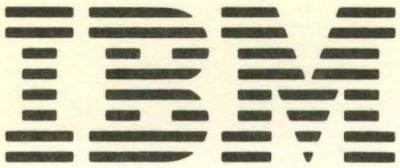


SC21-5155-8

File No. S34-36

IBM System/34
System Support
Reference Manual

Program Number 5726-SS1

SC21-5155-8

File No. S34-36

IBM System/34
System Support
Reference Manual

Program Number 5726-SS1

Ninth Edition (January 1982)

This is a major revision of, and obsoletes, SC21-5155-7. Additions were made to support the 5224 Printer, expanded spool file control, history file scroll, the X.21 Feature, the auto response facility, autocall capability in countries other than the United States and Canada, and documentation enhancements to the COPYPRT procedure. The SESSION OCL statement is now documented in the *IBM System/34 Interactive Communications Reference Manual*, SC21-7751. These and other miscellaneous changes and additions to text and illustrations are indicated by a vertical line to the left of the change or addition.

This edition applies to release 8, modification 0 of the IBM System/34 System Support Program Product (Program Number 5726-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters.

Changes are periodically made to the information herein; these changes will be reported in technical newsletters or in new editions of this publication.

Use this publication only for the purposes stated in the *Preface*.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country. For example, ideographic support is available in only Far East countries.

Publications are not stocked at the address below. Requests for copies of IBM publications and for technical information about the system should be made to your IBM representative or to the branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. Use the Reader's Comment Form at the back of this publication to make comments about this publication. If the form has been removed, address your comments to IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

© Copyright International Business Machines Corporation 1977, 1978, 1979, 1980, 1981, 1982

Preface

This reference manual provides programmers with information needed to establish administrative and operating procedures for an IBM System/34. This manual also provides programmers with the reference information needed to identify and use the OCL statements, SSP procedures, control commands, and SSP utility programs required to perform desired functions.

This manual contains:

- A detailed description of each OCL statement
- A detailed description of each SSP procedure (except special-purpose procedures, such as procedures used for SSP installation and modification, which are described in related publications) and the procedure command statement that causes the procedure to run
- A detailed description of the control commands available to the system operator, the subconsole operator, and the display station operators
- A detailed description of each SSP utility program (except special-purpose programs, such as utility programs used for data communications and for SSP installation and modification, which are described in related publications) and the utility control statements for the utility program
- A detailed description of the statements that you can use and the rules that you must follow if you write your own procedures
- A discussion of programming considerations related to the use of jobs, job steps, screen formats, IPL, and forms alignment

Appendixes to this manual contain:

- A description of the relationship of disk records, blocks, and sectors
- Tables for hexadecimal and decimal conversion
- A description of the diskette data formats for System/34
- A description of the SSP service procedures
- A list of the OCL statements and utility control statements executed for each SSP procedure
- Lists of the characters on the standard 48-, 48HN-, 64-, 64B-, 64C-, 96-, and 188-character print belts
- A list of the BSC polling and addressing characters for System/34 tributary stations
- A list of EBCDIC and ASCII
- A description of the error messages printed by the $SFGR and $LABEL utility programs
- Translation tables for character translation
- A description of the multinational character set conversion programs
- A glossary that defines terms and abbreviations used in the manual

Insert tabs are available for this manual to divide sections. This will help you locate information quickly. Requests for insert tabs should be made to your IBM representative or the IBM branch office serving your locality. The title and order number is: *Insert Tabs for the IBM System/34 System Support Reference Manual,* SX21-7837.

Note: This manual follows the convention that *he* means *he* or *she*.

SYSTEM REQUIREMENTS

For a list of system requirements, see the *IBM System/34 Planning Guide*, GC21-5154.

PREREQUISITE PUBLICATIONS

- *IBM System/34 Introduction*, GC21-5153
- *IBM System/34 Planning Guide*, GC21-5154

RELATED PUBLICATIONS

- *IBM System/34 Operator's Guide*, SC21-5158
- *IBM System/34 Displayed Messages Guide*, SC21-5159
- *IBM System/34 Installation and Modification Reference Manual: Program Products and Physical Setup*, SC21-7689
- *IBM System/34 Data Communications Reference Manual*, SC21-7703
- *IBM System/34 Command and OCL Statements Reference Summary*, GX21-7690
- *IBM System/34 Master Index*, SC21-7739
- *The IBM Diskette General Information Manual*, GA21-9182
- *IBM System/34 Concepts and Design Guide*, SC21-7742
- *IBM System/34 Interactive Communications Feature Reference Manual*, SC21-7751
- *IBM System/34 System Measurement Facility Reference Manual*, SC21-7828
- *IBM System/34 RPG II Reference Manual*, SC21-7667
- *IBM System/34 BASIC Reference Manual*, SC21-7835
- *IBM System/34 COBOL Reference Manual*, SC21-7741
- *IBM System/34 3270 Device Emulation User's Guide*, SC21-7868
- *IBM System/34 FORTRAN Reference Manual*, SC21-7706
- *IBM System/34 Source Entry Utility Reference Manual*, SC21-7657
- *IBM System/34 Basic Assembler and Macro Processor Reference Manual*, SC21-7705
- *IBM System/34 Screen Design Aid Programmer's Guide and Reference Manual*, SC21-7716
- *IBM System/34 Overlay Linkage Editor Reference Manual*, SC21-7707
- *IBM System/34 System Data Areas and Diagnostic Aids Manual*, LY21-0049
- *General Information—Binary Synchronous Communications Systems Reference Library Manual*, GA27-3004
- *IBM Synchronous Data Link Control General Information*, GA27-3093
- *Systems Network Architecture General Information*, GA27-3102
- *IBM System/34 Character Generator Utility User's Guide and Reference Manual*, SC21-7845. This manual is for the ideographic version of the SSP.

IBM publications are available that describe the IBM-supplied ideographic characters and list their corresponding IBM codes. Contact your country representative for further information.

CODING AND DEBUGGING MATERIAL

- *IBM System/34 Display Screen Format Specifications*, GX21-9253
- *IBM 5250 Information Display Station Keyboard Template*, GX21-9266
- *IBM WSU/$SFGR Debugging Template*, GX21-7697
- *IBM 5251 Display Station Keyboard Template Assignment Sheet and Display Screen Layout*, GX21-9271

Contents

HOW TO USE THIS MANUAL . . . . . **ix**
Detailed Information ix
Summary Information ix
Programming Considerations x
Reader's Comments x

CONVENTIONS USED FOR DESCRIBING STATEMENT FORMATS . . . . . **xi**

INTRODUCTION . . . . . **xiii**
OCL Statements xiii
Utility Programs and Utility Control Statements xiii
SSP Procedures and Procedure Commands xiv
Writing Your Own Procedures xiv
Control Commands xiv
Programming Considerations xv
SSP Function Summary xv

CHAPTER 1. OCL STATEMENTS **1-1**
Types of Information Conveyed in OCL Statements . . 1-1
Identifiers 1-1
Parameters 1-2
General OCL Coding Rules 1-2
Continuation 1-3
Comments 1-4
OCL Statement Descriptions 1-6
ATTR Statement 1-6
COMM Statement 1-10
COMPILE Statement 1-12
DATE Statement 1-15
EVOKE Statement 1-17
FILE Statement (for Disk Files) 1-19
FILE Statement (for Diskette Files) 1-29
FORMS Statement 1-35
IMAGE Statement (for IBM 5211 and 3262 Printers Only) 1-37
INCLUDE Statement 1-42
JOBQ Statement 1-45
LIBRARY Statement 1-46
LOAD Statement 1-47
LOCAL Statement 1-48
LOG Statement 1-50
MEMBER Statement 1-51
MENU Statement 1-53
MSG Statement 1-54
OFF Statement 1-55
PAUSE Statement 1-56
PRINTER Statement 1-57
PROMPT Statement 1-62
REGION Statement 1-66
RESERVE Statement 1-68
RUN Statement 1-69
SESSION Statement 1-69
SWITCH Statement 1-70
SYSLIST Statement 1-72
WORKSTN Statement 1-74
* (comment) Statement 1-76
/* (end of data) Statement 1-76
// * (message) Statement 1-77
// ** (message) Statement 1-78

CHAPTER 2. SSP PROCEDURES **2-1**
ALTERBSC Procedure 2-3
ALTERSDL Procedure 2-6
BACKUP Procedure 2-8
BLDFILE Procedure 2-11
BLDLIBR Procedure 2-15
BLDMENU Procedure 2-19
BUILD Procedure 2-34
CATALOG Procedure 2-35
COMPRESS Procedure 2-37
CONDENSE Procedure 2-38
COPYI1 Procedure 2-39
COPYPRT Procedure 2-43
CREATE Procedure 2-52
CRESTART Procedure 2-54
DATE Procedure 2-55
DEFINEID Procedure 2-57
DEFINEPN Procedure 2-59
DEFINX21 Procedure 2-66
DELETE Procedure 2-70
DISABLE Procedure 2-72
DISPLAY Procedure 2-73
ENABLE Procedure 2-75
EXTRACT Procedure 2-77
FORMAT Procedure 2-77
FROMLIBR Procedure 2-81
HELP Procedure 2-86
HISTCRT Procedure 2-93
HISTORY Procedure 2-102
INIT Procedure 2-107
INSTCOPY Procedure 2-110
INSTINIT Procedure 2-110
JOBSTR Procedure 2-110
KEYSORT Procedure 2-114
LIBRLIBR Procedure 2-115
LINES Procedure 2-117
LISTFILE Procedure 2-119
LISTLIBR Procedure 2-124
LOG Procedure 2-127
ORGANIZE Procedure 2-128
OVERRIDE Procedure 2-132
POST Procedure 2-136
PRESTOR Procedure 2-141
PRLIST Procedure 2-142
PRMENU Procedure 2-143
PROF Procedure 2-144
PRSAVE Procedure 2-145
PRSRC Procedure 2-146
PRSRCID Procedure 2-147
REBLD Procedure 2-150
RELOAD Procedure 2-150
REMOVE Procedure 2-153
RENAME Procedure 2-155
REQUESTX Procedure 2-156
RESPONSE Procedure 2-159

RESTORE Procedure 2-160
SAVE Procedure 2-165
SAVELIBR Procedure 2-170
SET Procedure 2-171
SETFILE Procedure 2-174
SETRETRY Procedure 2-175
SPECIFY Procedure 2-176
STARTM Procedure 2-178
STOPM Procedure 2-179
SYSLIST Procedure 2-180
TOLIBR Procedure 2-182
TRANSFER Procedure 2-185
WSUTXCR Procedure 2-192
WSUTXEX Procedure 2-192
WSUTXRV Procedure 2-192
XREST Procedure (For the Ideographic Version of the SSP) 2-193
XSAVE Procedure (For the Ideographic Version of the SSP) 2-195

CHAPTER 3. CONTROL COMMANDS 3-1
DISPLAY STATION OPERATOR CONTROL COMMANDS 3-3
CANCEL Command 3-3
CHANGE Command 3-4
CONSOLE Command 3-6
HOLD Command 3-7
IDELETE Command 3-8
JOBQ Command 3-9
MENU Command 3-10
MODE Command 3-11
MSG Command 3-12
OFF Command 3-14
PRTY Command 3-15
RELEASE Command 3-16
STATUS Command 3-17
TIME Command 3-20
SUBCONSOLE OPERATOR CONTROL COMMANDS . . 3-21
CANCEL Command 3-21
CHANGE Command 3-23
HOLD Command 3-27
MSG Command 3-28
RELEASE Command 3-29
REPLY Command 3-30
RESTART Command 3-31
START Command 3-32
STATUS Command 3-33
STOP Command 3-36
TIME Command 3-37
SYSTEM OPERATOR CONTROL COMMANDS 3-38
ASSIGN Command 3-38
CANCEL Command 3-40
CHANGE Command 3-42
HOLD Command 3-46
IDELETE Command 3-47
MSG Command 3-48
PRTY Command 3-49
RELEASE Command 3-50
REPLY Command 3-52
RESTART Command 3-53
START Command 3-54
STATUS Command 3-57
STOP Command 3-60
TIME Command 3-63
VARY Command 3-64

CHAPTER 4. SSP UTILITY PROGRAMS 4-1
Main Storage Requirements 4-1
Rules for Coding Utility Control Statements 4-2
Utility Program Descriptions 4-4
$ARSP–Auto Response Utility Program 4-4
$BACK–Backup Library Utility Program 4-9
$BICR–Basic Data Exchange Utility Program 4-10
$BMENU–Build Menu Utility Program 4-13
$BUILD–Alternative Sector Rebuild Utility Program 4-18
$COPY–Disk Copy/Display Utility Program 4-20
$DDST–Key Sort Utility Program 4-37
$DELET–File Delete Utility Program 4-38
$DUPRD–Diskette Copy Utility Program 4-48
$FBLD–File Build Utility Program 4-52
$FREE–Disk Reorganization Utility Program 4-55
$HELP–Help Utility Program 4-57
$HIST–History File Display Utility Program 4-57
$HSML–History File Scroll Utility Program 4-62
$IDSET–Switched Line Remote ID Specification Utility Program 4-70
$IEDS–Subsystem Termination Utility Program . . . 4-72
$IENBL–Subsystem Initialization Utility Program . . 4-73
$INIT–Diskette Labeling and Initialization Utility Program 4-75
$LABEL–VTOC Display Utility Program 4-81
Sample Disk VTOC Displays 4-82
Sample Diskette VTOC Display 4-87
$LOADI–Reload Library Utility Program 4-93
$MAINT–Library Maintenance Utility Program . . . 4-94
$MGBLD–Create Message Member Utility Program 4-119
$MMSP–Stop Monitoring Line Utility Program . . 4-125
$MMST–Start Monitoring Line Utility Program . . 4-126
$PACK–Disk Reorganization Utility Program . . . 4-127
$PDSR–Primary SDLC Retry Count Reset Utility Program 4-128
$PNLM–Phone List Utility 4-130
$POST–Data Exchange Utility Program 4-133
$PRES–Resource Security Utility Program 4-136
$PRLT–Security Report Utility 4-136
$PRMN–Menu Security Utility Program 4-137
$PROF–Password Security Utility Program 4-138
$PRON–Resource Owner Utility Program 4-139
$PRST–Security File Restore Utility Program . . . 4-140
$PRSV–Security File Save Utility Program 4-141
$RENAM–File Rename Utility Program 4-142
$RSTRT–Program Restart Utility Program 4-143
$SETCF–Set Configuration Utility Program 4-144
$SFGR–Screen Format Generator Utility Program . 4-157
$SLFL–SETFILE Utility Program 4-192
$UASC–CRT Display of Spool File Entries 4-193
$UASF–User Access to Spool File Utility Program 4-194
$XNLM–X.21 Phone List Utility 4-197
$XREST–Extended Character File Restore Utility (for the Ideographic Version of the SSP) 4-201
$XSAVE–Extended Character File Save Utility (for the Ideographic Version of the SSP) 4-202

CHAPTER 5. WRITING AND USING PROCEDURES . . 5-1
Creating a Procedure 5-1
Calling a Procedure 5-2
Keyboard Entry of the INCLUDE Statement 5-2
Calling a Procedure by Selecting an Item from a Menu 5-2
Calling a Procedure from Another Procedure 5-3
Executing a Procedure 5-5
Procedure Parameters 5-6
Procedure Attributes 5-7
Substitution Expressions 5-8
Substitution Expression Formats 5-8
Nested Substitution Expressions 5-21
Conditional Expressions (IF and ELSE) 5-23
IF Expressions 5-23
ELSE Expressions 5-38
RESET Statement 5-40
CANCEL Statement 5-40
RETURN Statement 5-41
GOTO and TAG Statements 5-41
Sample Procedure 5-43

CHAPTER 6. PROGRAMMING CONSIDERATIONS . . 6-1
System/34 Concepts 6-1
Specifying Region Size 6-2
Specifying Region Size for a Job 6-2
Specifying Region Size for a Job Step 6-2
Libraries 6-3
Library Members 6-3
Storing Library Members in Disk or Diskette Files . . . 6-4
Record-Mode Files 6-4
Sector-Mode Files 6-6
Using Display Screen Formats 6-7
Steps in Creating a Display Screen Format 6-7
Computing Self-Check Digits 6-13
Multiple Formats 6-15
Read Under Format 6-17
Initial Program Load (IPL) 6-18
Forms Alignment when Changing Lines Per Page . . . 6-20

APPENDIX A. RECORDS, BLOCKS, AND SECTOR CONVERSION A-1
Records to Blocks Conversion for Disk–Records Given on File Statement A-1
Determining the Number of Blocks in a Sequential or Direct File A-1
Determining the Number of Blocks in an Indexed File A-1
Disk Sector Address to Block Address Conversion . . . A-2
Disk Block Address to First Sector in Block Conversion A-2

APPENDIX B. HEXADECIMAL AND DECIMAL CONVERSION B-1
Hexadecimal-to-Decimal Example B-2
Decimal-to-Hexadecimal Example B-2

APPENDIX C. DISKETTE FORMATS AND DISKETTE DATA FILES C-1
Diskette Types C-1
Diskette Drives C-1
Diskette Formats C-2
Formats for Diskette 1 Diskette C-2
Formats for Diskette 2D Diskette C-2
Diskette Record Attributes C-2
Records Unblocked and Unspanned C-2
Diskette Data File Types C-3
Basic Data Exchange Files C-3
I Exchange Files C-3
System Files C-4

APPENDIX D. SSP SERVICE PROCEDURES AND COMMANDS D-1
APAR Procedure D-3
APPLYPTF Procedure D-5
DFA Procedure D-7
DUMP Procedure D-8
ERAP Procedure D-11
PATCH Procedure D-12
SEC Procedure D-13
SETDUMP Control Command D-14
STATEST Procedure D-15
TRACE Procedure D-16

APPENDIX E. SSP PROCEDURE CONTENTS E-1
ALTERBSC E-2
ALTERSDL E-2
BACKUP E-3
BLDFILE E-3
BLDLIBR E-4
BLDMENU E-5
BUILD E-6
CATALOG E-7
COMPRESS E-7
CONDENSE E-7
COPYI1 . . . . . E-8
COPYPRT E-8
CREATE E-9
CRESTART E-9
DATE E-9
DEFINEID E-9
DEFINEPN E-10
DEFINX21 . . . . . E-10
DELETE E-11
DISABLE E-11
DISPLAY E-12
ENABLE E-12
FORMAT E-13
FROMLIBR E-14
HELP E-15
HISTCRT E-15
HISTORY E-15
INIT E-16
JOBSTR E-16
KEYSORT E-17
LIBRLIBR E-17
LINES E-17
LISTFILE E-18
LISTLIBR E-19

