>ACTIVATING THE UPPER FUNCTION OF A KEY - Certain control keys have dual functions, which are printed on their top, and certain data keys have a control function printed on their front. To activate the upper function of a control key or the front face of a data key, press the UPPER FUNCTION key and, while holding it down, press the desired key.</p> |
|  | <p>ENTERING SYSTEM MODE - You may issue interactive commands only in SYSTEM mode. To enter SYSTEM mode, press the MSG WAIT key.</p> |
|  | <p>ENTERING WORKSTATION MODE - You obtain the LOGON menu and fill in data only in WORKSTATION mode. To enter WORKSTATION mode, press the MSG WAIT key, the W key, the S key, and the XMIT key.</p> |
|  | |
|  | |
|  | |

Table A-3. UTS 400 Terminal Keyboard Functions (Part 2 of 3)



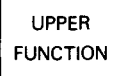
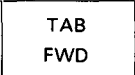
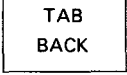







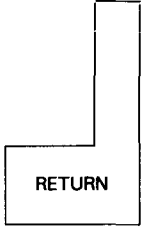





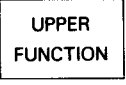

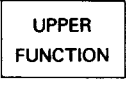
| Key | Function |
|---|---|
| UTS 400 Keyboard (cont) | |
|  <p style="margin-left: 40px;">to</p>  <p style="margin-left: 40px;">+</p>  | <p>USING FUNCTION KEYS - The UTS 400 terminals have the same number of function keys as the System 80 workstation. Some are lowercase function keys; the rest are uppercase function keys. To use the lowercase function keys, press the function key you wish to use. To use the uppercase function keys, press the UPPER FUNCTION key and, while holding it down, press the function key you wish to use.</p> |
|  | <p>TAB FORWARD KEY - Pressing the TAB FWD key advances the cursor one tab setting.</p> |
|  | <p>TAB BACK KEY - Pressing the TAB BACK key moves the cursor back one tab setting.</p> |
|  | <p>SPACE BAR - Pressing the space bar moves the cursor forward one position.</p> |
|  | <p>CURSOR TO HOME KEY - Pressing the CURSOR TO HOME key repositions the cursor to the upper left-hand corner of the screen.</p> |
|     | <p>CURSOR SCAN KEYS</p> <p>Pressing the forward arrow moves the cursor forward one position. If the cursor is at the end of a line, it will move to the beginning of the next line.</p> <p>Pressing the back arrow moves the cursor back one position. If the cursor is at the beginning of a line, it will move to the last character position of the previous line.</p> <p>Pressing the up arrow moves the cursor up one line without changing its horizontal position.</p> <p>Pressing the down arrow moves the cursor down one line without changing its horizontal position.</p> <p>NOTE:</p> <p style="margin-left: 40px;">Holding down any cursor scan key causes the cursor to move continuously until the key is released.</p> |
|  | <p>BACK SPACE KEY - Pressing the BACK SPACE key moves the cursor back one position.</p> |

Table A-3. UTS 400 Terminal Keyboard Functions (Part 3 of 3)

| Key | Function |
|--|---|
| UTS 400 Keyboard (cont) | |
|  | <p>RETURN KEY - Pressing the RETURN key moves the cursor to the beginning of the next line.</p> |
|  | <p>ERASING A CHARACTER - To erase a character (replace it with a blank), place the cursor on the character and press the ERASE CHAR key.</p> |
|  | <p>DELETING A CHARACTER IN A LINE - To delete a character in a line, place the cursor on the character to be deleted and press the DELETE IN LINE key. All characters on the same line to the right of the deleted character are moved left one position.</p> |
|  | <p>INSERTING A CHARACTER IN A LINE - To insert a character in a line, place the cursor in the character position immediately following the point of insertion. Press the INSERT IN LINE key. This inserts a blank to the left of the cursor position and moves right one position all subsequent characters on the same line. Move the cursor left one position. Press the desired character key.</p> |
|  | <p>ERASING CHARACTERS TO THE END OF A LINE - Pressing the ERASE TO EOL key erases all unprotected characters (that is, data you entered) from the cursor position to the end of the line or the end of the field, whichever occurs first.</p> |
|   | <p>DELETING A LINE - Pressing the UPPER FUNCTION key and, while holding it down, pressing the DELETE LINE key removes the line in which the cursor is located. All lines following the deleted line are moved up one line, and a blank line is inserted at the bottom of the screen. The cursor remains in the same position after the DELETE LINE key is pressed.</p> |
|   | <p>INSERTING A LINE - Pressing the UPPER FUNCTION key and, while holding it down, pressing the INSERT LINE key insert a blank line in front of the line where the cursor is located. All lines below the inserted blank line are moved down one line. The cursor remains in the same position as before pressing the INSERT LINE key.</p> |

A.4.1. Resuming Data Output

The procedure for resuming data output on the UTS 400 terminal is identical with that for the UNISCOPE 100 and 200 terminals.

A.4.2. LOGON Procedure

The LOGON procedure for the UTS 400 terminal is identical with that for the UNISCOPE 100 and 200 terminals. You must first connect the terminal through ICAM, and then log on.

A.4.3. LOGOFF Procedure

The LOGOFF procedure for the UTS 400 terminal is the same as for UNISCOPE terminals.

A.5. SYSTEM 80 CONSOLE WORKSTATION

Figure A-6 shows two available models of the console workstation keyboard. The control keys are the same as for the System 80 workstation keyboards. See Table A-1 for details.

A.5.1. Mode Switching

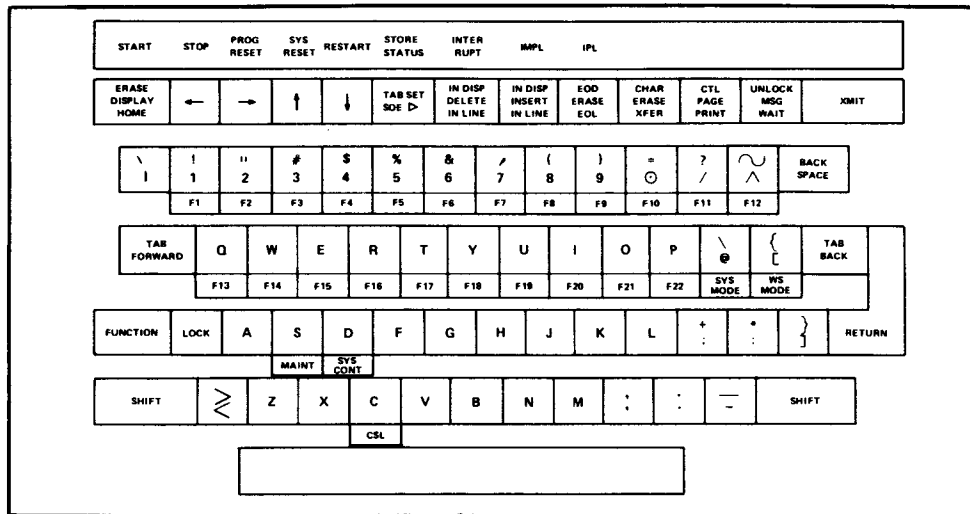
Switching between WORKSTATION and SYSTEM modes is handled in the same way as on the System 80 workstation. To switch from either mode to a console function mode, press the FUNCTION key and, while holding it down, press alphabetic C, D, or S for the appropriate console mode. If you are using the console workstation as a workstation (operating in either SYSTEM or WORKSTATION mode) and an informational message(s) develops concerning a console function, the following is displayed on the console workstation indicator line:

```
CNSMSG
```

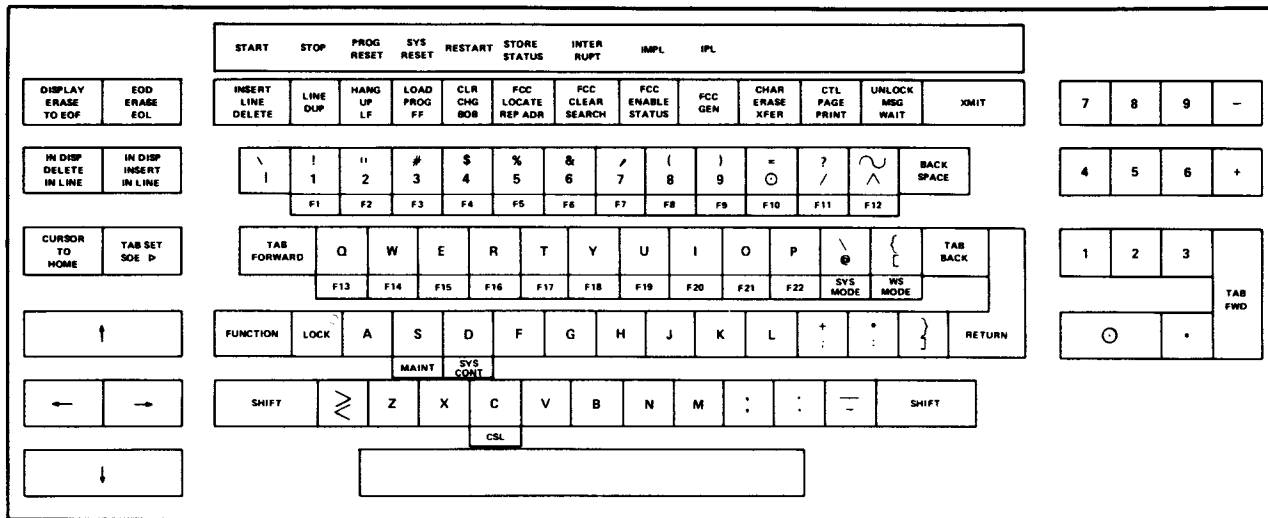
To view the message(s), press the FUNCTION key and, while holding it down, press alphabetic C. The message(s) is displayed on the second line of the screen, and CNSMSG is extinguished from the indicator line.

NOTE:

The CTL PAGE and UNLOCK key functions present on the System 80 console workstation keyboard are not necessary to the normal functioning of the console workstation. Depressing these keys could cause unpredictable results in commands or programs being executed when they are depressed.



Console Workstation Keyboard A



Console Workstation Keyboard B

Figure A-6. Console Workstation Keyboards

If a message develops concerning a console function that requires a response or if more than 14 informational messages accumulate, the following is displayed on the console workstation indicator line:

CNSREQ

The TRANSMIT key is also locked, preventing you from performing any workstation functions until you have responded to the console message. When you enter console mode, the message is displayed and CNSREQ is extinguished from the indicator line. After informational messages are displayed, they roll up and off the screen.

If, prior to entering console mode, you had begun a job in which the console workstation is used as a workstation, informational messages concerning the workstation job could develop. When this occurs, the following is displayed on the console workstation indicator line:

WSMSG

To view the message(s), press the FUNCTION key and, while holding it down, press the SYS MODE key. The message is displayed on the second line of the screen, and WSMSG is extinguished from the indicator line. If a message develops concerning the workstation job that requires a response, the following is displayed on the console workstation indicator line:

WSREQ

Note that the console functions are *not* locked by the appearance of this indicator. To view the message and respond to it, press the FUNCTION key and, while holding in down, press the SYS MODE key. The message is displayed and WSREQ is extinguished from the indicator line. Since console functions are not locked by this indicator, response messages may roll off the screen before you have a chance to answer them. Should this happen, use the REBUILD command (2.5.4) to redisplay them.

A.5.2. LOGON Procedure

If you wish to perform any workstation functions from the console workstation, you must first log on.

To log on when you are in any of the three console modes, press the FUNCTION key and, while holding it down, press the WS MODE key. At this point, you may press the TRANSMIT key to obtain the LOGON menu, or you may enter SYSTEM mode to enter the LOGON command. You may also enter SYSTEM mode directly from any of the console modes.

A.5.3. LOGOFF Procedure

The LOGOFF procedure for the System 80 console workstation is the same as for the System 80 workstation.

NOTE:

When you are using the console workstation as a workstation (either in SYSTEM or WORKSTATION mode), you are subject to the same restrictions as other workstation users. You may only control jobs either initiated or running under the user-id you logged on with.

A.6. INFORMATION FOR ICAM-CONNECTED TERMINALS

In order to use the interactive facilities from a remote terminal, you must first connect the terminal to your system through the ICAM (Integrated Communications Access Method) terminal support facility. Since terminal identifications and file names vary from system to system, you should consult your system administrator for particular information about your system. The following sections provide information on the standard terminal dialog, the \$\$\$ON, and \$\$\$OFF commands provided by ICAM for connection of remote terminals.

A.6.1. Standard Terminal Dialog

The ICAM terminal support facility provides a command processor module to process terminal operator requests to dynamically establish or end a communications session. The terminal operator enters sign-on or sign-off commands when it is necessary to communicate with a user program, a process file, or another terminal in the network, or to end a session with one of these. Standard terminal dialog is available with global networks that provide dynamic session establishment.

When the command processor receives a valid sign-on command, the message:

```
SPERRY-UNIVAC DCA NETWORK, LEVEL I,I, NODE ID xxxx
```

is sent to the initiating terminal, immediately.

where:

I,I

Is the operating system release level.

xxxx

Is the global network node identifier you specified in the TYPE operand of the CCA macroinstruction in your network definition.

If the sign-on request is honored, the message:

```
SESSION PATH OPEN
```

is sent to the initiating terminal; and if the session is between terminals, the two messages are sent to the requested terminal.

If the request for session establishment is rejected, the message:

```
SESSION PATH CLOSED
```

is sent to the initiating terminal.

If the request for sign-off (of an established session) is issued by another terminal or a user program, the messages:

```
SESSION PATH CLOSED  
$$SOFF
```

are sent to your terminal following session disestablishment.

If your program aborts an established session, the message:

```
SESSION PATH ABORTED
```

is sent to the affected terminal.

If an invalid command is entered at a terminal, the message:

```
INVALID $$ COMMAND
```

is returned to the terminal.

A.6.2. SIGN-ON Command (\$\$SON)

This command enables you to establish a session with a user program, a process file, or another terminal.

Format:

```
$$SON xxxxyyyy
```

where:

xxxx

Is the logical name of the terminal from which you issue this command; i.e., the label of the TERM macroinstruction with which you defined the terminal in your network definition.

yyyy

Is the logical name of:

- a user program (as defined in the label of a LOCAP macroinstruction you defined in your network definition or the APPS operand of an NATTACK interface macroinstruction);
- a process file (as defined in the label of a PRCS macroinstruction); or
- another terminal (as defined in the label of a TERM macroinstruction).

A.6.3. SIGN-OFF Command (\$\$SOFF)

This command requests dynamic session disestablishment. You use it to end a dynamic session.

If your terminal is connected to ICAM by means of a dial line and you issue this command, ICAM holds the telephone connection for 60 seconds after it has ended (disestablished) the session. Thus, you can issue a subsequent sign-on command to establish another session without needing to redial.

Format:

\$\$SOFF

There are no parameters associated with this command.

A.7. PROCEDURE TABLE

Table A-4 summarizes the various procedures involved in communicating with the interactive services, and how each type of terminal performs each procedure.

Table A-4. Procedure Table

| Procedure | Workstation | UNISCOPE Terminal | UTS 400 | System 80 Console Workstation |
|---------------------------------|-------------------------------------|--|---|-------------------------------------|
| \$\$\$SON | NO | YES | YES | NO |
| LOGON | YES | YES | YES | YES |
| SYSTEM mode | Press FUNCTION and SYS MODE keys. | Press MESSAGE WAITING key. | Press MSG WAIT key. | Press FUNCTION and SYS MODE keys. |
| WORK-STATION mode | Press FUNCTION and WS MODE keys. | Press MESSAGE WAITING key. Then, press W then S keys; then press TRANSMIT. | Press MSG WAIT key. Then, press W then S keys; then press TRANSMIT. | Press FUNCTION and WS MODE keys. |
| Function keys | Press FUNCTION and F1-F22 keys. | Press MESSAGE WAITING key. Then, press F1-F4; for rest, press F, 5-22. | Press F1-F22 UPPER FUNCTION when required. | Press FUNCTION and F1-F22 keys. |
| Message waiting indicator | SYS MSG displayed on indicator line | MESSAGE WAITING light is lit. Audible alarm sounds | MESSAGE WAITING light is lit. Audible alarm sounds | SYS MSG displayed on indicator line |
| Save and restore SYS mode lines | YES | NO | NO | YES |
| LOGOFF | YES | YES | YES | YES |
| \$\$\$SOFF | NO | YES | YES | NO |

NOTE:

You do not enter the \$\$\$SON command to log on from a direct-connected System 80 workstation. However, to connect the System 80 workstation to a program such as the Information Management System (IMS), which requires the use of the Integrated Communications Access Method (ICAM), you must enter \$\$\$SON after logging on normally. At the conclusions of the session, you must enter \$\$\$SOFF before logging off the workstation.