LOG E-20
ORGANIZE E-20
OVERRIDE E-21
POST E-21
PRESTOR E-23
PRLIST E-23
PRMENU E-23
PROF E-23
PRSAVE E-24
PRSRC E-24
PRSRCID E-24
RELOAD E-24
REMOVE E-25
RENAME E-25
REQUESTX E-25
RESPONSE E-25
RESTORE E-26
SAVE E-27
SAVELIBR E-28
SET E-28
SETFILE E-28
SETRETRY E-29
SPECIFY E-29
STARTM E-29
STOPM E-29
SYSLIST E-30
TOLIBR E-30
TRANSFER E-31
XREST E-33
XSAVE E-33

APPENDIX F. SYSTEM/34 CHARACTERS F-1

APPENDIX G. POLLING AND ADDRESSING CHARACTERS FOR SYSTEM/34 BSC TRIBUTARY STATIONS G-1
EBCDIC G-1
ASCII G-2

APPENDIX H. EBCDIC AND ASCII CODES H-1
EBCDIC H-1
ASCII H-2

APPENDIX J. PRINTED MESSAGES J-1

APPENDIX K. SUMMARY OF DISPLAY SCREEN FORMAT SPECIFICATIONS K-1

APPENDIX L. SYSTEM/34 TRANSLATION TABLES L-1
192- To 96-Character Set Fold (#188E96) L-1
192- To 64-Character Set Fold (#188E64) L-3
192- To 48-Character Set Fold (#188E48) L-5
96- To 64-Character Set Fold (#96E64) L-7
96- To 48-Character Set Fold (#96E48) L-9

APPENDIX M. MULTINATIONAL CHARACTER SET CONVERSION UTILITY PROGRAMS M-1
Functions M-1
5250 Hardware Support M-2
Initiating the Conversion Utility M-2
Library Member Conversion M-3
Library Member Modification M-7
Conversion Tables M-10
Source Statement Length M-11
MCSCU Audit List M-11
Library Statement List M-12
Library Statement Count M-13
Library Statement Modify M-13
RPG M-13
Procedures M-15
Message Members M-15
Menu Members M-15
Screen Formats M-16
Sort Members M-16
Work Station Utility M-16
Other Statement Type M-16
Library Member Copy Back M-17
Additional Library Member Considerations M-17
Data File Conversion M-18
Data File List M-22
Data File Record Count M-23
Data File Modify M-23
Batch Interface M-23
Library Members M-24
Data Files M-25
Table M-1. Changed Characters by Language Group M-26

GLOSSARY OF TERMS AND ABBREVIATIONS . . . N-1

INDEX X-1

How to Use This Manual

This manual is divided into six chapters:

- *Chapter 1. OCL Statements* describes in detail each OCL statement.
- *Chapter 2. SSP Procedures* describes in detail each SSP procedure.
- *Chapter 3. Control Commands* describes in detail each control command.
- *Chapter 4. SSP Utility Programs* describes in detail each SSP utility program.
- *Chapter 5. Writing and Using Procedures* describes the statements that you can use and the rules that you must follow if you write your own procedures.
- *Chapter 6. Programming Considerations* describes how to:
  - Specify region size for a job or job step.
  - Store library members in disk or diskette files.
  - Use display screen formats.
  - Change parameters when performing IPL.
  - Ensure that halts are issued when printer alignment is required.

Note: Conceptual information related to the use of OCL statements, SSP procedures, and SSP utility programs that used to appear in Chapter 6 is now located in the *Concepts and Design Guide.*

DETAILED INFORMATION

If you need detailed information about a specific OCL statement, SSP procedure, control command, or SSP utility program, refer to the appropriate description in one of the first four chapters. If you need detailed information about writing your own procedures, see Chapter 5.

SUMMARY INFORMATION

If you want to find out which OCL statements, SSP procedures, control commands, or SSP utility programs can be used to perform a desired general function, see *SSP Function Summary* in the *Introduction.*

PROGRAMMING CONSIDERATIONS

Chapter 6 contains information that you should consider when using your System/34. You should scan all of Chapter 6 and carefully read the sections that pertain to your system.

READER'S COMMENTS

If you find an error or have a suggestion for improving this publication, please use the Reader's Comment Form at the back of the manual. If the form has been removed, address your comments to IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901.

Conventions Used for Describing Statement Formats

When statement formats are described in this manual, capitalized expressions, brackets, braces, parentheses, and underlining have special meanings:

- *Capitalized expressions* must be entered as they are shown in the statement formats. Numbers and special characters within a capitalized expression also must be entered as they are shown. An expression that is not capitalized must be replaced with a value that is appropriate. For example, the statement:

  // FORMS LINES-value

  could be coded:

- *Brackets* ([ ]) shown in a statement format are not coded as part of the statement. Brackets indicate that the expression they enclose is optional. If a list of values is enclosed in brackets, you can choose not to code a value or to code one of the items in the list. For example,

  [mmddyy
ddmmyy
yymmdd]

  indicates that you need not code a date, but if you choose to code a date, it must be in one of the three formats shown: mmddyy, ddmmyy, or yymmdd.

- *Braces* ({ }) shown in a statement format are not coded as part of the statement. Braces indicate that one of the values enclosed within the braces must be coded. For example,

  [REQD-{YES
NO}]

  indicates that either REQD-YES or REQD-NO must be coded if the REQD parameter is used. (Remember, the brackets indicate that the REQD parameter is optional.)

- *Parentheses* (( )) shown in a statement format are not coded as part of the statement. Parentheses indicate that the value enclosed within the parentheses is an abbreviation and can be entered in place of the characters above the parentheses. For example,

```
RESTART  PRT,[page number] [,ws-id ]
(T)    (P)
```

  indicates that a RESTART control command that restarts printing at the beginning of the printed output (see Chapter 3 for a description of control commands) can be entered in one of two forms:

```
RESTART  PRT
   or
T  PRT
```

- *Underlining* identifies default values. The system automatically uses the default value if you do not code an optional value. For example,

$$\left[\text{REQD-}\left\{\begin{matrix}\text{YES}\\ \underline{\text{NO}}\end{matrix}\right\}\right]$$

  indicates that the system assumes REQD-NO if you do not code the REQD parameter.

Notes:

1. The statement formats often indicate that commas are required (commas are not shown in brackets) preceding parameters that are optional, whether the optional parameters are coded or not. The commas are shown in this manner to remind you that if a positional parameter is omitted, a comma must be entered to indicate the position of the omitted parameter when one or more parameters are coded in positions that follow the omitted parameter. For example,

$$\text{BACKUP vol-id,}\left[\begin{matrix}\text{retention days}\\ \underline{1}\end{matrix}\right],\left[\begin{matrix}\text{label}\\ \underline{\#\text{LIBRARY}}\end{matrix}\right],\left[\begin{matrix}\underline{\text{S1}}\\ \text{S2}\\ \text{S3}\\ \text{M1.nn}\\ \text{M2.nn}\end{matrix}\right],\left[\begin{matrix}\text{NOAUTO}\\ \underline{\text{AUTO}}\end{matrix}\right]$$

   indicates that if the second parameter is not coded but the third parameter is, a comma should be coded in place of the missing parameter, as in:

```
BACKUP BACKØ1,,BACKØ1
```

   For procedure commands, commas following the last parameter coded are optional. For control commands, commas cannot be entered following the last parameter.

2. Many procedures prompt for required parameters if they are not coded on the command statements that call the procedures.

The IBM System/34 SSP (system support program product) controls the execution of all jobs on your system and must be in main storage before jobs are run. The SSP resides on disk or diskettes and is placed into main storage by a process called IPL (initial program load), which the system operator initiates.

In addition to the parts of the SSP that control the execution of jobs, many utility programs and procedures are provided. These programs and procedures can be used to perform often-required functions.

This manual provides the information you need to communicate with and use the various parts of the SSP.

OCL STATEMENTS

The OCL (operation control language) statements are your means of communicating with the portion of the SSP that controls job execution. They provide the SSP with all the information it must have about jobs to be run.

Chapter 1 describes the OCL statements in detail.

UTILITY PROGRAMS AND UTILITY CONTROL STATEMENTS

Utility programs are supplied by IBM as part of the SSP. When a system utility program is run, OCL statements identify the program and supply to the SSP any information that it requires about the program. In addition to the OCL statements, utility control statements define the functions to be performed by the utility program. Following is an example of the OCL statements and utility control statements that could be coded to use the $MAINT utility program to copy a procedure from a diskette to the system library on disk:

```
// LOAD $MAINT
// FILE NAME-PAYROLL,UNIT-I1
// RUN
// COPY FROM-DISK,TO-F1,FILE-PAYROLL
// END
```

Chapter 4 describes the utility programs and the OCL statements and utility control statements required for running the utility programs.

SSP PROCEDURES AND PROCEDURE COMMANDS

Procedures provide a means of selecting sets of frequently used OCL statements and utility control statements without having to recode and reenter those statements each time they are required. A procedure is a set of OCL statements and utility control statements that is processed by the SSP when a procedure command (a special form of the INCLUDE OCL statement) that identifies the procedure is read. A procedure command is a simple statement that contains the name of the procedure to be run and information (parameters) that defines the function to be performed by the procedure. The actual statements processed when the procedure is run depend upon the information specified on the procedure command.

Many procedures are supplied as part of the SSP. Most of these procedures cause SSP utility programs to be run. The procedures make it possible for you to use the SSP utility programs without having to code all of the required OCL statements and utility control statements. For example, the following procedure command could be coded to use the TOLIBR procedure to copy a procedure from a diskette to the system library on disk:

```
TOLIBR PAYROLL
```

Note: The preceding procedure command causes the same function to be performed as the OCL statements and utility control statements shown under *Utility Programs and Utility Control Statements* earlier in this introduction.

Chapter 2 describes the SSP procedures in detail.

WRITING YOUR OWN PROCEDURES

You can write your own procedures for System/34. Chapter 5 describes the special statements you can use when you are writing your own procedures.

CONTROL COMMANDS

Control commands are statements used by the system operator and display station operators to control the operation of the system and the display stations. These commands can be entered only from the keyboard at the system console or from a command display station keyboard. Certain commands can be entered from the system console, others can be entered only from display stations (including the system console when it is being used as a display station), and others can be entered from either the system console or a display station.

Chapter 3 describes the control commands in detail.

PROGRAMMING CONSIDERATIONS

For a discussion of programming considerations related to the use of OCL statements, SSP procedures, and SSP utility programs, see the *Concepts and Design Guide*.

SSP FUNCTION SUMMARY

Figure 1 lists many general functions that you may want to perform using the SSP. The OCL statements, SSP procedures, control commands, and SSP utility programs that can be used to perform each function are listed. Detailed descriptions of OCL statements, SSP procedures, control commands, and SSP utility programs are in Chapters 1 through 4, respectively.

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Changing your system configuration		ALTERBSC ALTERSDL OVERRIDE SET SPECIFY	ASSIGN VARY	$SETCF
Compiling and executing programs				
Assigning the following:				
Attributes of a program	ATTR COMPILE			
Job date	DATE	DATE		
Priority	ATTR		PRTY	
Region size	REGION			
Communicating between programs	LOCAL SWITCH			
Compiling a program	COMPILE			
Canceling a job			CANCEL	
Defining a disk or diskette file for use by a program	FILE			
Defining a display station for use by a program	WORKSTN			
Defining a printer for use by a program	PRINTER			
Loading a program to be run	LOAD			
Pausing during procedure execution	PAUSE			
Placing a job on the input job queue	JOBQ		JOBQ	

Figure 1 (Part 1 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Compiling and executing programs (continued)				
Restarting a checkpointed job		CRESTART		$RSTRT
Running a procedure	INCLUDE EVOKE			
Running (executing) a program	RUN			
Data communications				
Changing data communication configuration	COMM SESSION	ALTERBSC ALTERSDL OVERRIDE SPECIFY DEFINEID ENABLE DISABLE		$SETCF $IDSET $IENBL $IEDS
Disk file processing				
Copying a spool file to disk		COPYPRT		$UASF
Creating a disk file	FILE	BLDFILE		$FBLD
Deleting a disk file	FILE	DELETE		$DELET
Examining the contents of a disk file		DISPLAY LISTFILE		$COPY
Moving a disk file to a diskette(s)		SAVE TRANSFER POST		$COPY $BICR $POST
Moving a diskette file to disk		RESTORE TRANSFER POST		$COPY $BICR $POST
Maintaining disk files and their locations		COMPRESS ORGANIZE RENAME		$PACK $FREE $COPY $RENAM

Figure 1 (Part 2 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Disk file processing (continued)				
Obtaining information about a disk file or files		CATALOG LISTFILE		$LABEL $COPY
Reserving an area for scratch and job files	RESERVE			
Sorting index keys		KEYSORT		$DDST
Diskette processing				
Initializing a diskette		INIT		$INIT
Copying a diskette		COPYI1		$DUPRD
Deleting a diskette file		DELETE		$DELET
Examining the contents of a diskette file		LISTFILE		$COPY $BICR $MAINT
Moving a disk file to a diskette		SAVE TRANSFER POST		$COPY $BICR $POST
Moving a diskette file to a disk		RESTORE TRANSFER POST		$COPY $BICR $POST
Obtaining information about a diskette file or files		CATALOG LISTFILE		$LABEL $COPY
Display screen functions				
Building a message load member		CREATE		$MGBLD
Generating screen formats and menus		BLDMENU FORMAT		$BMENU $SFGR
Controlling information displayed on the system console	LOG SYSLIST	LOG SYSLIST	IDELETE	

Figure 1 (Part 3 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Display station functions				
Modifying the display station environment		SET	ASSIGN VARY	$SETCF
Defining a display station for use by a program	WORKSTN			
Displaying messages at a display station	// * MSG		MSG	
Display station operations:				
Controlling display station activity	MENU OFF		MENU MODE OFF STATUS TIME PRTY	
Communicating with the system console or other display stations			MSG	
Using the input job queue			CANCEL CHANGE JOBQ PRTY START STATUS STOP	
Library use and maintenance				
Assigning an active user library	LIBRARY	SET		$SETCF
Backing up the system library		BACKUP		$BACK
Condensing (removing gaps from) a library		CONDENSE		$MAINT
Copying library members to disk or diskettes		FROMLIBR		$MAINT

Figure 1 (Part 4 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Library use and maintenance (continued)				
Copying members from disk or diskette into a library		TOLIBR JOBSTR		$MAINT
Creating a new user library		BLDLIBR		$MAINT
Copying non-SSP members from one library to another		LIBRLIBR		$MAINT
Copying all non-SSP members from a library to a diskette file		SAVELIBR		$MAINT
Deleting a user library		DELETE		$DELET
Deleting members from a library		REMOVE		$MAINT
Displaying the contents of a library		LISTFILE LISTLIBR		$MAINT
Reloading a system library that was backed up on diskettes		RELOAD		$LOADI
Renaming a user library		RENAME		$RENAM

Figure 1 (Part 5 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
Printer functions				
Changing the printer configuration		SET	ASSIGN VARY	$SETCF
Controlling information printed on the system printer	LOG SYSLIST	LOG SYSLIST		
Controlling printer and printed forms for program output	FORMS IMAGE PRINTER	LINES SET		$SETCF
Controlling spooling functions	PRINTER	COPYPRT	CANCEL CHANGE HOLD RELEASE RESTART START STATUS STOP	$UASF
Problem determination		HISTORY HISTCRT	STATUS	$HIST $HSML

Figure 1 (Part 6 of 7). SSP Function Summary

Function	OCL Statement	SSP Procedure	Control Command	SSP Utility Program
System console functions				
Displaying messages on the system console	// *		MSG	
Controlling information displayed on the system console	LOG SYSLIST	LOG SYSLIST	IDELETE	
System operations:				
Managing spooling			CANCEL CHANGE HOLD RELEASE RESTART START STATUS STOP	
Managing the input job queue			CANCEL CHANGE START STATUS STOP	
Controlling the device status			ASSIGN STOP VARY	
Handling messages			MSG REPLY	
Responding automatically to system messages		RESPONSE		$ARSP
System security		PRESTOR PRMENU PROF PRSAVE PRSRC PRSRCID		$PRST $PRMN $PROF $PRSV $PRES $PRON

Figure 1 (Part 7 of 7). SSP Function Summary

Chapter 1. OCL Statements

In this chapter, each OCL statement is described separately. The following information is given for each statement:

- The function of the statement
- The placement of the statement in relation to other statements and the circumstances under which it is needed
- The format of the statement
- The contents of the statement (the parameters that can be used with it)

TYPES OF INFORMATION CONVEYED IN OCL STATEMENTS

The OCL statements contain two types of information: an *identifier* and *parameters*. An identifier distinguishes one OCL statement from another; a parameter supplies information to the SSP. The general form of an OCL statement is:

```
// identifier parameter-1,parameter-2, . . .parameter-n
```

Identifiers

Every OCL statement except a procedure command requires a statement identifier. A procedure command is a special form of the INCLUDE OCL statement.

Most OCL statements begin with // followed by one or more blanks. The OCL statement identifiers that require // are:

ATTR	FORMS	LOCAL	PAUSE	SESSION
COMM	IMAGE	LOG	PRINTER	SWITCH
COMPILE	INCLUDE	MEMBER	PROMPT	SYSLIST
DATE	JOBQ	MENU	REGION	WORKSTN
EVOKE	LIBRARY	MSG	RESERVE	*(message)
FILE	LOAD	OFF	RUN	**(message)

For example, in the statement:

the statement identifier is LOAD.

Identifiers that do not require // are:

* (comment)
/* (end of data)

For example, in the statement:

```
* END OF JOB
```

the statement identifier is *. Because // does not precede the *, the * indicates the statement is a documentation comment. (// * at the beginning of a statement indicates the statement is a displayed message.)

Parameters

Parameters are either *symbolic* (positional) or *keyword* parameters. In the following statement, $COPY is a *symbolic parameter*–the name of a system utility program:

```
// LOAD $COPY
```

NAME-COPYIN, UNIT-F1, and LABEL-label are *keyword parameters* in the following statement:

```
// FILE NAME-COPYIN,UNIT-F1,LABEL-label
```

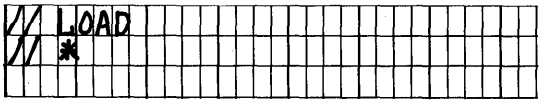
A keyword parameter contains a *keyword* (NAME, UNIT, and LABEL are the keywords in the preceding OCL statement) that distinguishes the parameter from other parameters, just as statement identifiers distinguish one OCL statement from another. In addition to a keyword, a keyword parameter usually contains a *value* (COPYIN and F1 are values in the preceding OCL statement).

GENERAL OCL CODING RULES

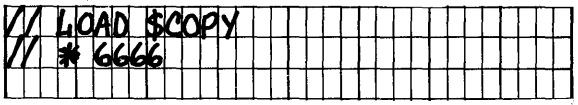
The OCL statement formats described in this manual can include special characters, such as //, and words written in capital letters, such as the FILE statement parameter, LABEL. These special characters and words must be entered exactly as shown in the statement descriptions given in this manual. Words written in lowercase letters, such as label, represent information that you must supply. OCL statements cannot exceed 120 characters, except OCL statements that contain keyword parameters. (See *Continuation* later in this chapter.)