Appendix B. Workstation Commands

For quick reference, all the commands you may enter from a workstation are listed here.

| |
|--|
| <p>ALLOCATE△(ST), FILENAME= {filename } [, RDPASS=password] [, WRPASS=password]</p> <p style="margin-left: 20px;"> IR IS DA SQ NI MI </p> <p style="margin-left: 20px;"> { 'filename' } { "filename" } </p> <p style="margin-left: 20px;"> , VSN=volume [CONTIG= {YES}] [INC= {n}], SIZE=n { NO } </p> |
| <p>ASK△[user-id], 'text'</p> |
| <p>Rescheduling All Jobs or Jobs in a Particular Job Queue:</p> <p>BEGIN△JBQ [{H }] { N } { P }</p> |
| <p>Rescheduling Individual Jobs:</p> <p>BEGIN△jobname</p> |
| <p>Releasing Held Spooled Files:</p> <p>BEGIN△SPL, [{LOG }] [, ACCT=acctno] [, CART=cartridge-name] [, DEV= {768 }] { PRINT } { PUNCH } { RDR }</p> <p style="margin-left: 200px;"> { 770 } { 773 } { 776 } { 778 } { 789 } { 9300 } </p> <p style="margin-left: 20px;"> [, FILE=filename] [, FORM=formname] [, JOB=jobname] [, STEP=stepno] </p> |
| <p>BRKPT△ {P } [, ACCT=acctno] [, CART=cartridge-name] { I } { PU }</p> <p style="margin-left: 20px;"> [, DEV= {768 }] [, FILE=filename] [, FORM=formname] [, JOB=jobname] { 770 } { 773 } { 776 } { 778 } { 789 } { 9300 } </p> |



Copying Spool Files:

```

COPYΔ[JOB=jobname] [ ,HOLD={L} ] [ ,FILENAME={filename } ] [ ,ACCT=acct ], QUEUE={LOG }
                               {N}                               { 'filename' }
                               {Y}                               { "filename" }
                               {PRINT }
                               {PUNCH }
                               {RDR }

[ ,ALL={YES } ] [ ,SKIP={n} ] ΔTOΔ[JOB=jobname] [ ,HOLD={N} ] [ ,FILENAME={filename } ]
                               {NO }                {0}                               { 'filename' }
                                               {Y}                               { "filename" }

, QUEUE={PRINT } Δ[NUMBER][ ,HEX ][ ,WAIT ]
        {PUNCH }
        {RDR }

```



Copying Tape Files:

```

COPYΔ [ ,FILENAME={filename } ] [ ,RDPASS=password ], VSN=volume, DEVICE={did }
                               { 'filename' }
                               { "filename" }
                               {TAPE }

[ ,BKNO={YES } ] ΔTOΔ
  {NO }

[ ,FILENAME={filename } ] [ ,WRPASS=password ], VSN=volume, DEVICE={did }
  { 'filename' }
  { "filename" }
  {TAPE }

[ ,INIT={YES } ] [ ,EXTEND={YES } ]
  {NO }           {NO }

[ ,BESZ=n ] [ ,BKNO={YES } ] [ ,RCFM={FIXUNB } ] [ ,RCSZ=n ] Δ[NUMBER][ ,HEX ][ ,WAIT ]
  {NO }
  {FIXBLK }
  {VARUNB }
  {VARBLK }
  {UNDEF }

```

Copying Unit Record Files:

```

COPYΔDEVICE={did } , FILENAME={filename } , VSN=volume
             {DISKETTE }
             {RDR }
             { 'filename' }
             { "filename" }

ΔTOΔDEVICE={did } [ ,RCFM={FIX } ] [ ,RCSZ=n ]
            {DISKETTE }
            {PRINT }
            {PUNCH }
            {VAR }

, FILENAME={filename } , VSN=volume Δ[NUMBER][ ,HEX ][ ,WAIT ]
  { 'filename' }
  { "filename" }

```


DISPLAY△LOG

DISPLAY△SPL [, $\left\{ \begin{array}{l} \text{LOG} \\ \text{PRINT} \\ \text{PUNCH} \\ \text{RDR} \end{array} \right\}$] [, ACCT=acctno] [, CART=cartridge-name] [, DEV= $\left\{ \begin{array}{l} 768 \\ 770 \\ 773 \\ 776 \\ 778 \\ 789 \\ 9300 \end{array} \right\}$]

[, FILE=filename] [, FORM=formname] [, JOB=jobname] [, STEP=stepno]

DLOAD△(program-name)
 $\left\{ \begin{array}{l} /END \\ /OFFLINE \end{array} \right\}$

EDT△[initial command]

To Run a Command Stream from a Library File:

ENTER△MODULE=modulename [, TYPE= $\left\{ \begin{array}{l} \text{module-type} \end{array} \right\}$] , FILENAME= $\left\{ \begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right\}$

[, RDPASS=password] , VSN=volume

To Run a Command Stream from a Spooled File:

ENTER△ [, HOLD= $\left\{ \begin{array}{l} N \\ Y \end{array} \right\}$] [, FILENAME= $\left\{ \begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right\}$] , QUEUE=RDR

To Run a Command Stream from a DSL Diskette:

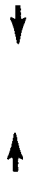
ENTER△ [FILENAME= $\left\{ \begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right\}$] [, RDPASS=password] , VSN=volume [, DEVICE= $\left\{ \begin{array}{l} \text{did} \\ \text{DISKETTE} \end{array} \right\}$]

To Run a Command Stream from a Tape:

ENTER△ [FILENAME= $\left\{ \begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right\}$] [, RDPASS=password] , VSN=volume [, DEVICE= $\left\{ \begin{array}{l} \text{did} \\ \text{TAPE} \end{array} \right\}$]

To Run a Command Stream from a Card Reader:

ENTER△ [DEVICE= $\left\{ \begin{array}{l} \text{did} \\ \text{RDR} \end{array} \right\}$]



Erasing a Library Module:

ERASE△MODULE=modulename [,TYPE= $\left\{ \begin{matrix} \text{module-type} \\ \blacksquare \end{matrix} \right\} \right]$,FILENAME= $\left\{ \begin{matrix} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{matrix} \right\}$
[,WRPASS=password],VSN=volume

Erasing Library and MIRAM Files:

ERASE△FILENAME= $\left\{ \begin{matrix} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{matrix} \right\}$ [,WRPASS=password],VSN=volume

ESCORT

EXECUTE Program Name

FILE $\left\{ \begin{matrix} ([did],label) \\ (RDR,label) \end{matrix} \right\}$ △ $\left[\begin{matrix} \left(\begin{matrix} \text{alt-filename} \\ \left(\begin{matrix} \text{alt-filename,} \left(\begin{matrix} \text{RES} \\ \text{RUN} \\ \text{vsn} \end{matrix} \right) \end{matrix} \right) \end{matrix} \right) \\ \left(\begin{matrix} \text{alt-filename,} \left(\begin{matrix} \text{RES} \\ \text{RUN} \\ \text{vsn} \end{matrix} \right) \text{ write-pass} \end{matrix} \right) \end{matrix} \right]$