Additional coding rules are:

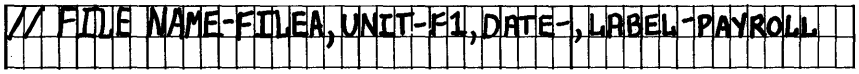
- The first character (* or /) of an OCL statement must be keyed in position 1. For example, // must be entered in positions 1 and 2.
- One or more positions between the // and the statement identifier must be blank. For example:



- One or more positions between the statement identifier and the first parameter must be blank. For example:



- If you need to include more than one parameter, use a comma to separate the parameters. No blanks are allowed within or between parameters. Anything following the first blank after a parameter is considered to be a comment. (See *Comments* later in this chapter.)
- If you are writing keyword parameters, place the keyword first and use a hyphen (-) to separate the keyword from the value.
- If a value is not specified with a keyword parameter, the parameter is ignored. For example:



When this statement is processed, the DATE keyword parameter is ignored.

Continuation

Expressing a single statement in two or more records is called *continuation*. Any OCL statement that contains keyword parameters can be continued.

Rules for using continuation are:

- Begin each new record with // in positions 1 and 2.
- Leave one or more blanks between the // and the first parameter in the record.
- Place a comma after the last parameter in every record except the last record. The comma, followed by a blank, tells the SSP that the statement is continued in the next record.

In the first of the following two examples of continued FILE statements, five records are used to express a single FILE statement. In the second example, two records express one FILE statement.

Example 1:

```
// FILE NAME-TRANS,
//      UNIT-F1,
//      LABEL-TRANS1,
//      RECORDS-225,
//      RETAIN-T
```

Note: Continuing OCL statements as in example 1 increases processing time.

Example 2:

```
// FILE NAME-TRANS,UNIT-F1,LABEL-TRANS1,
//      RECORDS-225,RETAIN-T
```

Comments

Comments are usually used to explain the purpose of the OCL statements and utility control statements stored in a procedure. Comments in a procedure are displayed when the procedure is displayed. Comments are not displayed when the procedure is executed. Comments can contain any combination of characters except question marks (?). Comments can be included in the following places:

- Following the * on the OCL comment statement.

```
* THIS IS AN EXAMPLE OF A COMMENT STATEMENT
```

  The comment here is THIS IS AN EXAMPLE OF A COMMENT STATEMENT.

- After the last parameter in a statement. Leave one or more blanks between the last parameter and your comment.

```
// LOAD SCOPY  LOAD THE DISK COPY UTILITY
```

  In this example, the comment is LOAD THE DISK COPY UTILITY.

- After the comma that follows the last parameter in an OCL record that is continued.

```
// FILE NAME-TRANS,          FILE STATEMENT
//      UNIT-F1,             FOR TRANSACTION FILE
//      LABEL-TRANS1,
//      RECORDS-225,
//      RETAIN-T
```

In this example, the first two records of the FILE statement contain comments.

- After the identifier on statements without parameters. Leave one or more blanks between the identifier and your comments.

```
// RUN  RUN THE DISK COPY UTILITY
```

The comment here is RUN THE DISK COPY UTILITY.

- After the procedure name in an INCLUDE OCL statement if the statement has parameters but none of them is coded. Leave a blank after the procedure name, code a comma, leave a blank after the comma, and code the comment.

```
// INCLUDE WEEKLY , RUN WEEKLY PROCEDURE
                    or
// WEEKLY , RUN WEEKLY PROCEDURE
                    or
WEEKLY , RUN WEEKLY PROCEDURE
```

The comment in these statements is RUN WEEKLY PROCEDURE.

Notes:

1. The three forms of the INCLUDE OCL statement are described later in this chapter.
2. An INCLUDE OCL statement that calls an MRT (multiple requestor terminal) procedure or that calls a procedure that passes data, not parameters, cannot contain a comment.

Although comments are useful for explaining the statements within a procedure member, a large number of comments can significantly increase the execution time of the procedure. By using block comments, you can provide a more efficient means of documenting a procedure.

ATTR Statement

Function

The ATTR statement:

- Assigns priority to a job or a job step
- Changes the MRTMAX value for a program
- Releases the requesting display station from the next job step when the job step begins executing
- Overrides the NEP attribute for a program

Placement

The ATTR statement can appear anywhere among the OCL statements except between a LOAD statement and a RUN statement.

Format

```
// ATTR [ PRIORITY-{HIGH|YES|MEDIUM|NO|LOW} ] [,MRTMAX-nnn]

[,RELEASE-{YES|NO}] [,NEP-{YES|NO}]
```

Parameters

PRIORITY: The PRIORITY parameter assigns priority to a job or job step. The system assigns system resources in order of decreasing priority. The order of priority is: HIGH or YES, MEDIUM, NO, and LOW. For example, if PRIORITY-MEDIUM is specified, system resources are assigned to the job or job step after they are assigned to any higher priority (HIGH or YES) job or job step, but before they are assigned to any lower priority (NO or LOW) job or job step. PRIORITY-HIGH and PRIORITY-YES are equivalent. The PRIORITY parameter can be specified more than once in a job; this parameter takes effect as soon as it is encountered.

If a display station operator enters the PRTY control command with HIGH, YES, MEDIUM, NO, or LOW specified before submitting a job, any PRIORITY parameters on the ATTR statement are ignored. If the system console operator enters the PRTY control command with HIGH, YES, MEDIUM, NO, or LOW specified after the job has started, any subsequent PRIORITY parameters on the ATTR statement are ignored.

MRTMAX: The MRTMAX parameter specifies the number of active, requesting display stations or SSP-ICF sessions that can be attached to the program executed in the next job step (leading zeros need not be coded). nnn changes the MRTMAX value specified on the COMPILE statement when the program was compiled; nnn cannot exceed the MRTMAX value specified on the COMPILE statement or in the RPG II file description specification. MRTMAX is valid only if an MRTMAX value of one or more was specified on the COMPILE statement. Only one ATTR statement specifying the MRTMAX parameter can be specified for a job step.

RELEASE: The RELEASE parameter specifies whether or not the display station remains allocated to the next job step. RELEASE-YES releases the display station when the job step begins executing. If the job step is the last or only step in a procedure, the COMMAND display appears on the display screen at the display station and the display station operator can submit another job. If the job step is not the last step in a procedure, only the step is released. The released step can execute concurrently with subsequent steps of the same procedure. The requesting display station remains allocated to the steps that follow the released step. If RELEASE-YES is specified, the following points should be considered:

- Existing job files cannot be passed to the released step.

  For information about job files, see the description of the FILE statement for disk files later in this chapter.

- Job files created by the released step are treated as scratch files; that is, those files cannot be used by subsequent steps in the procedure.

- RELEASE-YES is ignored if the OCL statements for the job step contain a WORKSTN statement that specifies REQD-YES for the requesting display station.

- System messages issued while the released step is being run are displayed on the system console, not on the display screen at the requesting display station.

- A released step uses a copy of the external indicators for the requesting display station and the display station local data area as they exist when the released step is initiated. If the released step modifies the display station local data area or the external indicators, the modifications are in effect only during the job step. The changes are not seen by subsequent steps in the procedure or by subsequent jobs submitted from the display station.

- If RELEASE-YES is specified for a job step that runs an MRT (multiple requestor terminal) program that is also defined as an NEP (never-ending program), the MRT program is initiated but has no requesting display stations attached. The MRT program then waits for the next requesting display station.

- If the released step is an RPG II program using SUBR01 to read source from SYSIN, unpredictable results may occur.

The RELEASE parameter is ignored for jobs on the input job queue. If the RELEASE parameter is not specified, RELEASE-NO is assumed.

NEP: The NEP parameter specifies whether or not the program is an NEP (never-ending program). An NEP is defined as a long-running program. Any system resources, except for shared files, that are allocated to an NEP are not available to other jobs. NEP-YES specifies that the program is an NEP. NEP-NO specifies that the program is not an NEP.

You should specify NEP-YES for a program that uses one or more system resources that cannot be shared (for example, a disk file that is not shared) for a period of time greater than the time a display station operator should have to wait for the SSP to initiate a program. If a display station operator attempts to run a program that uses a nonsharable system resource being used by an NEP, the SSP does not wait for the resource to become available but issues a message indicating that the requested program cannot be initiated because an NEP is using the required resource. The display station operator can then cancel the job or retry allocating the resource. On the other hand, if the display station operator attempts to run a job that uses a nonsharable system resource being used by a long-running program for which NEP-YES was not specified, the SSP waits for the resource to become available. While the SSP is waiting, the display station operator may use inquiry (by pressing the Attn key) to cancel the job or run other jobs.

The NEP parameter overrides the NEP value specified on the COMPILE statement. If the NEP parameter is not specified, the system uses the NEP value specified on the COMPILE statement. Only one ATTR statement specifying the NEP parameter can be specified for a job step.

Special considerations exist for jobs run from the input job queue and for MRT (multiple requestor terminal) programs:

- All programs run from the input job queue are run as NEPs, unless NEP-NO is specified on the ATTR OCL statement.

- For an MRT program, if the NEP parameter is not specified on the ATTR OCL statement and if NEP-YES is not specified on the COMPILE OCL statement, other programs will not wait for nonshared resources being used by the MRT program. However, when the MRT program releases its last requesting display station, the program goes to end-of-job processing.

- If NEP-YES is specified for an MRT program, other programs will not wait for nonshared resources being used by the MRT program. The MRT program will not terminate when it releases its last requesting display station. Instead, the MRT program will wait until it is requested by another display station. Normally, an MRT program with NEP-YES specified will not terminate until after the system operator enters the STOP SYSTEM control command and all requesting terminals are released. (The system operator can abnormally terminate the job by using the CANCEL control command.)

Example

Assign priority to the job and release the requesting display station from the next job step:

COMM Statement

Function

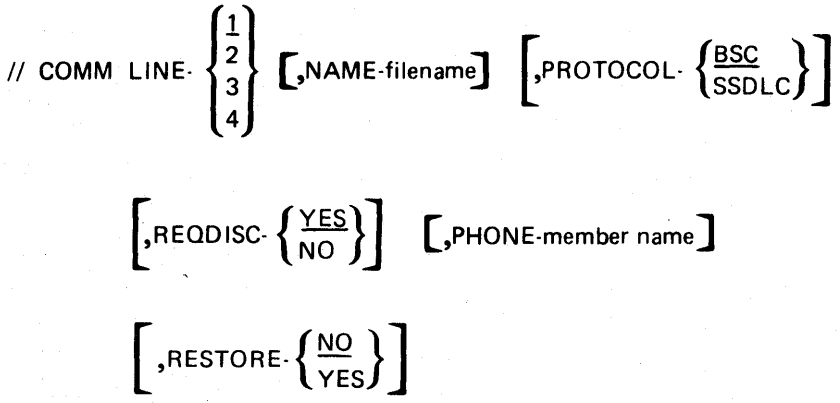
The COMM statement:

- Assigns a line number to the program
- Specifies the communication file name to be used with the program
- Specifies the line protocol to be used with the program
- Specifies whether or not the request discontact network services request will be sent

Placement

The COMM statement must be placed between the communications program's LOAD and RUN statements.

Format



Parameters

LINE: The LINE parameter indicates the physical line number to be associated with the communications program.

NAME: The NAME parameter specifies the communications file name defined in the preopen SNA DTF. This parameter is required only if PROTOCOL-SSDLC is specified.

PROTOCOL: The PROTOCOL parameter indicates whether Binary Synchronous Communications (BSC) or Secondary Synchronous Data Link Control (SSDLC) protocol is to be used.

REQDISC: The REQDISC parameter tells System/34 SNA whether or not to send a request discontact network services request. This parameter is used only if PROTOCOL-SSDLC is specified. If request discontact is sent, the line connection to the host system ends when the SNA session terminates. If the request discontact is not sent, the connection remains established until it is ended by some other means. If consecutive or concurrent SNA sessions are to be run, specify REQDISC-NO.

PHONE: The PHONE parameter specifies the phone list created by the DEFINEPN procedure or the DEFINX21 procedure.

RESTORE: The RESTORE parameter specifies whether a phone list used by a previous step in the job should be restored to its original state. No numbers are called and the retry count is reset. RESTORE-YES specifies that the SSP retrieves the phone list from the current user library. RESTORE-NO indicates that no check will be made to determine whether the previous phone list and the specified phone list are from different libraries.

Notes:

1. If a step of a procedure references a phone list that was specified in a previous job step, no check is made to see whether the previous phone list and the specified phone list are from different libraries. If the phone list names match, the system assumes that the specified list is the same list that was previously referenced.

   To ensure that the SSP will not use the previously referenced list, specify RESTORE-YES. This causes the SSP to retrieve the phone list from the current user library.
2. If an MRT procedure or a released job step encountered in a multiple-step procedure references a phone list via the COMM statement, it is given its own copy of the list.

COMPILE Statement

Function

The COMPILE statement supplies information required when a library source member is compiled. The COMPILE statement:

- Identifies the library source member that contains the source program to be compiled. A source program is a collection of statements, such as RPG II specifications, that can be translated into a load member.
- Identifies the library that contains the source program.
- Identifies the library that will contain the compiled load member. The load member can be loaded and executed using the LOAD and RUN statements.
- Specifies the maximum number of active, requesting display stations that can be attached to the program, if the program is an MRT program.
- Identifies the program as an NEP (never-ending program).

Placement

The COMPILE statement must be placed between the LOAD and RUN statements of the job step that compiles the source program. If the source program is in the procedure or keyboard job stream, the COMPILE statement may be omitted. However, if an in-stream source program is used, it must immediately follow the RUN statement. The end of the source program must be indicated by a /* (end of data) OCL statement.

Format

```
// COMPILE SOURCE-name [,INLIB-{name
                              #LIBRARY}]

      [,OUTLIB-{name
                #LIBRARY}] [,MRTMAX-{nnn
                                     0}]

      [,NEP-{YES
             NO}]
```

Parameters

SOURCE: The SOURCE parameter specifies the name of the source member that contains the source program to be compiled.

INLIB: The INLIB parameter specifies the name of the library that contains the source program. If INLIB is specified, only that library is searched; if INLIB is not specified, the system library (#LIBRARY) is searched.

OUTLIB: The OUTLIB parameter specifies the name of the library that will contain the compiled load member. If OUTLIB is not specified, the system library (#LIBRARY) is assumed. The name of the load member is specified in the source program.

MRTMAX: The MRTMAX parameter identifies the program as an MRT program and specifies the maximum number (nnn) of active, requesting display stations that can be attached to the program. nnn should not be greater than the number specified in the RPG II file description specifications. If nnn is 0 or if MRTMAX is not specified, the object program is not an MRT program. If MRTMAX is specified, it can be changed by an ATTR statement when the object program is executed.

NEP: The NEP parameter specifies if the program is an NEP (never-ending program). An NEP is defined as a long-running program. Any system resources, except for shared files and the spool file, that are allocated to an NEP are not available to other jobs. NEP-YES specifies that the program is an NEP. If NEP-NO is specified or if the NEP parameter is not used for an SRT (single requestor terminal) program that is not run from the input job queue, the program is not executed as an NEP. The NEP attribute can be changed by an ATTR statement when the object program is executed.

Special considerations exist for jobs run from the input job queue and for MRT (multiple requestor terminal) programs:

- All programs run from the input job queue are run as NEPs, unless NEP-NO is specified on the ATTR OCL statement.

- For an MRT program, if the NEP parameter is not specified on the ATTR OCL statement and if NEP-YES is not specified on the COMPILE OCL statement, other programs will not wait for nonshared resources being used by the MRT program. However, when the MRT program releases its last requesting display station, the program goes to end-of-job processing.

- If NEP-YES is specified for an MRT program, other programs will not wait for nonshared resources being used by the MRT program. The MRT program will not terminate when it releases its last requesting display station. Instead, the MRT program will wait until it is requested by another display station. Normally, an MRT program with NEP-YES specified will not terminate until after the system operator enters the STOP SYSTEM control command and all requesting terminals are released. (The system operator can abnormally terminate the job by using the CANCEL control command.)

For information about NEPs, SRTs, and MRTs, see the *Concepts and Design Guide.*

Example

The RPG II source member named PROG3, in the user library called ULIB1, is to be compiled. The compiled load member will be placed in the system library. The object program will be executed as an NEP.

```
// LOAD *RPG
// MEMBER PROGRAM1-*RPG*CPL1,PROGRAM2-*RPG*CPL2
   .
   .
   .
// COMPILE SOURCE-PROG3,INLIB-ULIB1,NEP-YES
// RUN
```

Note: The LOAD statement, MEMBER statement, and RUN statement are described later in this chapter. For information about libraries, see the *Concepts and Design Guide*; for information about RPG II, see the *RPG II Reference Manual*.

DATE Statement

Function

A DATE statement that is entered anywhere other than between a LOAD statement and a RUN statement changes the session date. (A session begins when the display station operator signs on and, normally, ends when the display station operator enters the OFF control command.) If a DATE statement or the DATE procedure is not used to establish a session date, the system date specified during IPL is used as the session date.

A DATE statement that is entered between a LOAD statement and a RUN statement specifies the program (job step) date. When the program ends, the program date is set to the session date. If a DATE statement is not entered or if the DATE procedure is not run between a LOAD statement and a RUN statement, the session date is used as the program date. If two or more DATE statements are between a LOAD statement and a RUN statement, the last DATE statement is used.

Notes:

1. The program date is used to determine the file retention period for diskette files used by the program and is printed on printed output. The program date is also used as the creation date of any disk or diskette files created by the program.
2. If a job is placed on the input job queue, the program date when the job was placed on the queue is assigned to the job.
3. If 2400 hours occurs, the system date is automatically updated, but the session date and program date are not.

Placement

A DATE statement can be entered anywhere within the OCL statements.

Format

```
// DATE {mmddyy}
        {ddmmyy}
        {yymmdd}
```

Parameters

The date specified on the DATE statement must be in current session date format. The session date can be in any of three formats: month-day-year (mmddyy), day-month-year (ddmmyy), or year-month-day (yymmdd). The STATUS control command can be used to determine the current session date format, and the SET procedure can be used to change the current session date format.

Month, day, and year must each be 2-digit numbers, but leading zeros in month and day can be omitted when punctuation is used. In the punctuated form, any characters except commas (,), single quotes ('), numbers, and blanks can be used as punctuation. However, question marks (?), slashes (/), and hyphens (-) should not be used as punctuation if the DATE statement is in a procedure.