FREE

ESTATUS△[MODULE=modulename] [,TYPE= $\left\{ \begin{matrix} \text{module-type} \\ \blacksquare \end{matrix} \right\} \right]$,FILENAME= $\left\{ \begin{matrix} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{matrix} \right\}$
[,RDPASS=password],VSN=volume△[LONG]

GO△jobname

HELP△ $\left\{ \begin{matrix} \text{command} \\ \text{message-no} \\ \text{keyword-parameter} \end{matrix} \right\}$

Holding All Jobs or All Jobs on a Particular Job Queue:

HOLD△JBQ $\left[\begin{matrix} \blacksquare \\ \text{H} \\ \text{N} \\ \text{P} \end{matrix} \right]$

Holding an Individual Job:

HOLD△jobname

| |
|---|
| <p>Holding Spooled Files:</p> <p>HOLDΔSPL, { PRINT PUNCH LOG RDR }, [ACCT=acctno][, CART=cartridge-name] [, DEV={ 768 770 773 776 778 789 9300 }]</p> <p>[, FILE=filename][, FORM=formname][, JOB=jobname][, STEP=stepno]</p> |
| <p>LOGONΔuser-id[, acct][, password][, exec-pro] [, BULLETIN={ NO YES }] [, LOG={ NO YES }]</p> |
| <p>LOGOFF</p> |
| <p>MENUΔ [menu-name] [{ filename, vsn SYSTEM { RES } vsns }]</p> |
| <p>MENUGEN</p> <p>OCL { ([did], label) } Δ [jobname-library-unit ([new-name])] [, { PRE HIGH NONE }]</p> <p>[, key-1=val-1, ..., key-n=val-n]</p> <p>OVΔ jobname-library-unit ([new-name]) [, { PRE HIGH NONE }] [, key-1=val-1, ..., key-n=val-n]</p> |
| <p>PAUSE Δ jobname</p> |
| <p>{ PR } Δ [function-code] [, ACCT=acctno] [, CART=cartridge-name] { PU }</p> <p>[, FILE=filename][, FORM=formname] [, JOB=jobname]</p> |
| <p>Printing a SAT or MIRAM Library Module:</p> <p>PRINTΔMODULE=modulename [, TYPE={ module-type }] , FILENAME={ filename 'filename' "filename" }</p> <p>[, RDPASS=password], VSN=volume [, COPIES={ n }] Δ [DIRECT] [, NUMBER] [, HEX] [, WAIT]</p> |
| <p>Printing a MIRAM Data File:</p> <p>PRINTΔFILENAME={ filename } [, RDPASS=password], VSN=volume { 'filename' } { "filename" }</p> <p>[, KEYNO={ n }] [, COPIES { n }] [, DEVICE={ did DISKETTE DISK }]</p> <p>Δ [DIRECT] [, NUMBER] [, HEX] [, WAIT]</p> |

Printing a Spool File:

$\text{PRINT} \Delta [\text{JOB}=\text{jobname}] \left[\text{HOLD}=\begin{cases} \text{L} \\ \text{N} \\ \text{Y} \end{cases} \right] \left[\text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases} \right] [\text{ACCT}=\text{acct}]$
 $\left[\text{QUEUE}=\begin{cases} \text{LOG} \\ \text{PRINT} \\ \text{PUNCH} \\ \text{RDR} \end{cases} \right] \left[\text{ALL}=\begin{cases} \text{YES} \\ \text{NO} \end{cases} \right] \left[\text{COPIES}=\begin{cases} \text{n} \\ \text{1} \end{cases} \right] \left[\text{SKIP}=\begin{cases} \text{n} \\ \text{0} \end{cases} \right] \Delta [\text{DIRECT}] [\text{NUMBER}] [\text{HEX}] [\text{WAIT}]$

Printing a Tape File:

$\text{PRINT} \Delta \left[\text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases} \right] [\text{RDPASS}=\text{password}], \text{VSN}=\text{volume}, \text{DEVICE}=\begin{cases} \text{did} \\ \text{TAPE} \end{cases}$
 $\left[\text{BKNO}=\begin{cases} \text{YES} \\ \text{NO} \end{cases} \right] \Delta [\text{DIRECT}] [\text{NUMBER}] [\text{HEX}] [\text{WAIT}]$

Printing a Unit Record File:

$\text{PRINT} \Delta \text{DEVICE}=\begin{cases} \text{did} \\ \text{DISKETTE} \\ \text{RDR} \end{cases}, \text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases}, \text{VSN}=\text{volume}$
 $\Delta [\text{DIRECT}] [\text{NUMBER}] [\text{HEX}] [\text{WAIT}]$

Punching a Library Module:

$\text{PUNCH} \Delta \text{MODULE}=\text{modulename} \left[\text{TYPE}=\begin{cases} \text{module-type} \\ \text{S} \end{cases} \right], \text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases}$
 $[\text{RDPASS}=\text{password}], \text{VSN}=\text{volume} \left[\text{COPIES}=\begin{cases} \text{n} \\ \text{1} \end{cases} \right] \Delta [\text{DIRECT}] [\text{WAIT}]$

Punching a MIRAM File:

$\text{PUNCH} \Delta \text{FILENAME}=\begin{cases} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{cases} [\text{RDPASS}=\text{password}], \text{VSN}=\text{volume} \left[\text{KEYNO}=\begin{cases} \text{n} \\ \text{0} \end{cases} \right]$
 $\left[\text{COPIES}=\begin{cases} \text{n} \\ \text{1} \end{cases} \right] \left[\text{DEVICE}=\begin{cases} \text{did} \\ \text{DISKETTE} \\ \text{DISK} \end{cases} \right] \Delta [\text{DIRECT}] [\text{WAIT}]$

Punching a Spool File:

PUNCH△[JOB=jobname] [HOLD={L
N
Y}] [FILENAME={filename
'filename'
"filename"}] [ACCT=acct]
[QUEUE={LOG
PRINT
PUNCH
RDR}] [ALL={YES
NO}] [COPIES={n
NO}] [SKIP={n
NO}]△[DIRECT][,WAIT]

Punching a Tape File:

PUNCH△[FILENAME={filename
'filename'
"filename"}] [,RDPASS=password],VSN=volume
[DEVICE={did
TAPE}] [BKNO={YES
NO}]△[DIRECT][,WAIT]

Punching a Unit Record File:

PUNCH△DEVICE={did
DISKETTE
RDR} [FILENAME={filename
'filename'
"filename"}] ,VSN=volume
△[DIRECT][,WAIT]

REBUILD

RECALL△ {LAST△nn
hh:mm:ss-hh:mm:ss}

RECOVER△MODULE=modulename [,IYPE={module-type}] [FILENAME={filename
'filename'
"filename"}]
[,RDPASS=password][,WRPASS=password],VSN=volume

REMARK△text

RESUME

RP△[function-code][,ACCT=acctno][,CART=cartridge-name][,FILE=filename]
[,FORM=formname][,JOB=jobname]

RUN {([did],label)}△ [jobname] [(new-name)] [:alt-filename
(RDR,label)] ((alt-filename, (RES
RUN
vsn))
: (alt-filename, [(RES),read-pass
:RUN
vsn]))

{PRE
HIGH
NO}

[,key-1=val-1,...,key-n=val-n]

RV△ jobname [(new-name)]

| | | | |
|---|---|---|----------------------------|
| { | :alt-filename | } | [, {PRE HIGH LOW}] |
| | :(alt-filename, {RES RUN vsn}) | | |
| { | :(alt-filename, {RES RUN vsn}, read-pass) | } | |
| | | | |

[, key-1=val-1, ..., key-n=val-n]

SI{ ([did], label) }△ [jobname] [(new-name)]

| | | | |
|---|---|---|----------------------------|
| { | :alt-filename | } | [, {PRE HIGH LOW}] |
| | :(alt-filename, {RES RUN vsn}) | | |
| { | :(alt-filename, {RES RUN vsn}, read-pass) | } | |
| | | | |

SC△ jobname [(new-name)]

| | | | |
|---|---|---|----------------------------|
| { | :alt-filename | } | [, {PRE HIGH LOW}] |
| | :(alt-filename, {RES RUN vsn}) | | |
| { | :(alt-filename, {RES RUN vsn}, read-pass) | } | |
| | | | |

SCREEN△ [, {SCROLL
ROLL
NP
WRAP
NOROLL}] [, {UPPER
LOWER}] [, XMIT={VAR
CHAN
ALL}] [, XFER={VAR
CHAN
ALL}] [, SPEED={2400
4800
2400
1200
600
300}]

[, SPACEBAR={DESTRUCT
NONDESTRUCT}] [, LINES={24
12}] [, KEYBOARD={STANDARD
KATAKANA}] [, INTENSITY={NORMAL
LOW
REVERSE}]

[, LOG={ALL
COMMANDS}] [, {WKSTN}] [, {BURST}] [, {CONTINUOUS}]

STATUS△ [{TERMINALS
RESOURCES
JOBS
FUNCTIONS
VOLUMES}]

| |
|--|
| <u>S</u> IOPΔjobname |
| <u>I</u> ELLΔ { {user-id}, } 'text' { ALL } |
| <u>U</u> LDΔ, filename, vsn, SIZE=nΔ { PRINT { Δ SCRATCH } { NOPRINT { } SAVE } } |
| <u>V</u> IOCΔ['file-prefix',]VSN=volumeΔ[FREE] |





Appendix C. Sample Workstation Sessions

C.1. GENERAL

This appendix contains sample workstation sessions. These "sessions" consist of the interactive command you would use to accomplish different system jobs. Each session is complete with all commands necessary, beginning with the LOGON command and ending with the LOGOFF command. They are included to give you a better understanding of how the various interactive commands work together. Two of the sample sessions perform utility tasks, or "system housekeeping" chores. Another runs a user job, while the last initializes the general editor and uses it to make corrections to some data in a file.

In these sample sessions, messages and other output from the system are shown indented. However, not all messages normally displayed during the course of a workstation session are shown, such as the LOGON ACCEPTED message and system bulletin. Shown are only those messages directly affecting the progress of the session and any output generated by commands.

C.2. A SESSION TO MAKE PUNCHED CARD COPIES OF A MODULE

```

1. LOGON USERA1,DEV4,SECRET,BULLETIN=NO
2. VTOC VSN=PAY001
3.     FILENAME  TYPE  EXT  CYL  %DIR  %DATA  %THIRD  AVAIL
4.     $VTOC      MIRAM 001  001
5.     FICACOMP   SAT   001  002   55    90      0     50
6.     CHECKWRIT SAT   001  003   40    87      0    100
7.     PAYREP     SAT   001  003   60    90      0    150
8.     TOTAL FREE CYLINDERS 392   TRACKS 000
9.     FST FILENAME=CHECKWRIT,VSN=PAY001
10.     S - PLANTCK      S - OFFCK      S - DIVCK
11.     S - RETIRCK     S - REDEVCK   S - VACCK
12.     TEL $Y$CON,'PUT CARDS IN PUNCH PUNCHING 2 DECKS NOW'
13.     PUN MODULE=RETIRCK,TYPE=S,FILENAME=CHECKWRIT,VSN=PAY001,COPIES=2
14.     IS90 PUNCH COMMAND TERMINATED NORMALLY
15.     LOGOFF

```

In line 1, we log on to the system with the user-id USER1A, the account number DEV4, and the password SECRET. We do not specify an execution profile. By specifying BULLETINNO, we have caused the system bulletin not to be displayed when the LOGON command is accepted.

Line 2 shows a VTOC command requesting display of the volume table of contents (VTOC) of volume PAY001. Lines 3 through 7 show the VTOC display for volume PAY001. Of the files listed, we are interested in CHECKWRIT. Line 9 shows an FSTATUS command to list the modules present in the file CHECKWRIT. Notice that we abbreviated FSTATUS. The command executed satisfactorily because we entered a sufficient portion of the command to enable the system to distinguish it. Lines 10 and 11 show the FSTATUS display of the modules in CHECKWRIT. Now, we want to make two punched card copies of RETIRCK. Line 12 shows a TELL command informing the system operator to place cards in the card punch so that the copies may be made. Line 13 shows the PUNCH command, which will make our copies. Line 14 shows the message that is displayed when the PUNCH command has completed the two copies. You then log off the system with the LOGOFF command and pick up the punched cards at the computer site.

C.3. A SESSION TO RUN A USER JOB

```
1. LOGON USER34,4567,BULLETIN=NO
2. TEL 'MOUNT D00026'
3. RV COMPILE:(USERLIB,D00026)
4. DI JS,COMPILE
5.     COMPILE NOT YET SCHEDULED - INSUFFICIENT MAIN STORAGE
6. DE JBQ,COMPILE
7. LOGOFF
```

In line 1, we log on to the system with the user-id USER34 and the account number 4567. No password or execution profile is entered, but we do specify that the system bulletin is not to be displayed. In line 2, we tell the system operator to mount disk volume D00026. We do this because in line 3, we enter an RV command to run a job named COMPILE, residing in a library file named USERLIB, on volume D00026. After a while, we decide to check on the progress of COMPILE and enter a DISPLAY JS command to obtain the status of compile. Line 5 shows the message produced by the DI JS command, showing that COMPILE has not been scheduled, due to insufficient main storage. Being pressed for time, we decide not to wait any longer to execute COMPILE and, in line 6, enter a DELETE JBQ command to remove COMPILE from the scheduling queue. COMPILE will not execute. In line 7 we log off the system.

C.4. A SESSION TO USE THE GENERAL EDITOR

```
1. LOGON BILLM
2. EDT
3. 1.0000 @READ FILENAME=NEWMEMBERS,RDPASS=RDR,VSN=D00029
4. 1.0000 @PRINT 1:3
5. 1.0000 DAVID R. MAST
6. 2.0000 ANN SMITH
7. 3.0000 T. AARON WALKER
8. 4.0000 @CHANGE 'T.' TO 'TIMOTHY'
9. 4.0000 @ON 2 INSERT 'MARY'
10. 4.0000 @DELETE 1
11. 4.0000 @PRINT 1:3
12. 2.0000 MARYANN SMITH
13. 3.0000 TIMOTHY AARON WALKER
14. 4.0000 @WRITE FILENAME=NEWMEMBERS,WRPASS=LKT,VSN=D00029,INIT=YES
15. 4.0000 @HALT
16. LOGOFF
```

In line 1, we log on with the user-id BILLM, and no account number, password, or execution profile. Since we did not specify that we do not want to see the system bulletin, it will be displayed. In line 2, we initialize the general editor through the EDT command. Line 3 causes a copy of the file NEWMEMBERS to be read into the workspace file. In line 4, we request that lines 1 through 3 of the workspace file be displayed. Lines 5, 6, and 7 display lines 1 through 3 of the workspace file. In line 8, we begin to use the editor to make changes to the file, in this line, changing the T of T. AARON WALKER to TIMOTHY. Line 9 causes MARY to be added to ANN in line 2 of the workspace file, and line 10 deletes line 1 of the workspace file. In line 11, we are displaying the corrections we have made, checking them to make sure we did them correctly. Lines 12 and 13 show the corrected lines. In line 14, we are writing a copy of the workspace file back to the original file. The INIT=YES parameter causes the workspace copy to overwrite the original file, causing the changes we made to be included. Line 15 terminates the general editor and, in line 16, we log off the system.