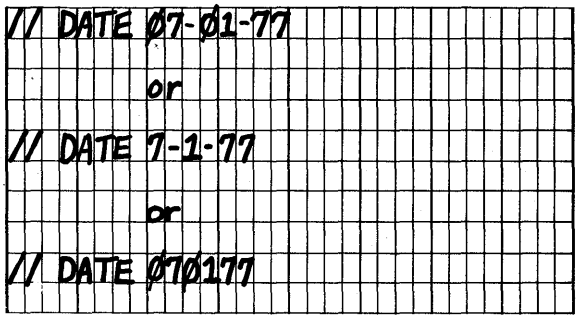
The date can be entered with or without punctuation. For example, July 24, 1977 can be specified in any of the following ways:

7-24-77	mm-dd-yy
24-7-77	dd-mm-yy
77-7-24	yy-mm-dd
072477	mmddyy
240777	ddmmyy
770724	yymmdd

A date of 000000 (all zeros) is invalid.

Example

The DATE statement for July 1, 1977:



EVOKE Statement

Function

The EVOKE statement can be used to run a procedure that contains a single step or multiple steps. When a procedure is evoked, it begins executing as a separate job, while control returns immediately to the calling procedure. Thus, it is possible to have several procedures executing at once as a result of several EVOKE statements.

When a job is evoked, the priority of the evoked job is the same as the priority of the job that issued the evoke.

Messages issued from the evoked job are displayed at the system console, not at the display station (if other than the system console) that issued the EVOKE statement. When a procedure is evoked, the evoked procedure uses a copy of the external indicators (also known as UPSI switches) and the local data area as they exist on the requesting display station. If the evoked procedure modifies the external indicators or local data area, the modifications are in effect only for the evoked job; the changes are not seen by subsequent jobs run or evoked at the requesting display station.

If no REGION statement is specified for the procedure containing the EVOKE statement, the evoked procedure uses the session region size specified during system configuration or specified by the SET procedure. If a REGION statement is specified in the procedure containing the EVOKE statement, the evoked procedure uses the region size specified in the REGION statement.

CAUTION
Do not overcommit the system with too many EVOKE statements. To do so might degrade system performance.

If there are insufficient storage resources for processing evoked procedures, a halt is issued with a 1 or 3 option. The 1 option retries the EVOKE statement, while the 3 option terminates the job containing the EVOKE statement.

Placement

The EVOKE statement can be placed anywhere among the OCL statements except between the LOAD and RUN statements.

Format

// EVOKE procedure name [parameter(s) / data]

Parameters

procedure name: The valid name of any procedure.

parameters: These can be any parameters associated with the evoked procedure. The parameters are positional parameters that might or might not be required, depending on the procedure they are passed to. (Positional parameters are parameters that must appear in the same position, relative to other positional parameters, in the same statement.)

A parameter can be any combination of characters except question marks (?), commas (,), single quotes ('), slashes (/), hyphens (-), or blanks. The number of characters in a parameter must not exceed 8. A maximum of 11 parameters, separated by commas, can be passed with an EVOKE statement.

data: The data starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. The data is passed to the program on its first input operation from the requesting display station.

Example

In the following example, a set of OCL statements includes an EVOKE statement which calls the multistep procedure EMPUPDT. When EMPUPDT is evoked, it starts executing while control immediately returns to the calling procedure. Thus, both procedures can execute simultaneously and independently:

```
// ...
// ...
// EVOKE EMPUPDT LIST,PL
// ...
// LOAD PAYROLL
// RUN
```

FILE Statement (for Disk Files)

Function

The FILE statement supplies the SSP with information about disk files. The SSP uses this information to read records from and write records on the disk.

For more information about System/34 disk file concepts, see the *Concepts and Design Guide*.

Placement

A FILE statement is required for each new disk file that a program creates, and for each of the existing disk files that a program uses. The FILE statement must follow the LOAD statement and precede the RUN statement.

Format

```
// FILE NAME–filename [,UNIT–F1] [,LABEL–file label]

[,RECORDS–number]  [,LOCATION–{block number}]
[,BLOCKS–number]   [          {A1          }]
                   [          {A2          }]

[,RETAIN–{S}]  [,DATE–{mmddyy}]
         {J}   [      {ddmmyy}]
         {T}   [      {yymmdd}]
         {P}

[,DISP–{NEW}]  [,EXTEND-value]  [,DFILE-{YES}]
       {OLD}                            {NO }
       {SHR}

[,IFILE–{YES}]  [,BYPASS–{YES}]
        {NO }           {NO }
```

Parameters

NAME: The NAME parameter tells the SSP the name that the program uses to refer to the file. The filename can be any combination of characters (numeric, alphabetic, and special) except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. The first character of a filename must be alphabetic (A through Z, #, $, or @). The number of characters in a filename must not exceed 8.

UNIT: UNIT-F1 indicates that the file is on the disk. This keyword and value need not be specified for a disk file because F1 is the default value for the UNIT parameter.

LABEL: The LABEL parameter tells the SSP the label by which a file is identified on the disk. If the file is being created, the label supplied in the LABEL parameter is used to identify the file on the disk. If the LABEL parameter is omitted from a disk FILE statement, the filename from the NAME parameter is used as the label. If a program refers to an existing file by a filename that differs from the label by which the file is identified on the disk, a LABEL parameter must be supplied.

The label can be any combination of characters (numeric, alphabetic, and special) except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. The first character of a label must be alphabetic (A through Z, #, $, or @). The number of characters in a label must not exceed 8.

If a file label contains a period, the file is a member of a file group. The characters preceding the period identify the file group.

For information about saving all members of a file group, see the description of the SAVE procedure in Chapter 2 or the description of the $COPY utility program in Chapter 4. For information about deleting all members of a file group, see the description of the $DELET utility program in Chapter 4.

RECORDS: The RECORDS keyword parameter specifies the number of records needed for the file. The total space allocated is rounded up to the next block, allowing space to contain at least the number of records indicated. The smallest allocatable unit is one block. For example, if you specify ten 50-byte records, then 2,560 bytes (one block) are allocated. If the RECORDS parameter is used, the number can be up to 8 digits long (including left-justified zeros) but cannot exceed the value 08000000.

BLOCKS: The BLOCKS keyword parameter specifies the number of disk blocks needed for the file. There are 2,560 bytes in one disk block (one block = ten 256-byte sectors). If the BLOCKS parameter is used, the number can be up to 8 digits long but cannot exceed the value 100,811.

You can use the CATALOG procedure to determine the number of disk blocks available for files.

A file cannot be allocated if it contains more than 16,711,408 records.

Either RECORDS or BLOCKS, but not both, can appear in the FILE statement. The keyword must be followed by a number indicating the amount of space needed. Either the RECORDS or BLOCKS parameter must be used for a new file.

LOCATION: The LOCATION parameter specifies the number of the block where a file begins or indicates a preferred disk placement for a new disk file. When a block number is specified, up to 6 digits can be specified. You can use the CATALOG procedure to determine where blocks of available storage are located on the disk.

The following table gives the beginning block number location for disks 2 through 4:

Disk	Beginning Block Number Location
Disk 2	25203
Disk 3	50406
Disk 4	75609

A block number is required:

- If you prefer that a file be placed specifically on disk 3 or 4.
- If DISP is not specified and a new file is being created with the same label and size (RECORDS or BLOCKS) as an existing file. A location that is different from the location of the existing file(s) must be specified. The creation date of the new file must also be different from the creation date of any existing file with the same label.
- When DISP is not specified and an existing file is being overlaid. To overlay an existing file, the label of the existing file, its size (RECORDS or BLOCKS) when it was created, and its location must be specified; or DISP-OLD must be specified; or the file must be a job file. The creation date is changed to the current program date. After the first overlaid existing file, the space allocated value is always maintained in blocks, no matter what the original file allocation was.

Note: Ordinarily, DISP-NEW should be used when creating a new file and DISP-OLD should be used when processing an existing file. Use LOCATION for existing files with caution because both the COMPRESS procedure and the RESTORE procedure move files on disk.

If you do not specify the location (block number) for a new file on the FILE statement, the SSP allocates space for the file according to the following rules:

- If you specify a preferred placement on disk A1 (LOCATION-A1) for a multiple-disk configuration, the SSP allocates the file in the first segment (lowest address) of available storage on disk A1 that is large enough to contain the file. If space is not available on disk A1, the SSP attempts to allocate the file on disk A2.

- Although disks 2 through 4 are physically separate, the system treats these disks as one logical disk: A2 is the logical name. If you specify a preferred placement on disk A2 (LOCATION-A2) for a multiple-disk configuration, the SSP allocates the file in the last segment (highest address) of available storage on logical disk A2 that is large enough to contain the file. For example:
  - On a three-drive system, the file is placed in the last available segment of disk 3.
  - On a four-drive system, the file is placed in the last segment of disk 4.

  If not enough space is available on the last physical disk of logical disk A2, the SSP attempts to find space on the other disks that make up A2. If not enough sapce is available on logical disk A2, the SSP attempts to allocate the file on disk A1.

- If you do not specify a preferred disk placement for a multiple-disk configuration, the SSP attempts to place permanent and temporary files on disk A2 and scratch and job files on disk A1. The SSP allocates permanent and temporary files in the last segment (highest address) of available storage on disk A2 that is large enough to contain the file; the SSP allocates scratch and job files in the first segment (lowest address) of available storage on disk A1 that is large enough to contain the file. In both cases, if space is not available on the first disk searched, the SSP continues its search on the other disk. The SSP can allocate a file that spans the disk juncture.

- If you specify LOCATION-A1, or do not specify LOCATION for a single-disk configuration, the SSP allocates the file in the first segment (lowest address) of available storage that is large enough to contain the file. If you specify LOCATION-A2 for a single-disk configuration, the SSP allocates the file in the last segment (highest address) of available storage that is large enough to contain the file.

RETAIN: The RETAIN parameter classifies the file as a scratch, job, temporary, or permanent file:

Code	Meaning
S	Scratch file
J	Job file
T	Temporary file
P	Permanent file

A *scratch file* can be used only by the job step creating it and does not exist after the job step has ended.

A *job file*, after it is created, can be used by all of the subsequent job steps in a job. The job file is defined only within the job and does not exist after the job ends. If a job step is released (via the RELEASE parameter on the ATTR OCL statement) or runs an MRT program, that step cannot use job files defined by other steps in the job. Any job files created by a released job step or a job step that runs an MRT program are treated as scratch files and cannot be used by other steps in the job. Jobs using job files with the same label can run concurrently because the job files are defined only within the individual jobs.

When a file is created with RETAIN-J specified, subsequent steps in the job can refer to that file by not specifying the RETAIN parameter or by specifying RETAIN-J or RETAIN-S. If the subsequent step specifies RETAIN-J or does not specify the RETAIN parameter, the file is kept for later use by other job steps. If, however, the subsequent step specifies RETAIN-S for an existing job file, that file is scratched at the end of the step and cannot be used by other steps in the job.

If a file is created with RETAIN-J, a subsequent job step can access or create another disk file with the same label by specifying RETAIN-T or RETAIN-P. A temporary or permanent disk file cannot be processed in the same job step with a job file having the same label.

CAUTION
If a system failure occurs, the contents of a job file or a new scratch file are lost.

A *temporary file* is usually used more than once. The area containing a temporary file becomes available for another file only under one of the following conditions:

- A FILE statement containing RETAIN-S is supplied for the temporary file to identify the file as a scratch file. If the file is allocated by the program, the file will not exist after the program ends. If the file is not allocated by the program, the file will still exist as a temporary file when the program ends.

- Another file with the same label is loaded into the area occupied by the temporary file, changing only the data. The RECORDS (or BLOCKS) and LOCATION parameters must be specified and must be the same as the original file, or the DISP-OLD parameter must be specified.

- The DELETE procedure is used to delete the file.

The area containing a *permanent file* cannot be used for any other file until the DELETE procedure is used to delete the permanent file.

The combined total of permanent and temporary files that can exist at one time is limited by the maximum number of VTOC entries that was specified when the system was last reloaded. Any number of additional scratch and job files can exist. The actual number of files that can be placed on the disk depends upon the size of the files.

A disk file is classified as scratch, job, temporary, or permanent when it is created. If the RETAIN parameter is omitted from the FILE statement when the file is created, the file is assumed to be a temporary file.

The RETAIN parameter can be omitted from a FILE statement for an existing file. If an existing permanent file is referenced by a FILE statement with RETAIN-T, it remains a permanent file. If an existing temporary file is referenced by a FILE statement with RETAIN-P, it remains a temporary file. No message is issued by the SSP to reflect the preceding situations. However, a message is issued if an existing permanent file is referenced by a FILE statement with RETAIN-S. The operator can then cancel the job or continue processing. If the operator chooses to continue processing, the file remains permanent.

DATE: The DATE parameter identifies the creation date of an existing file and ensures that the proper version of a file is used. When a file is created on disk, its label and creation date are written on the disk as identification. The program (job step) date when the file is created is the date used as the creation date. More than one permanent or temporary file can be given the same label; however, the creation dates of these files must be different. To refer to such a file, specify its label and date, its label and location on disk, or its label and size if the size is unique. If neither the date nor the location is specified, the file having the latest date is used.

The date can be entered in one of three formats: month-day-year (mmddyy), day-month-year (ddmmyy), or year-month-day (yymmdd). However, the format chosen must conform to that of the current session date format. The STATUS control command can be used to determine the current session date format.

Note: If an existing file is being overlaid, DISP-OLD or the LOCATION parameter must be specified along with the DATE parameter. If the BLOCKS or RECORDS parameter is specified, the parameter value must be the same as when the file was created.

DISP: The DISP parameter specifies that the file is a new file or an old file, or that the file can be shared by other jobs running on the system. The DISP parameter is not allowed if RETAIN-J is specified. If the DISP parameter is not specified, the system determines if a file is new or old based upon whether the file is in the disk VTOC.

DISP-NEW specifies that the file is a new file. If a file already exists with the same label and creation date as the new file, an error message is displayed. If DISP-NEW is specified for a file with the same label (but different creation date) as an existing file, the LOCATION parameter or a different RECORDS (or BLOCKS) value does not have to be specified. The new file can be created using any disk access method. The file cannot be shared by any other programs until the program that created it ends.

DISP-OLD specifies that the file is an existing file. If the file does not exist, an error message is displayed. The file cannot be shared by any other programs until the program that allocated it as an old file ends. DISP-OLD allows the user to process an existing file as an output file without having to specify the RECORDS (or BLOCKS) or LOCATION parameter. When overlaying an existing file, the creation date is changed to the current program date. After the first case of overlaying an existing file, the space allocated value is always maintained in blocks no matter what the original file allocation is.

DISP-SHR specifies that the file is an existing file that can be shared by other programs running on the system. Only input, update, and add operations can be performed on the file.

A description of programming considerations for file sharing can be found in the *Concepts and Design Guide*.

Notes:

1. A file can be shared only if all FILE statements for that file specify DISP-SHR.
2. If single-program mode was specified during system configuration or during IPL, DISP-SHR is not required to allow access to active files by an inquiry program.
3. If single-program mode was not specified during system configuration or during IPL, DISP-SHR is required to allow access to active files by an inquiry program.
4. When DISP-SHR is specified for sequential-by-key processing, no sorting by keys is done when the file is allocated to the program. Therefore, if the file being processed does not have the IFILE characteristic, any records added since the file was last sorted by keys cannot be accessed.

EXTEND: The EXTEND parameter indicates that a file may be extended by the specific value whenever additional space is needed. The value is a 1- to 8-digit blocks/records value that specifies the amount of additional space needed for the extension. The value, in blocks or records, adheres to the format specified when the file was created. The extension should be large enough to contain at least one record. The file will be extended on all output operations for all file types, and on ADD operations for consecutive and indexed files. The file will be extended on an input operation for an update to a direct file if the record to input is beyond the end of file. If the file is being shared, the extendable disk file function (EDF) will be evoked only when the user of the file needs additional space and has specified a file extension value. If more than one user specifies a value for the extension, the value of the user that caused the extension will be used. It should be understood that when sharing files, if one user causes an extension to occur, all users of that file can take advantage of the additional file space, regardless of whether or not they specified the EXTEND parameter on their FILE OCL statements. If EXTEND-0 is specified, no file extension will occur.

Notes:

1. The EXTEND parameter, if used on file statements for system utilities $BICR, $COPY, and $MAINT, is ignored except for record mode $MAINT processing. The EXTEND parameter is also ignored for a sort work file.
2. In order to use the EXTEND parameter with J files, the EXTEND parameter must be specified when the file is created. All subsequent references within the job to that J file use the extend specification referenced when the file was created. If the EXTEND parameter is specified on subsequent statements, it will be ignored.

DFILE: The DFILE parameter specifies that a delete-capable file should be created. If you specify DFILE-NO or omit this parameter, the SSP does not create the file as delete-capable. The DFILE parameter is only valid for new files or existing files that are reloaded with new information. If DFILE-NO is specified for an existing delete-capable file, or if DFILE-YES is specified for an existing file that is not delete-capable, the SSP will issue a warning message. If the DFILE parameter is specified on a system not configured with extended disk data management, an error message will be issued when attempting to use the file.

Note: Deleting a record within a delete-capable file does not result in the physical compression of the file. Instead, the record is marked as a deleted record, and is inaccessible to all users of the file.

IFILE: The IFILE parameter is used to establish the IFILE characteristic for new indexed files and verify the IFILE characteristic for existing files. If IFILE-YES is specified, the file is created with the IFILE characteristic, provided it is an indexed file. If the file is not an indexed file, the IFILE parameter is ignored. Files having the IFILE characteristic allow immediate access to added records when a file is shared and being processed sequentially by key. If IFILE-NO is specified, the file is created without the IFILE characteristic. If this parameter is omitted, the file is processed according to its current state. For a new file, the default is IFILE-NO.

BYPASS: The BYPASS parameter allows adding a record to an indexed file randomly by key without checking for a duplicate record key. If BYPASS-YES is specified, disk data management does not check for a duplicate record key before adding a record to the file randomly. Use BYPASS-YES only if the program that is adding records to the file can ensure the keys of the added records are not already present in the file. The BYPASS parameter is valid only when adding records to an indexed file and is ignored for other types of files. If BYPASS-NO is specified or if this parameter is omitted, disk data management checks for a duplicate record key before adding a record randomly.

Note: In order to use the BYPASS parameter with J files, the extend parameter must be specified when the file is created. All subsequent references within the job to that J file use the BYPASS specification referenced when the file was created. If the extend parameter is specified on subsequent statements, it will be ignored.

Examples

Example A

A program is creating a disk file. Assume the following facts about the file:

- The name the program uses to refer to the file is TRANS.
- The label of the file on the disk is TRANS1.
- The file contains 225 records.
- The system chooses the disk area to contain the file.