C.5. A SESSION TO ERASE OBSOLETE FILES

```
1. LOGON USER10,ED06,36RACD
2. STATUS VOLUMES
3.     D-REL070    D-ACTPAY    D-WORK01
4. TEL 'MOUNT DISK VOLUME PAYWK1'
5.     -CONS-: 'VOLUME PAYWK1 MOUNTED'
6. VTOC VSN=PAYWK1
7.     FILENAME  TYPE  EXT  CYL  %DIR  %DATA  %THIRD  AVAIL.
8.     PAYFIC31880  MIRAM 002  007   50   87       0    90
9.     PAYBND31880  MIRAM 001  005   35   75       0    60
10.    PAYCHK31880  MIRAM 001  006   47   90       0    15
11.     TOTAL FREE CYLINDERS 382 TRACKS 000
12. ER FILENAME=PAYFIC31880,VSN=PAYWK1
13. IS51 ERASING ENTIRE FILE,PROCEED? (Y,N)
14.     Y
15. IS90 ERASE COMMAND TERMINATED NORMALLY
16. LOGOFF
```

Line 1 logs us on with the user-id USER10, the account number ED06, and the password 36RACD. We want to find out what volumes are currently mounted on the system, so in line 2 we enter the STATUS command with the VOLUMES parameter. Line 3 shows the display produced by the STATUS command. Since the volume we want to work with is not currently mounted, in line 4, we send a message to the system operator, directing him to mount the volume we want. In line 5, we receive the reply that the volume is mounted. In line 6, we perform a VTOC command on the volume just mounted, and discover, in lines 7 through 11, that there are three files on the volume. Notice that each file name has a 5-digit number, this number is a calendar date, indicating when the file was created. We decide that the file PAYFIC31880 is obsolete, and we want to erase it. We enter the ERASE command on line 12. When you enter the ERASE command to erase an entire file, you receive the message shown on line 13. This helps prevent accidental erasure of files. We enter Y on line 14 and the command is executed. Line 15 shows the message you receive upon successful completion of the command. In line 16, we log off.

Appendix D. Positional Format for File Parameters

Certain file parameters may be entered in a *positional* format, instead of using keywords. In a positional format, only the actual parameters are entered; keywords are not used to identify them.

Parameters are identified to the system by their *position* in the parameter string. The positional parameters are separated from each other by commas, and from the command with which they are associated by a space. Four parameters may be entered in the positional format; module name, file name with read and write passwords, volume serial number, and module type. Table D-1 equates the positional parameters to their keyword equivalents:

Table D-1. Positional and Keyword Parameters

| Positional | Keyword |
|--------------------------------------|---|
| module | MODULE=modulename |
| filename 'filename' "filename" | FILENAME=filename 'filename' "filename" |
| readpass/writepass | RDPASS=password WRPASS=password |
| volume | VSN=volume |
| type | TYPE=moduletype |

The general format for positional file parameters is as follows:

$$[\text{modulename}], \left[\left(\begin{array}{l} \text{filename} \\ \text{'filename'} \\ \text{"filename"} \end{array} \right) [([\text{readpass}][/\text{writepass}])] , [\text{volume}], [\text{type}] \right]$$

If, in entering a command, you use *both* positional and keyword parameters, you must enter the positional parameters *first*, with the keyword parameters following the *last* positional parameter.

The positional parameters *must* be entered in the order shown, or the command will be rejected. If a positional parameter is optional in a command and you do not use it, you must enter a comma in its place in the parameter string. If you do not enter a module name but do enter a file name and volume serial number, you must enclose the file name in quotation marks ("...") or enter a leading comma (comma *before* the file name). The read and write passwords must be entered in parentheses. If both are entered, the read password must be entered first, followed by a slash (/) and then the write password. If both are entered, they must both be enclosed in one set of parentheses. No command is entered between the file name and passwords.

The following is an example of a positional parameter string:

```
PAYJOB1,PAYFILE(SECRET),ABC466,S
```

This example shows a source module (S) named PAYJOB1, residing in a file named PAYFILE, located on a volume with the volume serial number ABC466. The read password for PAYFILE is SECRET.

The following is an example of a positional parameter string in which no module name is entered.

```
"PAYROLL",ABC466
```

In this example, only the file name, PAYROLL, and the volume serial number, ABC466, are given. Therefore, the file name is enclosed in quotation marks. Note also that no password is given.

The previous example could also be written as follows:

```
,PAYROLL,ABC466
```

Appendix E. Function Key Summary

Table E-1 provides a summary of the function keys used by interactive services and the various interactive facilities.

Table E-1. Function Key Summary (Part 1 of 2)

| Software Component | Key | Function |
|----------------------|--|--|
| Interactive services | F15 | Informs the system you have no more data to input from the workstation (end of file) |
| | F17 | Temporarily halts the workstation display (See F19.) |
| | F19 | Restarts workstation display after it has been temporarily stopped using F17 or when the screen is full of data |
| General editor | F1 | Suppresses any printing options associated with a command (Same as F18) |
| | F2 | Terminates processing of a command |
| | F3 | Displays a screen showing the parameters on the @SET directive, or those that make up your EDT environment. F3 is the same as issuing the PARAMS screen command. |
| | F4 | Displays a free-form screen through which you can switch to block mode for entering multiple commands or data. F4 is the same as issuing the BLOCK screen command. |
| | F5 | Displays free-form screens, showing the EDT work-space file, where you can update lines or simply view them. F5 is the same as issuing the ROLL screen command. |
| | F6 | Displays help screens for any EDT error messages. F6 is the same as issuing the HELP screen command. |
| | F12 | Shows a previously displayed additional help screen for a specific command when that command requires several help screens to fully describe it |
| | F13 | Displays the EDT command menu screen and help screens for any of the EDT commands (that is, EDT commands, modifiers, directives, procedure file commands, variables, and screen commands). F13 is the same as issuing the PROMPT screen command. From a help screen, F13 also lets you see subsequent help screens needed to fully describe the command. |
| | F14 | Returns you to the point in your EDT session where you originally entered a screen command. F14 is the same as issuing the RESTORE screen command. |
| F15 | Recognized as EOF indicator. Halts an EDT session run in batch mode or produces an error message in interactive mode | |

Table E-1. Function Key Summary (Part 2 of 2)

| Key | Function | |
|------------------------|----------|---|
| General editor (cont) | F18 | Suppresses printing option associated with a command. (Same as F1) |
| | F19 | Restarts workstation display after it has been temporarily stopped using F17 or when the screen is full of data |
| Screen format services | F1 | Returns the screen format generator to the HOME screen. Returning home deletes all work done up to that point. |
| | F5 | Breakpoints the spool file to a printer. Records currently in the spooled printer output are directed to a printer (pending an available device). Records added to the spool file after the breakpoint are held until a subsequent breakpoint is requested or until the end of the job. |
| | F13 | Displays a HELP screen appropriate to the step you're at when generating a screen format (See F14.) |
| | F14 | Removes the HELP screen displayed by using F13; returns the screen format generator to the point it was at before display of the HELP screen |
| | F15 | Indicates end of input data |
| | F16 | Indicates input data cannot be entered properly |
| | F20 | Restores the screen to its original contents for the current pass if it has inadvertently been destroyed |
| BASIC | F1 | <p>Pauses or terminates execution of a BASIC program. When no I/O operation is in progress, the system displays the message:</p> <p style="text-align: center;">EXECUTION PAUSED AT LINE xxxx CONTINUE (Y,N)?</p> <p>Key in Y to continue (resume) execution. Key in N to terminate the program.</p> <ul style="list-style-type: none"> ■ If a BASIC program is requesting output, you must press XMIT after pressing F1 to display the above message. ■ If the BASIC program is sending output, the workstation screen must be full of data for F1 to be recognized. Press F19 to display the above message. |
| | F19 | See F1. |
| ESCORT | F1 | Signals the end of input and returns to master menu. (Used only with the structure processor) |
| | F2 | Cancels display output. (Used only with the structure processor) |
| | F3 | Cancels the current screen and returns to the previous screen. (Used only for program and tutorial modes) |
| | F4 | <ul style="list-style-type: none"> ■ Structure processor: aborts structure and returns to menu ■ Program mode: ends free-form input and returns to previous menu ■ Run-time processor: terminates program and returns to caller |