The FILE statement that defines the file is:

```
// FILE NAME-TRANS,LABEL-TRANS1,
//      RETAIN-T,RECORDS-225
```

Example B

A program overlays the contents of an existing temporary disk file labeled IVENTORY. The file is located at block 500 and is 12 blocks long. The filename used by the program is OUTFILE. Either of the following FILE statements can be used:

```
// FILE NAME-OUTFILE,LABEL-IVENTORY,
//      BLOCKS-12,LOCATION-500,RETAIN-T

                or

// FILE NAME-OUTFILE,LABEL-IVENTORY,
//      RETAIN-T,DISP-OLD
```

Example C

A program requires that an additional 250 records of file space is needed for a file named MSTR. The additional space will not be allocated until it is actually needed as a result of the program's action.

```
// FILE NAME-MSTR,RETAIN-T,
//      DISP-OLD,EXTEND-250
```

Example D

Create a delete-capable file named MSTIN.

```
// FILE NAME-MSTIN,RETAIN-T,
//      DISP-NEW,DFILE-YES,
//      RECORDS-250
```

FILE Statement (for Diskette Files)

Function

The FILE statement supplies the SSP with information about a diskette file. The SSP uses this information to read records from and write records on the diskettes.

For more information about System/34 diskette file concepts, see the *Concepts and Design Guide.*

Note: The only programs that use FILE statements for diskette files are the following SSP utility programs, which are described in Chapter 4:

$BACK–Backup Library Utility Program
$BICR–Basic Data Exchange Utility Program
$COPY–Disk Copy/Display Utility Program
$DUPRD–Diskette Copy Utility Program
$LOADI–Reload Library Utility Program
$MAINT–Library Maintenance Utility Program
$POST–Data Exchange Utility Program
$PRST–Security File Restore Utility
$PRSV–Security File Save Utility

Placement

A FILE statement is required for each new diskette file that a program creates, and for each of the existing diskette files that a program uses. The FILE statement must follow the LOAD statement and precede the RUN statement.

Format

```
// FILE NAME–filename,UNIT–I1 [,LABEL–file label]

[,RETAIN–{retention days}] [,DATE–{mmddyy}]
         {    1      }       {ddmmyy}
                             {yymmdd}

[,PACK–vol-id] [,LOCATION–{S1  }]] [,AUTO–{NO }]]
                          {S2  }          {YES}
                          {S3  }
                          {M1.nn}
                          {M2.nn}
```

Note: The LOCATION and AUTO parameters are ignored for systems without a diskette magazine drive.

Parameters

The NAME and UNIT parameters are always required. The other parameters are required only under certain conditions.

NAME: The NAME parameter tells the SSP the name that the program uses to refer to the file. The filename can be any combination of characters (numeric, alphabetic, and special) except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. The first character of a filename must be alphabetic (A through Z, #, $, or @). The number of characters in a filename must not exceed eight.

UNIT: The UNIT parameter tells the SSP whether the file is on the disk or on diskette(s). The code for the UNIT parameter on a FILE statement for a diskette is I1. UNIT-I1 must be specified for a diskette file. If the UNIT parameter is omitted, UNIT-F1 is assumed.

LABEL: The LABEL parameter tells the SSP the label by which the file is identified on the diskette. If the file is being created, the label supplied in the LABEL parameter is used to identify the file on a diskette. If the LABEL parameter is omitted from a diskette FILE statement, the name specified by the NAME parameter is used. If the file is an existing file, a LABEL parameter is required when the name the program uses to refer to the file differs from the label that identifies the file on a diskette.

The label can be any combination of characters (numeric, alphabetic, and special) except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. The first character of a label must be alphabetic (A through Z, #, $, or @). The number of characters in a label must not exceed eight.

If more than one file on a diskette has the same label, those files must have different creation dates. In all cases except when you are using the SAVE procedure or the $COPY utility program to save all files, the SSP displays a warning message before it will create a file with the same label as an existing file on the diskette. The operator can then (1) allow the SSP to create the file with a different creation date, (2) insert an alternative diskette and then allow the SSP to create the file on that diskette, or (3) cancel the job. If you are using the SAVE procedure or the $COPY utility program to save *all* files, the creation date of each disk file becomes the creation date of the corresponding diskette file. Therefore, if more than one disk file exists with the same label, the diskette(s) will also contain those files with the same label and different creation dates. The SSP does not display a warning message in this case.

If more than one diskette file has the label specified on a FILE statement and you do not specify the DATE parameter, the program processes the *first file* in the diskette VTOC with the specified label. Unlike the disk, there is no search for the file with the most recent date.

RETAIN: The RETAIN parameter specifies the number of days a file is to be retained. The RETAIN parameter is used to compute an expiration date. Whenever the RETAIN parameter is specified for a file, the SSP determines the expiration date of the file by adding the program (job step) date to the number of days specified by the RETAIN parameter. If the program date is invalid, an error message is issued. The operator can cancel the job or continue processing. If the operator decides to continue, the expiration date is set equal to the creation date. The RETAIN parameter can be any decimal number from 0 through 999. If the RETAIN parameter is not specified when a new file is created, 1 is assumed. If any number up to 998 is specified, the file is retained for the specified number of days. If 999 is specified, the file is considered permanent and can be deleted only by the DELETE procedure.

When creating a diskette file, the SSP writes the creation date and the calculated expiration date of the file in international format (yymmdd). If an existing nonpermanent diskette file is referenced by a FILE statement with a RETAIN parameter, the expiration date of the file is changed to the date determined by the RETAIN parameter.

If an existing permanent diskette file is referenced by a FILE statement with a nonpermanent RETAIN parameter, an error message is issued. The operator can then cancel the job or continue processing. If the operator decides to continue processing after the message is displayed, the file remains a permanent file.

Whenever the SSP is creating a file on a diskette or adding to an existing file on a diskette, all other files on the diskette whose expiration dates are the same as or earlier than the program date and all files with blank (hex 40) expiration dates are deleted automatically. If a file being added to has expired, it is not deleted.

When the expiration dates are checked for having been met, each expiration date is checked for the international format (yymmdd) by comparing the first two characters to 70. If the first two characters are less than 70, it is assumed that the expiration date is not in the international format and is in the same format as the session date. (International format dates earlier than 1970 and expiration dates not in the international or session date formats, will be misinterpreted by the SSP. This could cause the SSP to delete unexpired files or to retain expired files.)

When a new file is created on a diskette, the new file starts at the first available sector beyond the last unexpired file. The space that was occupied by a deleted file is not reused if any unexpired files follow it.

DATE: The DATE parameter specifies the creation date of an existing file and ensures that the proper version of a file is processed.

If you specify both a label and a creation date on the FILE statement, the SSP displays an error message if it cannot find a file with that label and date on the inserted diskette. The operator can then insert an alternative diskette and the SSP will look for the specified file on that diskette.

Note: When a file is created on diskette, its label, expiration date, and creation date (program date) are written on the diskette as identification. When a diskette file is created on System/34, the SSP writes both the creation date and the expiration date in the international format (yymmdd). The SSP converts the program date to the international format before writing it on the diskette. If the DATE parameter is specified for an existing diskette file, the SSP converts the specified date to international format before accessing the file. To ensure correct processing, files created by other systems should be created so that the creation date and the expiration date are written in the international format.

For information about the program (job step) date, see the description of the DATE statement earlier in this chapter.

PACK: The PACK parameter is required when a program is creating a file or adding to a file on a diskette. The PACK parameter specifies the volume ID of the diskette associated with the FILE statement. The volume ID is put on the diskette by the INIT procedure and can be any combination of 6 or less alphameric characters.

The volume ID specified by the PACK parameter is compared with the volume ID of the inserted diskette; if they are not the same, an error message is displayed. The operator can then continue processing (ignore volume ID), insert the correct diskette, or cancel the job.

If the PACK parameter is not supplied on the diskette FILE statement for an output file or for an addition to a file, an error message is displayed. The operator must then cancel the job.

The PACK parameter is not required for a diskette input file; however, the operator should ensure that the proper diskette is inserted.

LOCATION: Specifies a diskette location.

S1, S2, or S3 identifies an individual diskette slot. If the file is a multivolume file, the specified slot must contain the first volume of the file. If LOCATION is not specified, LOCATION-S1 is assumed.

M1.nn or M2.nn identifies a magazine location. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location within the magazine. Specifying LOCATION-M1 or LOCATION-M2 is the same as specifying LOCATION-M1.01 or LOCATION-M2.01, respectively.

AUTO: Controls multivolume processing.

AUTO-NO specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, the program uses only the specified slot. After a volume has been processed, the SSP displays a message; the operator must insert the next volume in the specified slot.

- If M1.nn or M2.nn is specified in the LOCATION parameter, the program uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more volumes remain to be processed, the SSP displays a message; the operator must insert the next magazine in the same magazine slot. Processing resumes at the first location in the magazine.

AUTO-YES specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, the program can use all three individual slots. Processing begins with the diskette in the slot specified on the LOCATION parameter. After the diskette in slot S3 has been processed, the SSP displays a message; the operator must then insert the next volumes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is specified in the LOCATION parameter, the program can use both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more volumes remain to be processed, the SSP displays a message; the operator must then insert the next magazine(s). Processing resumes at location M1.01 and continues through M2.10.

For output to diskettes, the first diskette to receive output must be in the specified location. If the diskette in that location is not capable of receiving the output, the system will not search the following slots for a valid diskette. Instead, an error message will be displayed and the operator will have to insert a diskette that is capable of receiving the output into the specified slot.

If the AUTO parameter is not specified, AUTO-YES is assumed.

Example

Assume the following facts about a file to be created on a diskette:

- The program that creates the file refers to the file as COPYO.
- The label of the file once it is on a diskette is to be TRANS1.
- The file is to be saved for use at the end of the month, but the file can be deleted the first day of the next month. Seven days remain in the current month.
- The file is on the diskette identified by 666666.

The FILE statement for the file is:

```
// FILE NAME-COPYO,UNIT-I1,LABEL-TRANS1,
//      RETAIN-8,PACK-666666
```

FORMS Statement

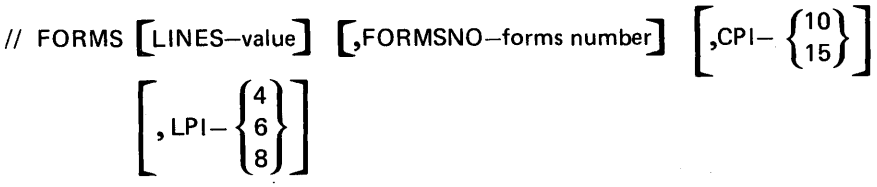
Function

The FORMS statement specifies the number of lines to be printed per page, the forms number to be used, the horizontal print density to be used, and the vertical print density to be used during a display station session. A job placed on the input job queue uses initially the values that were in effect when the job was placed on the queue. However, if a procedure executing from the job queue contains a FORMS statement, the FORMS statement in the procedure will be used.

Placement

The FORMS statement can appear anywhere among the OCL statements.

Format



Note: If the FORMS statement is used, at least one parameter must be specified.

Parameters

LINES: The LINES parameter specifies the number of print lines per page. The maximum number of lines that can be specified per page is 112. The LINES parameter remains in effect until another FORMS statement is used, until the LINES procedure with the LINES parameter is used, or until the display station session is ended. If the LINES parameter is not specified and the number of lines per page was not previously set during the session, the system uses the value specified for the display station during system configuration or by the SET procedure or the $SETCF utility program. If a line counter specification is used in an RPG II program, that specification remains in effect only for the duration of that program.

For SSP utility programs, the printer skips (overflows) to a new page when six less than the number of lines specified are printed. For example, if LINES-84 is specified, the printer skips to a new page after printing line 78. If LINES-13 is specified, one line is printed per page. When 12 or less lines are specified, printing is on every line (no overflow).

For print operations from user programs, the SSP indicates an overflow condition when six less than the number of lines specified are printed, unless this value is overridden.

Note: RPG II and COBOL programs can specify overflow rules that override the values specified in a FORMS statement. Assembler language programs can specify overflow rules via the $DTFP macroinstruction. For further information, see the *RPG II Reference Manual*, the *COBOL Reference Manual*, and the *Basic Assembler and Macro Processor Reference Manual.*

FORMSNO: The FORMSNO parameter specifies the forms number of the printer forms to be used for printed output from the display station session. (Each type of form should have a unique, user-assigned forms number.) The forms number can be any combination of up to 4 characters except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. If the FORMSNO parameter is specified, the SSP prompts the system operator to mount the forms with the specified forms number if the specified forms are not already mounted.

CPI: The CPI (characters per inch) parameter specifies the horizontal print density to use for printed output from the display station session. The values that can be specified are 10 or 15. If CPI-15 is used, the output should be printed on a 5225 (Models 1 through 4) Printer or a 5224 (Models 1 and 2) Printer. If CPI-15 is attempted on another printer, an error message will appear.

The CPI parameter remains in effect until changed by another FORMS statement, a PRINTER statement, or the LINES procedure; or until the display station session is ended. If the CPI parameter is not specified and the characters per inch was not previously set during the session, the system uses a default value of 10.

LPI: The LPI (lines per inch) parameter specifies the vertical print density to use for printed output from the display station session. The values that can be specified are 4, 6, and 8. If the output is printed on a printer other than the 5225 Printer or the 5224 Printer, the LPI parameter is ignored.

The LPI parameter remains in effect until changed by another FORMS statement or by the LINES procedure, or until the display station session is ended. The PRINTER statement can be used to change the LPI value for a particular print job, but it stays in effect only for that print job. If the LPI parameter is not specified and the lines per inch was not previously set during the session, the system uses the value that was set during work station configuration (either 4, 7, or 8). If no LPI value was specified during work station configuration, the system uses an LPI value of 6.

Note: For the 5225 Printer Models 11 and 12 and the 5224 Printer Model 12, an LPI value of 8 will cause characters to vertically overlap from one line to another.

Example

The following statement specifies that the forms length for the session is 50 lines per page and that form ABC3 is to be used:

IMAGE Statement (for IBM 5211 and 3262 Printers Only)

Function

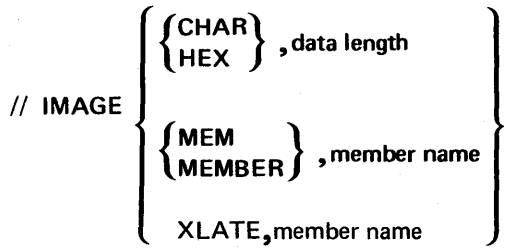
The IMAGE statement specifies the print belt image to be used for all output to the 5211/3262 printers from the requesting display station. For each display station, a special area on disk called the print belt image area contains the characters matching those on the print belt to be used when output for the display station is printed. That image can be changed any time by the IMAGE statement. When output from the display station is directed to a 5211/3262 Printer, the SSP checks that the print belt matches the image in the print belt image area. If the belt does not match, the SSP displays a message instructing the system operator to change the belt.

For users with translation capability, the IMAGE statement permits the specification of a translation table for each display station or job on the input job queue. When a job initiates from a display station or the input job queue, it will execute with the system default translation table unless one is specified via an IMAGE OCL statement. If the XLATE parameter is specified on a system, without translation capability, the translate table data is processed, but not sent to the printer.

Placement

The IMAGE statement can appear anywhere among the OCL statements.

Format



Parameters

The IMAGE statement tells the SSP to:

- Read the new print belt characters from the system input device (either from the keyboard or from a procedure member being executed)
- Read the new print belt characters from a source member in the system library
- Initialize a new translation table for a particular display station or job on the input job queue

Characters from the System Input Device

To indicate that the new print belt characters are to be entered from the keyboard or from the procedure member if a procedure contains the IMAGE statement, use the following parameters:

CHAR or *HEX:* CHAR indicates that the characters are in alphameric form. HEX indicates that the characters are in hexadecimal form. (See Appendix F for the hexadecimal form of characters.)

Note: If the print belt contains characters that cannot be entered from the keyboard, HEX must be specified.

data length: The data length parameter must be used with CHAR and HEX. This parameter specifies the number of characters to be entered from the keyboard. (If HEX is specified in the first parameter position, each character on the print belt requires 2 hexadecimal characters from the keyboard.) After the IMAGE statement is entered, the operator is prompted to enter the characters. For both the 3262 and the 5211 printers, the data length parameter must not exceed 384 when the characters are hexadecimal, or 192 when the characters are alphameric.

Following are the rules for entering the new characters from the keyboard:

- The characters must begin in position 1.
- Consecutive positions must be used and characters must be entered in the sequence in which they appear on the new print belt.

  The sequence for entering characters on the 48-character print belt is:

  1234567890#@/STUVWXYZ&,%JKLMNOPQR-$*ABCDEFGHI+.'

  The sequence for entering characters on the 64-character print belt is:

  1234567890#@/STUVWXYZ&,%JKLMNOPQR-$*ABCDEFGHI+.'¢<(!);¬\ _ >?:="`

  Note: The question mark (?) has a special meaning in procedures; therefore, if you use a procedure to enter the IMAGE statement and a question mark is one of the characters on the print belt, you must use the HEX form for all characters.

- All positions in a record must contain nonblank characters before characters can be continued on the succeeding record (beginning in position 1 of the new record).

Characters from a Source Member in the System Library

To indicate that the new print belt characters are to be read from a source member in the system library, use the following parameters:

MEM or *MEMBER:* Indicates that the new print belt characters are in a system library source member.

member name: Specifies the name of the source member that contains the new print belt characters. The first record in the source member must be an IMAGE statement that specifies the format as alphameric (CHAR) or hexadecimal (HEX) and specifies the number of characters in the source member. The new characters begin in the first position of the second record in the source member.

Note: For a 3262 Printer, the data length parameter in the source member's IMAGE statement must not exceed 576 when the characters are hexadecimal, or 288 when the characters are alphameric. For a 5211 Printer, the data length parameter must not exceed 384 when the characters are hexadecimal, or 192 when the characters are alphameric.

The SSP includes the following print belt members:

Print Belt Member Name	Associated Printer
BELT48	5211 or 3262
BELT64	5211
BELT96	5211 or 3262
BELT188	5211
BELT48HN (FORTRAN)	5211 or 3262
BELT64B	3262
BELT64C	3262
BELT188B	3262

Specifying a Translation Table

To initialize a new translation table from a display station or the input job queue, use the following parameters:

XLATE: Indicates that a translation table will be replaced.

member name: Specifies the name of the source member in the system library that contains a replacement version of a translation table.

In order to specify a translation table, the format of the first record within the source member must be in the form *pp,nnn*. *pp* is a 2-byte character representation of 1 hex byte specifying the position of the first character to be replaced in the translation table. *pp* must be in the range of hex 40 to hex FE. *nn* is a 1- to 3-character decimal value that specifies the number of hex character representations being supplied as translation table data. The allowable range for this value is 2 through 384.

The second and subsequent source records must contain 2-byte character representations of each hex byte to be replaced in the translation table. The number of hex representations must be equal to the value specified in *nnn*, and each representation must have a value greater than or equal to hex 40 and less than or equal to hex FE.

The SSP includes the following translation tables:

Translation Table Name	Translation Fold
#188E96	192- to 96-character set fold
#188E64	192- to 64-character set fold
#188E48	192- to 48-character set fold
#96E64	96- to 64-character set fold
#96E48	96- to 48-character set fold

See Appendix L for the specific translation of each character in the translation tables listed above.

Examples

The IMAGE statements in Examples A and B tell the system that the new characters are to be entered from the keyboard. The IMAGE statement in Example C tells the system that the new characters are to be read from a system library source member.

Example A

The new characters entered from the keyboard are in hexadecimal form, and 96 hexadecimal characters are entered (48-character print belt).

```
// IMAGE HEX,96
```

Note: If this IMAGE statement were in a procedure member, the 96 hexadecimal characters would begin in position 1 of the record following the IMAGE statement.

Example B

The new characters entered from the keyboard are alphameric and 48 characters are entered.

```
// IMAGE CHAR,48
```

Example C

The new characters are in a system library source member called BELT48.

```
// IMAGE MEM,BELT48
```

For the 48-character print belt, BELT48 would contain:

```
// IMAGE CHAR,48
1234567890#@/STUVWXYZ¢,%JKLMNOPQR-$*ABCDEFGHI+.'
```

Example D

Initialize a new translation table using source member ALPHA.

```
// IMAGE XLATE,ALPHA
```

The source member ALPHA would contain the following records:

```
C1,6
E7E8E9
```

In the above example, C1 indicates the first character to be replaced, and 6 specifies the number of hex representations being supplied. As a result, the characters A, B, and C are changed to X, Y, and Z.

INCLUDE Statement

Function

The INCLUDE statement identifies a procedure member containing OCL and utility control statements to be merged into the job stream. If the procedure is not an MRT (multiple requestor terminal) procedure or if PDATA-YES was not specified when the procedure member was created, the INCLUDE statement can pass *parameters* to the procedure.

If the procedure is an MRT procedure or if PDATA-YES was specified when the procedure member was created, the INCLUDE statement can pass only *data* to the program.

Note: PDATA-YES appears only when you use the $MAINT utility program. If you use SEU, a prompt is issued instead.

For general information and programming considerations about MRT programs and procedures, see the *Concepts and Design Guide.*

For information about creating a procedure member, see the discussion of reader-to-library operations in the description of the $MAINT utility program in Chapter 4. You can also use the source entry utility to create a procedure member. For information about the source entry utility, see the *IBM System/34 Source Entry Utility Reference Manual.*

Placement

The INCLUDE statement can be placed anywhere within a set of OCL statements.

Formats

// INCLUDE procedure name [parameter(s) / data]

or

// procedure name [parameter(s) / data]

or

procedure name [parameter(s) / data]

The first form of the INCLUDE statement should be used if the procedure name is:

- The same as an OCL statement identifier
- A control command name (see Chapter 3)
- CANCEL, IF, IFF, IFT, RESET, or RETURN

For example, if the procedure name is LOAD, the following format is correct:

```
// INCLUDE LOAD parameter(s)
```

The form of the INCLUDE statement that does not contain the // or the statement identifier, INCLUDE, is called a procedure command.

Parameters

procedure name: Specifies the name of the procedure member to be merged into the job stream.

parameters: Parameters are valid only if the procedure is not an MRT procedure and if PDATA-YES was not specified when the procedure member was created. The parameters are positional parameters that might or might not be required, depending on the procedure they are passed to. (Positional parameters are parameters that must appear in the same position, relative to other positional parameters, in the same statement.)

A parameter can be any combination of characters except question marks (?), commas (,), single quotes ('), slashes (/), hyphens (-), or blanks. The number of characters in a parameter must not exceed eight. A maximum of 11 parameters, separated by commas, can be passed with an INCLUDE statement.

Note: For the ideographic version of the SSP, parameters cannot contain ideographic characters.

data: The data starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. The data is passed to the program on its first input operation from the requesting display station.

Examples

Example A

Procedure FILE1 is included between the LOAD statement and the RUN statement. The INCLUDE statement passes the label of a file, WEEKLY, to the procedure. FILE1 generates the FILE statements necessary to execute the PAYROLL program.

```
// LOAD PAYROLL
FILE1 WEEKLY
// RUN
```

Assuming that PAYROLL requires only two FILE statements and that procedure FILE1 generates those statements, the effect of the preceding three OCL statements would be the following sequence of OCL statements processed by the SSP:

```
// LOAD PAYROLL
// FILE LABEL-WEEKLY,NAME-OUTFILE,RETAIN-T,
//      DISP-NEW
// FILE LABEL-TRANSACT,NAME-INFILE,DISP-OLD
// RUN
```

Example B

In this example, JOE and SAM are two parameters that are interpreted by the PAYROLL procedure. Parameter 2 is omitted.

```
// INCLUDE PAYROLL JOE,,SAM
```

Example C

In this example, MRTPROC is an MRT procedure that causes an MRT program to be run. The number 126 is data passed to the MRT program on its first input operation from the requesting display station.

```
// INCLUDE MRTPROC 126
```

JOBQ Statement

Function

The JOBQ statement places a job on the input job queue.

Note: For the ideographic version of the SSP, jobs run from the input job queue print alphameric system headings.

Placement

The JOBQ statement can appear anywhere among the OCL statements. For information about jobs run from the input job queue, see the *Concepts and Design Guide*.

Format

```
// JOBQ [jobq prty 3 ,] [library name #LIBRARY] , procname [,parm1 ,parm2 . . .]
```

Parameters

jobq prty: Specifies the job queue priority for the job, that is, the job's order of execution from the input job queue. The job queue priority can be any decimal number from 1 through 5. The system runs jobs in order of decreasing job queue priority. For example, all jobs with a job queue priority of 5 are run before any other jobs in the job queue. Jobs with the same job queue priority are run in the order they were placed in the job queue. Jobs with a job queue priority of 1 jobs are the last jobs run by the system. If the job queue priority parameter is not specified, a job queue priority of 3 is assigned to the job, and the comma shown in this parameter should not be specified.

library name: Specifies the active user library for the job. The system searches the active user library and then the system library for the procedures and load members executed in this job and for the message members and display screen formats used in this job. If library name is not specified, the system library (#LIBRARY) is assumed.

procname: The name of the procedure (a procedure member) that defines the job to be placed on the input job queue.

parm1,parm2 . . .: Parameters required by the procedure. A maximum of 11 parameters can be specified.

Example

Place the PAYROLL procedure, which is located in library PAYLIB, on the input job queue with a job queue priority of 4:

```
// JOBQ 4,PAYLIB,PAYROLL
```

LIBRARY Statement

Function

The LIBRARY statement specifies the name of the active user library for the display station session or for the duration of a procedure. The active user library is the library that the SSP searches first for load programs specified on LOAD OCL statements, procedures, menus, message members, and display screen formats. If the SSP does not find the member in the active user library, the SSP then searches the system library. A user library remains active until the SSP processes another LIBRARY statement from the display station or until the display station session ends. If the LIBRARY statement is used in a procedure, the active user library is changed only for the execution of that procedure, unless SESSION-YES is specified. Upon completion of the procedure, the user library that was active when the procedure began is again active.

Notes:

1. Once a user library is specified as active during the sign-on procedure or by a LIBRARY OCL statement entered from the keyboard, that library remains allocated to the display station at least until a different active library or NAME-0 is specified. If a different active library or NAME-0 is specified while the display station is still using one or more members from the previous active library (for example, using a menu from that library), the previous active library remains allocated to the display station until the display station is no longer using members from that library. Consequently, certain functions, such as the CONDENSE procedure, cannot be performed on the library even though it is no longer the active library at the display station.
2. When a program within a procedure is interrupted, the active user library *for the session* is active during the inquiry request. The user library that was active for the interrupted procedure will again be active when the procedure is resumed. For example, a procedure contains the following statement:

   ```
   // LIBRARY NAME-JOBLIB
   ```

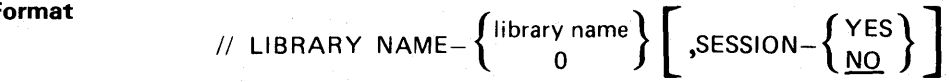
   If a program within the procedure is interrupted, the active library for the session (not JOBLIB) will be active during the inquiry request. For information about interrupting a program, see the *Concepts and Design Guide*.

3. If a LIBRARY statement is entered from a MENU display, one or more of the items on the displayed menu might no longer be valid because the procedures corresponding to those items do not exist in the new active library.

Placement

The LIBRARY statement can appear anywhere among the OCL statements except between a LOAD statement and a RUN statement.

Format



Parameters

NAME: The NAME parameter specifies the name of the user library that will be active. If NAME-0 is specified, no user library is active; only the system library is searched by the system.

SESSION: SESSION-YES causes the active session library to be changed to the active user library specified in the NAME parameter (session library is the user library in effect at the keyboard).

Note: SESSION-YES does not change the designation of the current library in a procedure. SESSION-YES changes only the library in effect when return to the keyboard occurs.

Example

The following statement specifies that a user library called ULIB1 is the active library:

LOAD Statement

Function

The LOAD statement identifies a program to be executed. The LOAD statement is the first statement in a set of statements that define a job step.

Placement

Two LOAD statements cannot be entered without an intervening RUN statement.

Format

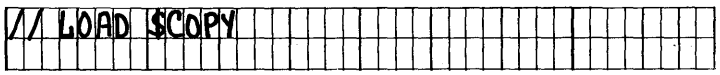
// LOAD program name

Parameter

program name: The name of the program to be loaded.

Example

In the following LOAD statement, $COPY identifies the disk copy/display utility program:



LOCAL Statement

Function

The LOCAL statement is used to modify a specified area in the display station local data area.

A display station local data area exists for each command display station. A local data area also exists for each executing MRT procedure. Each display station local data area is a 256-byte area on disk that can be used to pass information between jobs and job steps submitted during a display station session. (For information on using data from the display station local data area to modify a procedure, refer to the description of the ?L'offset,lng'? expression in Chapter 5.) The SSP sets the display station local data area to blanks at the beginning of a session. The display station local data area is available for use by all jobs run during the session. However, a job placed on the input job queue or a released job step uses a copy of the display station local data area as it was when the job was placed on the queue or when the job was released.

If the inquiry function is used, the contents of the display station local data area are saved and then restored when the inquiry processing is complete.

CAUTION
The RPG II procedures (AUTO, RPG, and RPGR) use bytes 201 through 256 of the display station local data area, the work station utility generation procedure (WSU) uses bytes 238 through 256 of the local data area, the screen design aid procedure (SDA) uses bytes 1 through 104 of the local data area, the HELP procedure uses bytes 249 through 256 of the local data area, the ICFVERIFY procedure (SSP-ICF) uses bytes 1 through 4 of the local data area, the system measurement facility procedures (SMFSTART and SMFPRINT) use bytes 220 through 256 of the local data area, and the 3270 emulation program product uses bytes 230 through 256 of the local data area. Therefore, any user data in those bytes is destroyed when one of those procedures is run.

Placement

The LOCAL statement can appear anywhere among the OCL statements.

Format

```
// LOCAL OFFSET-nnn,DATA-'data'
```

Parameters

OFFSET: The OFFSET parameter specifies the first position in the display station local data area to be changed. The OFFSET parameter value can be any decimal number from 1 through 256.

DATA: The DATA parameter specifies the data that is to be placed in the display station local data area. Entries for this parameter must be enclosed in single quotes. If the data contains imbedded quotes (such as the apostrophe in o'clock), then enter the imbedded quotes as double quotes. The data that is to be placed in the display station local data area should not include commas (,), question marks (?), slashes (/), or hyphens (-) because those characters have special meaning within procedures. The number of characters is limited by the maximum length of the OCL statement.

Note: For the ideographic version of the SSP, the data can contain ideographic characters.

Examples

Example A

The following statement places the word PAYROLL in the display station local data area in positions 18 through 24:

```
// LOCAL OFFSET=18,DATA='PAYROLL'
```

Example B

The following statement appears within a procedure and places the value of the first procedure parameter into the local data area, beginning at position 12:

```
// LOCAL OFFSET=12,DATA='&1'
```

For further information about substitution expressions within a procedure, see Chapter 5.

LOG Statement

Function

If single-program mode without spooling was specified during system configuration or during IPL, the LOG statement tells the SSP where to display messages and OCL statements and whether to skip to line 1 of the next form at the end of the job. If a LOG statement is not entered after the IPL, the system displays messages and OCL statements only on the display screen.

If single-program mode was not specified during system configuration or if spooling was specified, the LOG statement is ignored and the SSP displays messages and OCL statements on the display screen at the requesting display station unless the program being run was an MRT (multiple requestor terminal) program, was released (via the RELEASE parameter on the ATTR OCL statement), or was run from the input job queue. The SSP displays log output for these programs at the system console.

Placement

The LOG statement can appear anywhere among the OCL statements.

Format

```
// LOG {CRT    } [,EJECT  ]
       {PRINTER}  [,NOEJECT]
```

Parameters

CRT: Use only the display screen.

PRINTER: Use the system printer and the display screen.

EJECT: Skip to the first line of the next page at the end of the job step. EJECT is assumed if neither EJECT nor NOEJECT is specified.

NOEJECT: Do not skip to the first line of the next page at the end of the job step.

Example

In single-program mode, messages and OCL statements are to be displayed on both the display screen and the system printer, and EJECT is assumed.

```
// LOG PRINTER
```

MEMBER Statement

Function

The MEMBER statement specifies the names of the active message members. The system retrieves messages to be displayed from the active message members. The specified members remain active until the system processes another MEMBER statement from the display station or until the display station session is ended. If the MEMBER statement is used in a procedure, the active members are changed only during execution of the procedure. Upon completion of the procedure, the message members that were active when the procedure began are again active.

Four types of message members are specified by the MEMBER statement: PROGRAM1, PROGRAM2, USER1, and USER2. IBM program product (except SSP) first-level and second-level messages are retrieved from the PROGRAM1 and PROGRAM2 members, respectively. Customer-written programs should not use the program product first-level and second-level message members. Customer-written program first-level and second-level messages are retrieved from the USER1 and USER2 members, respectively. First-level messages are a maximum of 75 characters in length. Second-level messages are a maximum of 225 characters in length.

For information about creating message members, see the description of the $MGBLD utility program in Chapter 4.

Placement

The MEMBER statement can appear anywhere among the OCL statements.

Format

$$\text{// MEMBER } \left[\text{PROGRAM1}-\left\{\begin{matrix}\text{name}\\ \underline{0}\end{matrix}\right\}\right] \left[,\text{PROGRAM2}-\left\{\begin{matrix}\text{name}\\ \underline{0}\end{matrix}\right\}\right]$$

$$\left[,\text{USER1}-\left\{\begin{matrix}\text{name}\\ \underline{0}\end{matrix}\right\}\right] \left[,\text{USER2}-\left\{\begin{matrix}\text{name}\\ \underline{0}\end{matrix}\right\}\right]$$

Parameters

PROGRAM1: The name of the load member used for IBM program product (except SSP) first-level messages. Each IBM program product has its own set of names for related message load members.

If 0 (zero) is specified for name, there is no active program product first-level message member.

PROGRAM2: The name of the load member used for IBM program product (except SSP) second-level messages. Each IBM program product has its own set of names for related message load members.

If 0 (zero) is specified for name, there is no active program product second-level message member.

USER1: The name of the load member used for first-level and OCL statement messages for a program supplied by the user. The USER1 message member is also used in conjunction with the // * statement, the // ** statement, and procedure substitution from message members.

If 0 (zero) is specified for name, there is no active user first-level message member.

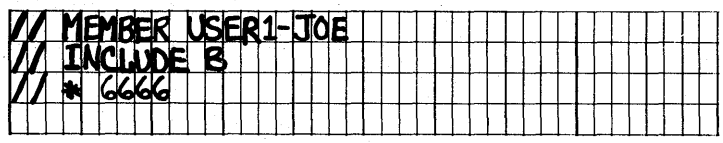
USER2: The name of the load member used for second-level messages for a program supplied by the user.

If 0 (zero) is specified for name, there is no active user second-level message member.

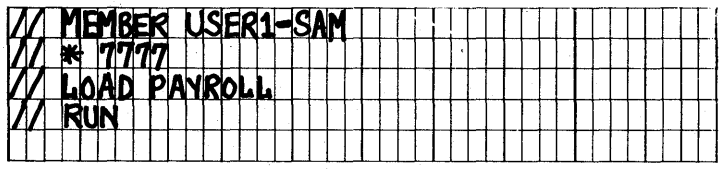
Example

In this example, a MEMBER statement is used with an included procedure.

Procedure A:



Procedure B:



In Procedure A and Procedure B, the // * statements cause messages to be displayed. (The // * statement is described later in this chapter.) In Procedure A, the message with a message identification code of 6666 comes from the message load member named JOE. In Procedure B, the message with a message identification code of 7777 comes from the message load member named SAM.

MENU Statement

Function

The MENU statement causes a specified menu to be displayed at end of job when the job containing the MENU statement ends.

Placement

The MENU statement can appear anywhere among the OCL statements.

Format

// MENU menuname [,library]

Parameters

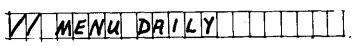
menuname: The name of the menu to be displayed at the end of job; the name can contain from 1 to 6 alphameric characters.

library: The name of the library containing the MENU. If a library name is not specified, the system searches the active user library if one exists and then searches the system library (#LIBRARY) for the menu.

Note: For the ideographic version of the SSP, the menu can contain ideographic characters; however, the SSP will not display the menu at a nonideographic display station. If an attempt is made to display an ideographic menu at a nonideographic display station, the SSP issues an error message.

Example

In this example, the MENU statement causes a menu called DAILY, which resides in the active user library, to be displayed when the job containing the MENU statement ends.



MSG Statement

Function

The MSG statement sends a message to the system console, to a selected display station, or to a selected display station operator.

Placement

The MSG statement can appear anywhere among the OCL statements.

Format

```
// MSG [ws-id | user-id] ,message text
```

Parameters

ws-id: The 2-character work station ID of the display station to which the message is sent. The STATUS WORKSTN control command can be used to determine the work station IDs. If the first parameter is not specified, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console. If the work station ID of the system console is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console.

user-id: The 1- to 8-character user ID that identifies the operator to whom the message is sent. Each display station operator enters a user ID on the SIGN ON display when the session is started. The STATUS WORKSTN control command can be used to determine user IDs. If the first parameter is not specified, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console. If the user ID of the system operator is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console.

message text: Up to 60 alphameric or special characters.

Note: For the ideographic version of the SSP, the message text can contain ideographic characters; however, the SSP does not send the message to a nonideographic display station. If an attempt is made to send an ideographic message to a nonideographic display station, the SSP issues an error message.

Example

In this example, job A is run from the input job queue, and the operator must know when job A ends. The following MSG statement (the last statement in job A) informs the operator at the requesting display station when job A ends.

```
// MSG ?WS?,JOB A IS COMPLETE
```

Note: ?WS? is an expression that causes the work station ID of the display station that placed job A on the input job queue to be substituted into the MSG statement. For information about substitution expressions, see *Chapter 5. Writing and Using Procedures.*

OFF Statement

Function

The OFF statement terminates a display station session. After the session ends, the SIGN ON display appears on the display screen at the display station. The OFF statement is not processed unless all jobs using the display screen have ended. Messages on the message queue for the display station are removed from the queue at this time.