Index

| Term | Reference | Page | Term | Reference | Page |
|--|-----------|------|--|-----------|------|
| A | | | C | | |
| Accounting, interactive | 2.8.1 | 2-14 | CANCEL command | 4.2.2 | 4-3 |
| Adding, modifying library module comments (COMMENT) | 4.4.4 | 4-47 | CHANGE command | 3.10.5 | 3-33 |
| ALLOCATE command | 4.4.3 | 4-44 | Changing job scheduling | | |
| Altering workstation display screen characteristics (SCREEN) | 4.2.8 | 4-10 | deferring jobs (HOLD) | 3.10.2 | 3-30 |
| ASK command | 4.2.1 | 4-2 | deleting jobs, job queue (DELETE) | 3.10.4 | 3-32 |
| | | | description | 3.10 | 3-28 |
| | | | displaying contents, job queue (DISPLAY JBQ) | 3.10.3 | 3-31 |
| | | | queues | 3.10.5 | 3-33 |
| | | | rescheduling jobs (BEGIN) | 3.10.1 | 3-28 |
| | | | COBEDT | 5.5.3 | 5-25 |
| | | | COBOL editor | 5.5.3 | 5-25 |
| | | | COMMENT command | 4.4.4 | 4-47 |
| | | | Connecting workstation to job | | |
| | | | automatic connect | 3.11 | 3-34 |
| | | | CONNECT command | 3.11.1 | 3-34 |
| | | | connecting to use WORKSTATION mode | 3.11.1 | 3-34 |
| | | | description | 3.11 | 3-34 |
| | | | COPY command | 4.4.5 | 4-49 |
| B | | | | | |
| BASIC | 5.7 | 5-30 | | | |
| BEGIN command | | | | | |
| description | 3.10.1 | 3-28 | | | |
| JBQ | 3.10.1 | 3-28 | | | |
| jobname | 3.10.1 | 3-28 | | | |
| BEGIN SPL command | 4.3.8 | 4-32 | | | |
| BRKPT command | 4.3.10 | 4-34 | | | |
| BRKPT LOG command | 4.5.1 | 4-98 | | | |

| Term | Reference | Page | Term | Reference | Page |
|---|-----------|-------|---|------------|-------|
| D | | | E | | |
| Data output, resuming UNISCOPE 100 and 200 terminals | A.3.1 | A-10 | EFP | 5.5.1 | 5-20a |
| UTS 400 terminal | A.4.1 | A-15 | ENTER command | 4.4.9 | 4-60 |
| DDP | 5.6 | 5-27 | ERASE command | 4.4.10 | 4-63 |
| Deferring jobs (HOLD) | 3.10.2 | 3-30 | Error file processor | 5.5.1 | 5-20a |
| DEFKEY command | 4.4.6 | 4-56 | ESCORT | 5.8 | 5-30 |
| DEFKEY (delete) command | 4.4.7 | 4-58 | EXECUTE, command and facility | 3.7 | 3-16 |
| DEFKEY DISPLAY command | 4.4.8 | 4-59 | EXPANDED TYPEWRITER keyboard | A.2 | A-1 |
| DELETE command | 3.10.4 | 3-32 | | | |
| DELETE SPL command | 4.3.9 | 4-33 | F | | |
| Deleting library files, modules, and MIRAM files (ERASE) | 4.4.10 | 4-63 | FILE command | 3.3 | 3-4 |
| Deleting spool files (DELETE SPL) | 4.3.9 | 4-33 | File parameters, positional parameters | Appendix D | |
| Disconnecting workstation (FREE) | 4.2.4 | 4-6 | Filing job control streams (FI) | 3.3 | 3-4 |
| DISPLAY ACT command See also DISPLAY SPL command. | 4.3.6 | 4-25 | FREE command | 4.2.4 | 4-6 |
| DISPLAY JS command | 4.2.3 | 4-4 | FSTATUS command | 4.4.11 | 4-65 |
| DISPLAY LOG command | 4.5.2 | 4-101 | Function key | | |
| DISPLAY SPL command See also DISPLAY ACT command. | 4.3.6 | 4-25 | BASIC | Table E-1 | E-2 |
| Displaying contents, job queue (DISPLAY JBQ) | 3.10.3 | 3-31 | console/workstation | A.3 | A-7 |
| Distributed data processing | 5.6 | 5-27 | ESCORT | Table E-1 | E-2 |
| DLOAD command | 3.8 | 3-25 | general editor | Table E-1 | E-1 |
| Downline loading programs | 3.8 | 3-25 | interactive services | Table E-1 | E-1 |
| | | | screen format services | Table E-1 | E-1 |
| | | | usage | Table E-1 | E-1 |
| | | | UTS 400 terminals | A.4 | A-11 |
| | | | workstations | A.2 | A-1 |

| Term | Reference | Page | Term | Reference | Page |
|---|-----------|------|---------------------------------------|-----------|------|
| G | | | J | | |
| General editor | | | Job control dialog | | |
| commands | Table 5-1 | 5-13 | description | 3.2 | 3-2 |
| description | 5.5 | 5-12 | HELP | 3.2 | 3-2 |
| initializing | 5.5 | 5-12 | SC JC\$BLD, command to initiate | 3.2 | 3-2 |
| RPG editor | 5.5.2 | 5-23 | | | |
| GO command | 4.2.6 | 4-8 | | | |
| H | | | L | | |
| HELP | | | LIBS PAC | 4.4.17 | 4-85 |
| command | 4.4.12 | 4-68 | Listing the contents of a VTOC (VTOC) | 4.4.20 | 4-94 |
| interactive data utilities | 5.3.3 | 5-10 | Log, workstation | 2.3.2 | 2-5 |
| job control dialog | 3.2 | 3-2 | Log files | 4.5 | 4-98 |
| Holding spooled files (HOLD SPL) | 4.3.7 | 4-31 | LOG parameter | 2.3.2 | 2-5 |
| HU command | 5.4 | 5-11 | | 4.2.8 | 4-12 |
| I | | | LOGOFF command | | |
| IBM System 3 control streams | 3.6 | 3-14 | console/workstation | A.3.3 | A-11 |
| Indicator line | A.2.1 | A-6 | description | 2.8 | 2-13 |
| Inserting comment, command stream (REMARK) | 4.4.18 | 4-88 | UNISCOPE 100 and 200 terminals | A.3.3 | A-11 |
| Integrated communications access method (ICAM) | | | UTS 400 terminals | A.4.3 | A-15 |
| connecting local workstations | 2.7 | 2-13 | workstation | A.2.2 | A-7 |
| standard terminal dialog | A.6.1 | A-18 | LOGON command | | |
| terminal information | A.6 | A-18 | console/workstation | A.2.1 | A-6 |
| Interactive accounting | 2.8.1 | 2-14 | description | 2.3.2 | 2-4 |
| Interactive data utilities | | | ENTER stream | 4.4.6 | 4-60 |
| description | 5.3 | 5-8 | LOGON ACCEPTED display | 2.3.3 | 2-6 |
| HELP | 5.3.3 | 5-10 | menu | 2.3.1 | 2-3 |
| initializing | 5.3.1 | 5-9 | UNISCOPE 100 and 200 terminals | A.3.2 | A-10 |
| using | 5.3.2 | 5-9 | UTS 400 terminals | A.4.2 | A-15 |
| Interactive dump restore hardware utility | 5.4 | 5-11 | workstation | A.2.1 | A-6 |

| Term | Reference | Page |
|---|-----------|-------|
| M | | |
| Making printed copies, files (PRINT) | 4.4.14 | 4-72 |
| Making punched card copies, file (PUNCH) | 4.4.15 | 4-78 |
| Master workstation | | |
| changing master workstation, job description | 3.12.1 | 3-36 |
| MAS=user-ID job control statement | 3.12.2 | 3-36 |
| ORI=user-ID job control statement | 3.12.2 | 3-36 |
| MAS=user-ID control statement | 3.12.2 | 3-36 |
| Menu generator | 5.2.1 | 5-6 |
| Menu processor | 5.2.2 | 5-7 |
| Menu services | 5.2 | 5-6 |
| Menus | | |
| calling system | 4.4.13 | 4-70a |
| | 2.4 | 2-7 |
| MESSAGE WAITING, UNISCOPE 100 and 200 terminals | A.3.1 | A-10 |
| Mode change keys | | |
| FUNCTION | Table A-1 | A-4 |
| SYS MODE | Table A-1 | A-4 |
| WS MODE | Table A-1 | A-4 |
| Modes | | |
| console/workstation | Table A-1 | A-4 |
| general description | 2.2.3 | 2-2 |
| SYSTEM | 2.2.2 | 2-2 |
| UNISCOPE 100 and 200 terminals | Table A-2 | A-8 |
| UTS 400 terminals | Table A-3 | A-12 |
| WORKSTATION | Table A-1 | A-4 |
| MSG WAIT key, UTS 400 terminal | Table A-3 | A-12 |