The OFF statement cannot be used in an MRT procedure or while operating in inquiry mode. It also cannot be used with an interactive communication feature session, or in a job running from the job queue.

Placement

The OFF statement cannot be placed between the LOAD and RUN statements.

Format

```
// OFF [DROP]
       [HOLD]
```

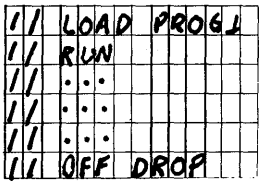
Parameters

DROP: The DROP parameter drops the communication line connection for a remote work station on a switched line. The Sign On display will not appear. However, a message indicating that the sign off was successful will be displayed. If a parameter is not entered, DROP is assumed.

HOLD: The HOLD parameter holds the communication line connection and displays the SIGN ON display for a remote work station on a switched line.

Example

In the following example, a procedure runs the user program PROG1 from a remote work station, and at the end of job time, drops the communication line and signs the user off.



PAUSE Statement

Function

The PAUSE statement causes a job to suspend processing and causes a message to be displayed at the requesting display station. The display station operator can then restart the job by selecting option 0. The system then continues reading the OCL statements that follow the PAUSE statement.

Note: Although the message length can be 109 characters, only 68 characters can be displayed. In order to receive the total message, you must print the history file.

Placement

The PAUSE statement can appear anywhere among the OCL statements.

Format

```
// PAUSE ['message']
```

Parameters

'message': The 'message' parameter specifies the message to be displayed when the job suspends processing. The message must be contained within single quotes ('). If the message text contains imbedded quotes (such as the apostrophe in o'clock), then enter the imbedded quotes as double quotes ('').

Note: For the ideographic version of the SSP, the message text can contain ideographic characters; however, the SSP does not send the message to a nonideographic display station. If an attempt is made to run a procedure that contains an ideographic message following a PAUSE statement at a nonideographic display station, the ideographic data is replaced by periods.

Example

In this example, the PAUSE statement informs the display station operator that the third step in the job is complete. The job does not continue until the operator selects option 0.

PRINTER Statement

Function

The PRINTER statement controls printer output for a printer file created by a job step. The printer file is identified by the NAME parameter on the PRINTER statement. The PRINTER statement specifies the following for a printer file:

- The filename used by the program to refer to the printer
- The number of lines per page
- The forms number to be used
- Whether or not printer output should be spooled
- The number of copies of spooled output to be printed
- Whether or not printing of spooled output can begin before the job step is complete
- Whether or not the operator aligns the forms before printing begins
- The priority of the spooled printed output
- The printer to be used for the printed output
- Whether or not printed output should remain in the spool file after being printed
- Whether or not the printing is to be done on a printer that can print ideographic characters
- Whether or not the extended character task is to be called to process any extended characters to be printed (ideographic only)
- The horizontal print density (characters per inch or pitch)
- The vertical print density (lines per inch)

Placement

The PRINTER statement can appear only between a LOAD statement and the associated RUN statement. A PRINTER statement may be specified for each printer file used by the job step. If a PRINTER statement is not used, the SSP uses the PRINTER statement default values when printing the output.

Format

```
// PRINTER NAME-{filename   } [,LINES-value]
                {$SYSLIST}

[,FORMSNO-forms number] [,SPOOL-{YES}]
                                {NO }

[,COPIES-{number}] [,DEFER-{YES}]
         {  1   }          {NO }

[,ALIGN-{YES}] [,PRIORITY-{number}]
        {NO }             {  1   }

[,DEVICE-{SYSTEM}] [,HOLD-{YES}]
         {ws-id }          {NO }

[,TYPE-{IGC}] [,EXTN-{ON}] [,CPI-{10}]
       {ANY}         {OFF}        {15}

[,LPI-{4}]
      {6}
      {8}
```

Parameters

NAME: The NAME parameter specifies the filename that the program uses to refer to the printer file. The filename can be any combination of characters (numeric, alphabetic, and special) except commas (,), periods (.), single quotes ('), blanks, question marks (?), slashes (/), and hyphens (-). The first character of a filename must be alphabetic (A through Z, #, $, or @). The number of characters in a filename cannot exceed eight.

If $SYSLIST is specified as the filename and the system list device is a printer, the PRINTER statement controls system list output. For information about system list output, see the explanation of the SYSLIST statement later in this chapter.

Notes:

1. If the NAME parameter does not match a file name specified in the program, the SSP will use the default values of the PRINTER statement when printing the output.
2. For the ideographic version of the SSP, if NAME-$SYSLIST is specified and the system list device is a printer, the EXTN parameter on the // PRINTER statement overrides the value specified on the // SYSLIST statement.

LINES: The LINES parameter specifies the number of print lines per page. The maximum number of lines that can be specified per page is 112. If LINES is not specified, the number of lines set previously by a FORMS statement or by the LINES procedure is used. If the number of lines per page is specified in a program, the program's value is used. If the number of lines per page was not set during the session, the value specified for the display station during system configuration or assigned by the SET procedure or the $SETCF utility program is used.

For system list output, the printer skips (overflows) to a new page when six less than the number of lines specified are printed. For example, if LINES-84 is specified, the printer skips to a new page after printing line 78. If LINES-13 is specified, one line is printed per page. When 12 or less lines are specified, printing is on every line (no overflow).

For print operations from user programs, the SSP indicates an overflow condition when six less than the number of lines specified are printed, unless this value is overridden.

Note: RPG II programs can specify overflow rules that override the values specified on the PRINTER statement. Assembler language programs and COBOL programs can specify overflow rules via the $DTFP macroinstruction. For further information, see the *IBM System/34 RPG II Reference Manual*, the *IBM System/34 Basic Assembler and Macro Processor Reference Manual*, and the *IBM System/34 COBOL Reference Manual*.

FORMSNO: The FORMSNO parameter specifies the forms number of the forms to be used. The forms number can be any combination of up to 4 characters except commas (,), single quotes ('), and blanks. However, question marks (?), slashes (/), and hyphens (-) should not be used because they have special meanings within procedures. If FORMSNO is not specified, the forms number set previously by a FORMS statement is used. If the forms number was not set during the session, the forms number specified for the display station during system configuration or assigned by the SET procedure or the $SETCF utility program is used.

SPOOL: SPOOL-YES specifies that output is to be spooled. SPOOL-NO specifies that output is not to be spooled; if the printer is not available, a message is displayed. If spooling was not selected during system configuration, the SPOOL parameter is ignored. If SPOOL is not specified, the default depends on how the system was configured at system configuration time. If 1 (yes) was selected for the spool all printers prompt, the default is SPOOL-YES. If 0 (no) was selected, the default is SPOOL-NO for all printers except the system printer.

COPIES: The COPIES parameter specifies the number of copies of spooled printer output to be printed for the job step. The number can be any decimal number from 1 through 99. If the COPIES parameter is not specified, 1 is assumed. The COPIES value affects only spooled output for the job step.

DEFER: DEFER-YES specifies that the system should not begin printing spooled output from the job step until the step is complete. DEFER-NO specifies that the system can begin printing spooled output from the job step before the step is complete. The DEFER parameter affects only spooled output. If the DEFER parameter is not specified, DEFER-YES is assumed.

Note: An overall throughput improvement might result if you specify DEFER-NO for long-printing job steps; however, due to increased system overhead, the job may actually take longer to run if you specify DEFER-NO.

ALIGN: The ALIGN parameter can be used to allow the operator to align the forms before printing of output for the job step begins. If the ALIGN parameter is specified, it overrides any alignment indicator used in the program. If the ALIGN parameter is not specified, the alignment indicator specified in the program is used.

If ALIGN-YES is specified, the system prints the first line of output and issues a message. The operator can then align the forms and select an option that causes the first line to be printed and the alignment message to be displayed again or an option that causes printing to continue with the second line of output. If ALIGN-NO is specified, the system prints the job step output without allowing the operator to align the forms.

PRIORITY: The PRIORITY parameter assigns a priority to spooled output from the job step. The priority can be any decimal number from 0 through 5. The system prints jobs in order of decreasing priority. For example, all print output with a priority of 5 is printed before any other jobs on the spool file. Jobs with the same priority are printed in the order that they were placed on the spool file. Priority 1 jobs are the last jobs printed by the system. Priority 0 jobs are not printed until the system operator uses the RELEASE control command to specifically release them. If the PRIORITY parameter is not specified, PRIORITY-1 is assumed.

DEVICE: The DEVICE parameter identifies the printer to be used for the file. DEVICE-SYSTEM specifies that the system printer is to be used. DEVICE-ws-id specifies the work station ID of the printer to be used. (Work station IDs are assigned during system configuration.) If DEVICE is not specified, the job step uses either the printer assigned to the display station during system configuration, or the printer assigned by the SET procedure or the $SETCF utility program. If the DEVICE parameter is not specified and if the job is run from the input job queue, the default printer is used.

HOLD: HOLD-YES specifies that the printed output remains in the spool file after all copies of the output have been printed; the number of copies to be printed is set to 1 when the output is held in the spool file. HOLD-NO specifies that the printed output is removed from the spool file after the output has been printed. If the HOLD parameter is not specified, HOLD-NO is assumed.

TYPE: TYPE-ANY specifies that the printer output can be printed on any type of printer. If an ideographic character is encountered and the printer cannot print ideographic characters, the character position is blanked. If the TYPE parameter is not specified, TYPE-ANY is assumed.

TYPE-IGC specifies that the output contains ideographic data and should be printed on a printer that can print ideographic characters. If the printer cannot print ideographic characters and if the output is not spooled, ideographic characters will be blanked. If the printer cannot print ideographic characters and if the output is spooled, the operator can choose to hold the spooled output or to print it, with blanks substituted for ideographic characters.

EXTN: The EXTN parameter is for the ideographic version of the SSP and is ignored for nonideographic systems. EXTN-ON specifies that the extended character task is called to process any extended characters in the printer output. EXTN-OFF specifies that the extended character task is not called; the system-defined default ideographic character is printed for any extended characters in the output. If the EXTN parameter is not specified, EXTN-ON is assumed.

CPI: The CPI (character per inch) parameter specifies the horizontal print density to use for printed output. The values that can be specified are 10 or 15. If CPI-15 is used, the output should be printed on a 5225 (Models 1 through 4) Printer or a 5224 (Models 1 and 2) Printer. If CPI-15 is attempted on another printer, an error message will appear.

If the CPI parameter is not specified, the CPI density set previously by a FORMS statement is used. If the CPI density was not set during the session, the system uses a default value of 10.

LPI: The LPI (lines per inch) parameter specifies the vertical print density to use for printed output from the display station session. The values that can be specified are 4, 6, and 8. If the output is printed on a printer other than the 5225 Printer or the 5224 Printer, the LPI parameter is ignored.

If LPI is not specified, the system uses the LPI value set previously by a FORMS statement or the LINES procedure. If the LPI parameter is not specified and the number of lines per inch was not previously set during the session, the system uses the value specified during work station configuration (either 4, 6, or 8). If no LPI value was specified during work station configuration, the system uses an LPI value of 6.

Note: For the 5225 Printer Models 11 and 12 and the 5224 Printer Model 12, an LPI value of 8 will cause characters to vertically overlap from one line to another.

Example

For a job step, output to the file called PRINT1 is to go to the system printer and is to be spooled. Three copies are to be printed and the SSP can begin printing before the job step is complete:

```
// PRINTER NAME-PRINT1,SPOOL-YES,COPIES-3,DEFER-NO,
//         DEVICE-SYSTEM
```

PROMPT Statement

Function

The PROMPT statement allows the user to:

- Prompt for up to 11 procedure substitution parameters with one user-supplied screen format
- Define each parameter for the operator
- Assign default parameters
- Write out the screen format to be read on the first read in a work station program
- Control various SFGR screen functions

The PROMPT statement provides a return code to indicate which command key or function key was pressed. This code is accessible in a procedure using the ?CD? substitute expression. The following chart shows the command/function keys and the equivalent return codes.

Key	Return Code
Enter/Rec Adv	0000
Cmd keys 1-24	2001-2024
Roll Up	2090
Roll Down	2091
Help	2092
Record Backspace (Home key pressed while the cursor is in the Home position)	2093

The SSP resets the return code to 0000 whenever it executes a RUN OCL statement.

The PROMPT statement can also be used to overlap the entering of input data on the first screen of a program with the processing of remaining procedure OCL statements (similar to the read-under-format function described in Chapter 6).

Note: The PROMPT OCL statement cannot be used to display a screen format that contains more than 88 characters of execution time output data.

Placement

The PROMPT statement can be placed anywhere within the procedure for which the parameters are being prompted. It cannot be used in an MRT procedure, nor is it allowed from the job queue or keyboard.

Format

// PROMPT MEMBER–screen format load member name,

FORMAT–display screen format name [,PDATA–{NO / YES}]

[,UPSI–{NO / YES}]

Parameters

MEMBER: The MEMBER parameter specifies the name of the screen format load member that contains the display screen format.

FORMAT: The FORMAT parameter specifies the name of the display screen format that is to be displayed. The display screen format can contain up to eleven 8-byte input, output, or input/output fields.

PDATA: The PDATA parameter specifies whether the input from the screen is to be retrieved and used as parameters, or held as input for the user program's first input request. If PDATA-NO is specified, the input from the screen is stored in the procedure parameter save area and treated as if it had been entered with the procedure. If PDATA-YES is specified, the user program will be given the screen input on the first input request to the work station file.

Note: The PROMPT-OCL statement cannot be used with a format that requires more than 88 characters of execution output data. Also, if PDATA-NO or the PDATA parameter is not specified, the PROMPT OCL statement cannot be used to display a screen that has more than 88 characters of input data. If PDATA-YES is specified, however, the maximum amount of input data is controlled by the screen format and by the user program that reads the format, and it can exceed 88 characters.

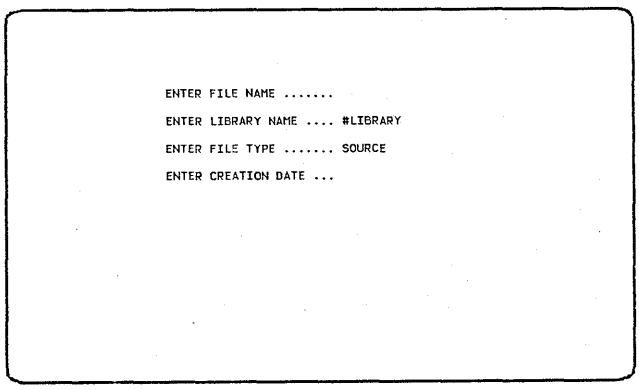
UPSI: The UPSI parameter specifies whether the current setting of the eight UPSI switches is mapped into the SFGR indicator table. If UPSI-NO is specified, PROMPT processing is unchanged. If UPSI-YES is specified, the current setting of the eight UPSI switches U1 through U8 is mapped into the SFGR indicator table as indicators 91 through 98.

Example

The following PROMPT statement displays a format named SCR1 from a format load member named MBR1. The input to the display is handled as procedure substitution parameters:

```
// PROMPT MEMBER-MBR1,FORMAT-SCR1,PDATA-NO
```

Following is the display SCR1:



Following are the S and D specifications for display SCR1 and the general rules for coding S and D specifications for formats displayed by the PROMPT statement.

IBM

System/34 Display Screen Format Specifications

Second Edition

Use this coding sheet only to define display screen formats for WSU and $SFGR. This coding sheet could contain typographical errors.

GX21-9253- U/M 050*
Printed in U.S.A.
*No. of sheets per pad may vary slightly.

S

Sequence Number	Form Type	Format Name	Format ID (WSU Only)	Start Line Number	Number of Lines to Clear	Lowercase	Return Input	Reset Keyboard (WSU Only)	Sound Alarm	Enable Function Keys	Enable Command Keys	Blink Cursor	Erase Input Fields	Override Fields	Suppress Input	Reserved	WSU Only: Enter Mode Sequence: Start	End	Entry Required	Repeat	Reserved	Priority	Preprocess	Review Mode Record Identifying Indicators: 1	2	3	Insert Mode Record Identifying Indicators: 1	2	3	Reserved	Key Mask	Reserved
1 2 3 4 5	6	7 8 9 10 11 12 13 14	15 16	17 18	19 20	21	22	23 24	25 26	27	28	29 30	31 32	33 34	35 36	37 38 39 40	41	42	43	44	45	46	47	48 49	50 51	52 53	54 55	56 57	58 59	60 61 62 63	64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	80
	S	SCR1																														

D

Sequence Number	Form Type	Field Name: WSU Field Name	Not Used by WSU	Field Length	Starting Location: Line Number	Horizontal Position	Output Data	Edit Code (WSU Only)	Input Allowed	Data Type	Mandatory Fill	Mandatory Entry	Self-Check	Adjust/Fill	Position Cursor	Enable Dup	Controlled Field Exit	Auto Record Advance	Protect Field	High Intensity	Blink Field	Nondisplay	Reverse Image	Underline	Column Separators	Reserved	Constant Type	Constant Data	Continuation
1 2 3 4 5	6	7 8 9 10 11 12	13 14	15 16 17 18	19 20	21 22	23 24	25	26	27	28	29 30	31	32 33	34	35	36	37 38	39 40	41 42	43 44	45 46	47 48	49	50 51 52 53 54 55	56	57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	80
	D			23	05	20	Y																				CENTER FILE NAME	
	D			80	54	40	1		Y																			
	D			23	07	20	Y																				CENTER LIBRARY NAME	
	D			80	74	40	2		Y																		#LIBRARY	
	D			23	09	20	Y																				CENTER FILE TYPE	
	D			60	94	40	3		Y																		SOURCE	
	D			20	95	18	0		Y										Y			Y						
	D			23	11	20	Y																				CENTER CREATION DATE ...	
	D			81	14	40	4		Y																			
	D							

The following rules apply to the D specifications for input fields and output/input fields on a format displayed by the PROMPT statement.

- Fields must be defined in parameter number order on the D specification. They do not, however, have to appear in parameter number order on the display screen. (The position on the display screen is determined by the line number and horizontal position entries.)

- The total field length for each positional parameter must be eight character positions. If you want to limit the operator to entering fewer than eight characters, you must define two fields: the input field where the operator enters the parameter and a fill field that completes the parameter. The fill field can be a nondisplay field and protected. In this example, the file type field is limited to six characters; therefore, an additional 2-character fill field is required. This fill field can occupy any vacant location on the screen; however, it must follow the modified input field on the D specification to obtain the desired results when the procedure accesses the input buffer created by the $SFGR.

- Normally, the output data entry (columns 23 and 24) should be an indicator number that corresponds to the parameter position in the procedure. If a parameter has already been assigned a value when the prompt format is displayed, the indicator for the parameter is turned on, and the assigned value is displayed. If the parameter has not already been assigned a value, the constant default value coded in columns 57 through 64 is displayed. If an indicator number greater than 11 is specified in the output data entry, the default value in columns 57 through 64 will always be displayed when the format is displayed by the PROMPT statement.

  This example shows indicators coded in columns 23 and 24 for displayed fields and fill fields. Indicators allow you to display parameters that have already been assigned a value. If parameters do not have values previously defined or if you want to ignore previously assigned values, you can code a Y in column 23 for all display fields. If you code a Y in column 23 for any parameter field or fill field, you should code a Y for all parameter fields and fill fields in the format. Similarly, if you code an indicator in columns 23 and 24 for any parameter field or fill field, you should code indicators for all parameter fields and fill fields in the format.

  Note: An indicator value of 80 is used for the fill field in this example. This indicator is never turned on; therefore, blanks will always be in the field when it is displayed.

- Y must be specified in the input allowed entry (column 26).

- Up to an 8-character default value can be coded in the Constant Data entry (columns 57 through 64).

REGION Statement

Function

The REGION statement specifies the region size for a job or job step. The value specified overrides the session default region size that was set by the SET procedure or the $SETCF utility program.

The system ensures that an amount of main storage at least as large as the job region is available for use by the job steps in the job. (If a job step releases the requesting display station or if the job step runs an MRT [multiple requestor terminal] procedure, the job step is, in effect, a new job. The SSP does not ensure that enough storage will be available for such job steps.)

Note: The SSP considers all user storage not occupied by non-swappable programs as available storage. (IBM-supplied, non-swappable programs include portions of MRJE, portions or all of SSP-ICF subsystem tasks, BSC support for RPG II, data communications programs, and non-swappable spool writers.) If no non-swappable programs are in storage, available storage is equal to user storage.

The job step region size is the maximum amount of storage that can be used by programs that dynamically request storage beyond their load member size. The job step region size is used to limit the size of a job step that uses a variable amount of main storage such as the sort program of the IBM Utilities Program Product.

Note: The REGION statement can be used to cause a COBOL loaded sort to execute in a larger memory space than is needed for the COBOL program.

For a discussion of programming considerations for specifying region size, see *Specifying Region Size for a Job* and *Specifying Region Size for a Job Step* in Chapter 6.

Placement

The REGION statement can appear anywhere among the OCL statements, except between a LOAD statement and a RUN statement.

When used to specify a job region size, the REGION statement must precede the first LOAD statement in the job.

Any REGION statement appearing after the first REGION statement in a job or after the first LOAD statement in a job applies only to the next job step. If more than one REGION statement is specified for a job step, the region size specified on the last REGION statement is used.

Format

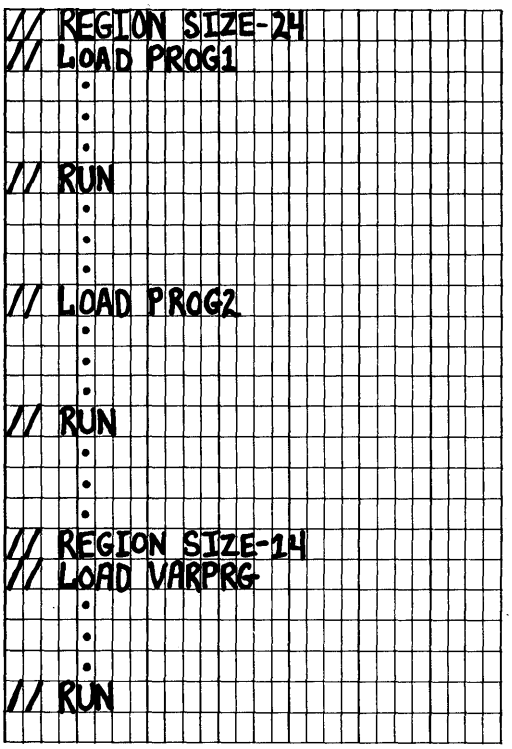
```
// REGION SIZE=xx
```

Parameter

SIZE: The SIZE parameter specifies the region size in K (K = 1,024) bytes. xx can be any decimal number from 1 through 64, but may not exceed the number of K bytes of main storage available to the user. The region assigned by the SSP is rounded up to an even number of K bytes.

Example

PROCA is a first-level procedure. The largest program executed as part of PROCA uses 24K bytes of main storage. The program called VARPRG, executed in PROCA, is capable of using a variable amount of main storage but does not run more efficiently if more than 14K bytes of main storage are used. The following statements are coded as part of PROCA:



RESERVE Statement

Function

The RESERVE statement specifies that a reserved disk area should be allocated for a job and specifies the size of the reserved area. The reserved area is an area on disk that is set aside to contain the scratch (RETAIN-S) and job (RETAIN-J) files used by a job. By using a RESERVE statement, you can ensure that enough disk space will be available for the scratch and job files used by the job.

Once a reserved area is allocated, the SSP allocates all new scratch and job files for the job in the reserved area. If enough space is not available in the reserved area for a new scratch or job file, the SSP issues an error message and the operator can choose to allocate the file elsewhere in the user area on disk or to cancel the job.

At the end of each job step, the SSP automatically compresses the reserved area if necessary (if any job files were deleted during the step). At the end of the job, the SSP automatically deletes the reserved area from the disk.

Notes:

1. A job step that runs an MRT program or a job step that releases the requesting display station, in effect, starts another job. Therefore, scratch and job files created by those job steps are not allocated in the reserved area for the job.
2. If a reserved area exists for the job, the SSP ignores the LOCATION parameter on FILE OCL statements for new scratch and job files.

Placement

If a RESERVE statement is used, it cannot be placed between the LOAD and RUN statements in a job. Only one RESERVE statement can be used in a job.

Format

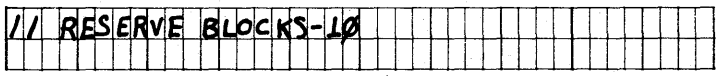
// RESERVE BLOCKS–xxxxx

Parameter

BLOCKS: The BLOCKS parameter specifies the length, in number of blocks, of the reserved area. If the specified number of blocks is not available in the user area on disk, the SSP issues an operator message and waits for the space to become available. If the space does not become available, the operator can use the Attn key to interrupt and then cancel the job.

Example

Assign a reserved area 10 blocks long:



RUN Statement

Function

The RUN statement indicates the end of the OCL statements for a job step. After the system reads the RUN statement, it executes the program named in the LOAD statement.

Placement

A RUN statement is needed for each program run on the system. In the job stream, it must be the last statement within the set of OCL statements for each job step.

Note: Utility control statements for the program may follow the RUN OCL statement.

Format

```
// RUN
```

Parameters

None

Example

Load and run a program called PASTDUE:

```
// LOAD PASTDUE
// RUN
```

SESSION Statement

The SESSION statement is not documented in this manual. For information about the SESSION statement, see the *Interactive Communications Feature Reference Manual.*

SWITCH Statement

Function

The SWITCH statement sets one or more of the external indicators (also known as UPSI switches) for the display station to on (1) or off (0). The switch setting remains in effect until one of the following occurs:

- Another SWITCH statement is used.
- The display station session is ended (all indicators are set off).
- A user program changes the setting of any of the indicators.

A job placed on the input job queue uses a copy of the external indicators for the display station as they existed when the job was placed on the queue.

Notes:

1. If an SSP procedure changes the setting of a switch, the switch returns to its original setting when the SSP procedure ends.
2. A set of external indicators also exists for each executing MRT procedure.

Placement

The SWITCH statement can appear anywhere among the OCL statements.

Format

// SWITCH indicator settings

Parameters

indicator settings: The indicator settings parameter consists of 8 characters, one for each of the eight external indicators (U1-U8). The first, or leftmost, character gives the setting of indicator U1, the second character gives the setting of indicator U2, and so on.

The parameter must always contain 8 characters. For each indicator, one of the following characters must be used:

Character	Meaning
0	Set the indicator off
1	Set the indicator on
X	Leave the indicator as it is

Note: If you specify multiple switch statements in a procedure, the result reflects the last time each switch was set on or off by a SWITCH statement.

Example

```
// SWITCH 1X0110XX
```

The example causes the following results:

Indicator	Result
U1	Set on
U2	Unaffected
U3	Set off
U4	Set on
U5	Set on
U6	Set off
U7	Unaffected
U8	Unaffected

SYSLIST Statement

Function

The SYSLIST (system list) statement changes the method of listing system list output. (System list output is the output generated by all SSP utility programs except for the data communications utility programs and service aids. Assembler programs can optionally generate system list output.) The SYSLIST statement causes the system list output to be:

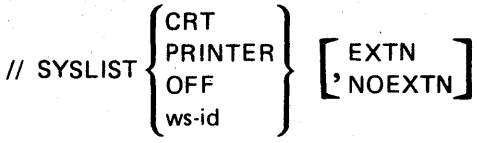
- Listed on the printer that was assigned to the display station during system configuration
- Listed on one of the other printers
- Displayed at the display station
- Not listed at all

The SYSLIST statement remains in effect until the system processes another SYSLIST statement from the display station, until the SYSLIST procedure is run from the display station, or until the display station session is ended. A PRINTER OCL statement with NAME-$SYSLIST specified further controls the system list output for one job step if the system list output is going to a printer. In this case, the device specified on the printer statement overrides the device specified on the SYSLIST statement.

Placement

The SYSLIST statement can appear anywhere among the OCL statements.

Format



Parameters

CRT: Display output on the display screen at the requesting display station. Up to 18 lines are displayed at a time on a 1920-character display screen. Up to 9 lines are displayed at a time on a 960-character display screen. After each system list display has been shown, the operator can request that the next set of lines be displayed or that the display be terminated. If the operator enters a blank, the number of lines shown on the next display will be the same as the number of lines shown on the previous display. If the operator enters 0, the display will be terminated. If the operator enters any characters other than a blank or 0 through 18 for a 1920-character display screen, the next 18 lines are displayed; if less than 18 lines remain, fewer lines are displayed. If the operator enters any character other than a blank or 0 through 9 for a 960-character display screen, the next 9 lines are displayed; if less than 9 lines remain, fewer lines are displayed.

Note: System output from MRT (multiple requestor terminal) programs, from programs that release the requesting display station, and from programs that are run from the input job queue is listed on the system printer.

PRINTER: Print output on the printer that was assigned to the display station during system configuration or assigned by the SET procedure or the $SETCF utility program.

OFF: Do not print or display output.

ws-id: Print output on the printer with the specified work station ID (ws-id). The work station IDs are assigned during system configuration and can be modified by the ASSIGN control command. To print output on the system printer, specify the work station ID of the system printer.

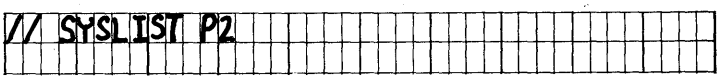
EXTN: Print or display extended characters in the system list output.

NOEXTN: Do not print or display extended characters in the system list output. The system-defined ideographic character is listed for any extended characters.

Note: The EXTN and NOEXTN parameters are for the ideographic version of the SSP and are ignored for nonideographic systems. When a session begins, extended character processing for system list output is assumed. During the session, the EXTN and NOEXTN parameters can be used to turn on and turn off extended character processing for system list output.

Example

The following is an example of assigning printer P2 to list system list output:



WORKSTN Statement

Function

The WORKSTN statement supplies the system with information about a display station used by a program.

A display station can be assigned for use by a program in one of the following ways:

- The requesting display station is automatically acquired, unless the program is placed on the input job queue. (No WORKSTN statement is required for the requesting display station.)
- A WORKSTN statement specifying REQD-YES causes the system to acquire the specified display station.
- The program can perform an acquire operation. If the program acquires a display station, a WORKSTN statement is optional. Even though the program acquires the display station, you might want to use a WORKSTN statement so that you can specify information about the display station.

Placement

The WORKSTN statement can appear anywhere among the OCL statements.

Format

```
// WORKSTN UNIT–ws-id [,SYMID–symbolic ws-id] [,REQD–{YES}]
                                                 {NO }

[,RESTORE–{YES}] [,PRINT–{ws-id}]
          {NO }          {NO  }

[,EXTN–{ON }]
       {OFF}
```

Parameters

UNIT: Specifies the 2-character work station ID of the display station to be used by the program. The work station ID was assigned to the display station during system configuration or was assigned by the ASSIGN control command.

SYMID: Specifies the symbolic work station ID used by the program. The symbolic work station ID is a 2-character ID (the first character must not be numeric) that the program passes to or receives from display station data management. The symbolic ID cannot be the same as the work station ID assigned to any other display station or printer on the system. The symbolic work station ID can be the same as the work station ID specified by the UNIT parameter. If SYMID is not specified, the symbolic work station ID is assumed to be the same as the work station ID specified in the UNIT parameter.

REQD: If a display station needs to be acquired, the REQD parameter specifies whether the SSP is to acquire the display station or whether the application program is to acquire the display station. REQD-YES indicates that the SSP is to acquire the display station; REQD-NO specifies that the application program is to acquire the display station. If the REQD parameter is not specified, REQD-NO is assumed. If the display station is the requesting display station, it does not need to be acquired.

RESTORE: Specifies:

- Whether or not the COMMAND display (or MENU display if a menu is being used) should be restored when a command display station is released by the program or when the job terminates.

- Whether or not the STANDBY display should be restored when a data display station is released by the program or when the job terminates.

RESTORE-YES indicates that the last information displayed by the program is replaced by the COMMAND (or MENU) display on a command display station or by the STANDBY display on a data display station. RESTORE-NO indicates that the last information displayed by the program remains on the display screen. The operator must press the Enter key to restore the COMMAND, MENU, or STANDBY display. If RESTORE is not specified and the last display station operation was an input to the program, the COMMAND (or MENU) or STANDBY display is restored. If RESTORE is not specified and the last display station operation was an output from the program, the COMMAND (or MENU) or STANDBY display is not restored.

PRINT: Specifies the printer to be used as the copy device for the display station when the operator presses the Print key. If the specified printer is already being used when the operator presses the Print key, an error message is displayed and the operator must press the Error Reset key; no information is printed. If PRINT-ws-id is specified, ws-id is the 2-character work station ID assigned to the printer during system configuration or assigned by the ASSIGN control command. If PRINT-NO is specified, the Print key is ignored for the display station. If PRINT is not specified, the printer *assigned to the display station that submitted the job* is used.

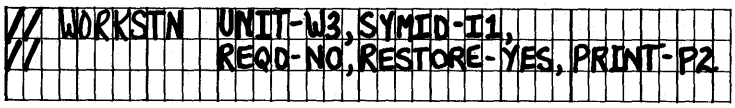
Note: The PRINT parameter is in effect only while the display station is allocated to the program (the display station is in data mode). If the operator presses the Print key while the display station is in command mode, the printer assigned to the display station during system configuration is used.

EXTN: The EXTN parameter is for the ideographic version of the SSP and is ignored for nonideographic systems. EXTN-ON specifies that the extended character task is called to process any extended characters in the data stream to be displayed. EXTN-OFF specifies that the extended character task is not called; the system-defined default ideographic character is displayed for any extended characters in the data stream. If EXTN is not specified, EXTN-ON is assumed.

Note: The EXTN parameter does not affect characters entered from the keyboard at the display station.

Example

Display station W3 is to be acquired for use by the program. The program uses the symbolic name I1. The COMMAND display is restored when the display station is released. If the operator presses the Print key, displayed information is printed on printer P2.



* (comment) Statement

Function

Comment statements are usually used to explain the purpose of the OCL statements and utility control statements stored in a procedure. Comments in a procedure are displayed when the procedure is displayed. Comments are not displayed when the procedure is executed. (See also the descriptions of the // * (message) and // ** (message) statements in this chapter.)

Although comments are useful for explaining statements within a procedure, a large number of comments can significantly increase the execution time of the procedure.

Placement

Comment statements can appear anywhere among the OCL statements except between IF and ELSE expressions within a procedure member. For an explanation of IF and ELSE expressions, see *Substitution Expressions* in Chapter 5.

Format

* comment

Parameters

The comment can be any combination of words and characters, except the question mark (?).

Note: For the ideographic version of the SSP, the comment can contain ideographic characters.

/* (end of data) Statement

Function

The /* statement indicates the end of a data file entered from the keyboard or the end of inline source data.

Placement

A /* statement must be the last record of a data file entered from the keyboard.

Format

/*

Parameters

None

// * (message) Statement

Function

The // * statement causes a message to be displayed at the display station that submitted the job unless the //* statement is in a job run from the input job queue or a job that released its requesting display station. In those cases, the //* statement causes a message to be displayed at the system console. (See also the descriptions of the * (comment) and // ** (message) statements in this chapter.)

Note: The message is not displayed if you have previously entered IDELETE.

Placement

The message statement can appear anywhere among the OCL statements.

Format

```
// * {message identification code}
   {'message'                    }
```

Parameters

message identification code: Specifies the message identification code of a message in the active first-level (USER1) message member for the display station. (The MEMBER OCL statement assigns the first-level message member.) The text of the specified message is displayed.

'message': The 'message' parameter is a character string enclosed by single quotation marks. The character string is the actual message. (See Chapter 5 for a description of the use of the question mark in an OCL message statement.) Any character except a single question mark (?) can be used in the character string. If the message contains imbedded quotes (such as the apostrophe in o'clock), then enter each imbedded quote as double quotes (''). The character string has a maximum of 120 characters, minus the OCL statement characters (slashes, asterisks, single quotes, and blanks) not included in the actual message. However, when the message is displayed on the display screen, only the first 75 characters of the character string are displayed. When the message is printed, all characters in the character string are printed.

The message is always displayed when the statement is processed in the job stream.

For information about creating message members, see the description of the $MGBLD utility program in Chapter 4.

Note: For the ideographic version of the SSP, the message text can contain ideographic characters. If an attempt is made to send ideographic characters to a display station that cannot display ideographic characters, the SSP displays periods for those characters.

Example

In the following example, the message statement is followed by a PAUSE statement to allow the operator to change the diskettes:

// ** (message) Statement

Function

The // ** statement causes a user-specified message to be displayed at the system console. The message is displayed with a message identifier of SYS-1118. The job stops processing until the system operator responds to the message by entering a 0 (zero) option. (See also the descriptions of the * (comment) and // * (message) statements in this chapter.)

Note: If the procedure containing the // ** OCL statement is run from the display station assigned as the system console, the message is displayed at the system console in command mode.

Placement

The message statement can appear anywhere among the OCL statements.

Format

```
// ** {message identification code}
   {'message'                    }
```

Parameters

message identification code: Specifies the message identification code of a message in the active first-level (USER1) message member for the display station. (The MEMBER OCL statement assigns the first-level message member.) The text of this message is the text of the message issued with the message identifier of SYS-1118.

'message': The 'message' parameter is a character string enclosed by single quotation marks. The character string is the actual message. Any character except a single question mark (?) can be used in the character string. (See Chapter 5 for a description of the use of the question mark in an OCL message statement.) If the message contains imbedded quotes