| Term | Reference | Page |
|--|-----------|------------|
| O | | |
| OCL/OV commands | 3.6 | 3-14 |
| OPTION LOG job control statement | 4.3.5 | 4-22 |
| OPTION OUT job control statement | 4.3.5 | 4-22 |
| ORI=user-ID job control statement | 3.12.2 | 3-37 |
| Output writer | | |
| function codes | 4.3.3.1 | 4-17 |
| general | 4.3.3 | 4-17 |
| manually loading | 4.3.4 | 4-20 |
| manually loading to an auxiliary printer | 4.3.5 | 4-22 |
| P | | |
| PAUSE command | 4.2.5 | 4-7 |
| Positional format for file parameters | | Appendix D |
| Prerun job execution | | |
| description | 3.5 | 3-10 |
| running expanded job control streams | 3.5.1 | 3-10 |
| SI/SC commands | 3.5.1 | 3-10 |
| PRINT command | 4.4.14 | 4-72 |
| PR/PU command | 4.3.4 | 4-20 |
| Printing files | | |
| editor print command | Table 5-1 | 5-13 |
| PRINT command | 4.4.14 | 4-72 |
| PUNCH command | 4.4.15 | 4-78 |
| Punching files | | |
| editor punch command | Table 5-1 | 5-13 |
| PUNCH command | 4.4.15 | 4-78 |

| Term | Reference | Page | Term | Reference | Page |
|--|-----------|------|--|-----------|------|
| R | | | S | | |
| Reactivating suspended jobs (GO) | 4.2.6 | 4-8 | SCREEN command | 4.2.8 | 4-10 |
| REBUILD command | 2.6.4 | 2-12 | Screen format coordinator description | 5.1.2 | 5-5 |
| RECALL command | 4.4.16 | 4-83 | USE job control statement | 5.2.2 | 5-7 |
| RECOVER command | 4.4.17 | 4-85 | Screen format generator description | 5.1.1 | 5-2 |
| Releasing held spooled files (BEGIN SPL) | 4.3.8 | 4-32 | HELP function | 5.1.1 | 5-5 |
| REMARK command | 4.4.18 | 4-88 | 5.3.3 | 5-10 | |
| Rescheduling jobs (BEGIN) | 3.10.1 | 3-28 | HOME screen | Table A-1 | A-5 |
| Restoring response message (REBUILD) | 2.6.4 | 2-12 | Screen format services coordinator | 5.1.2 | 5-5 |
| RESUME command | 4.2.7 | 4-9 | description | 5.1 | 5-1 |
| Resuming execution, general editor (RESUME) | 4.2.7 | 4-9 | generator | 5.1.1 | 5-2 |
| ROUTE job control statement | 4.3.5 | 4-23 | Sending messages, system operator (TELL) | 4.2.10 | 4-14 |
| RP command | 4.3.5 | 4-22 | SI/SC commands | 3.5.1 | 3-10 |
| RPG editor | | | Standard terminal dialog description | A.6.1 | A-18 |
| description | 5.5.2 | 5-23 | sign-off command (\$\$SOFF) | A.6.3 | A-20 |
| formatted | 5.5.2 | 5-23 | sign-on command (\$\$SON) | A.6.2 | A-19 |
| free form | 5.5.2 | 5-24 | STATUS command | 4.4.19 | 4-89 |
| initializing | 5.5.2 | 5-23 | STOP command | 4.2.9 | 4-13 |
| positional | 5.5.2 | 5-23 | Super set job control stream | 3.7 | 3-16 |
| Running jobs, workstation | | | Suspending job processing (PAUSE) | 4.2.5 | 4-7 |
| changing job scheduling | 3.10.5 | 3-33 | Switching | 2.2.3 | 2-2 |
| CONNECT command | 3.11.1 | 3-34 | SYS MODE key | Table A-1 | A-4 |
| connecting workstation, job | 3.11 | 3-34 | SYSTEM mode | | |
| deleting jobs, job queue (DELETE) | 3.10.4 | 3-32 | description | 2.2.2 | 2-2 |
| displaying contents, job queue (DISPLAY JBQ) | 3.10.3 | 3-31 | switching | 2.2.3 | 2-2 |
| filing job control streams (FI) | 3.3 | 3-4 | SYS MODE key | Table A-1 | A-4 |
| general commands | Section 4 | | System-supplied menus | 2.4 | 2-7 |
| job control dialog | 3.2 | 3-2 | System 80 console workstation | | |
| master workstation | 3.12 | 3-36 | description | A.5 | A-15 |
| prerun job execution | 3.5 | 3-10 | function keys | A.5.2 | A-17 |
| rescheduling jobs (BEGIN) | 3.10.1 | 3-28 | keyboard | Fig. A-6 | A-16 |
| running expanded job control streams (SI/SC) | 3.5.1 | 3-10 | LOGOFF procedure | A.5.3 | A-17 |
| running IBM System 3 control streams (OCL/OV) | 3.6 | 3-14 | LOGON procedure | A.5.2 | A-17 |
| running jobs (RUN/RV) | 3.4 | 3-6a | mode switching | A.5.1 | A-15 |
| RUN/RV commands | 3.4 | 3-6a | | | |

| Term | Reference | Page | Term | Reference | Page |
|--|-----------|-------|---------------------------------|-----------|------|
| System 80 workstation function keys | A.2 | A-1 | | V | |
| LOGOFF procedure | A.2.2 | A-7 | | | |
| LOGON procedure | A.2.1 | A-6 | VTOC command | 4.4.20 | 4-94 |
| mode switching | A.2 | A-1 | | | |
| resuming data output | A.2 | A-1 | | | |
| T | | | | | |
| TELL command | 4.2.10 | 4-14 | | | |
| Terminating job (STOP) | 4.2.9 | 4-13 | | W | |
| TRANSMIT key See also XMIT key. | A.5.1 | A-17 | Workstation logging | 4.5 | 4-98 |
| | | | WORKSTATION mode description | 2.2.1 | 2-1 |
| | | | switching | 2.2.3 | 2-2 |
| | | | WS MODE key | A.5.1 | A-17 |
| | | | See also System 80 workstation. | | |
| | | | WS MODE key | A.5.1 | A-17 |
| U | | | | | |
| ULD command | 3.9 | 3-26a | | | |
| UNISCOPE 100 and 200 terminals function keys | Table A-2 | A-9 | | | |
| LOGOFF procedure | A.3.3 | A-11 | | | |
| LOGON procedure | A.3.2 | A-10 | | | |
| resuming data output | A.3.1 | A-10 | | | |
| Upline dumps | 3.9 | 3-26a | | | |
| Using interactive commands, batch environment (ENTER) | 4.4.9 | 4-60 | | X | |
| UTS 400 terminal function keys | Table A-3 | A-13 | XMIT key | | |
| LOGOFF procedure | A.4.3 | A-15 | UTS 400 terminal workstation | Table A-3 | A-12 |
| LOGON procedure | A.4.2 | A-15 | See also TRANSMIT key. | Table A-1 | A-4 |
| resuming data output | A.4.1 | A-15 | | | |

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

(Document Title)

(Document No.)

(Revision No.)

(Update No.)

Comments:

Cut along line.

From:

(Name of User)

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

(Document Title)

(Document No.)

(Revision No.)

(Update No.)

Comments:

Cut along line.

From:

(Name of User)

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

(Document Title)

(Document No.)

(Revision No.)

(Update No.)

Comments:

Cut along this line.

From:

(Name of User)

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424



FOLD

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

(Document Title)

(Document No.)

(Revision No.)

(Update No.)

Comments:

Cut along line.

From:

(Name of User)

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

(Document Title)

(Document No.)

(Revision No.)

(Update No.)

Comments:

Cut along line.

From:

(Name of User)

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD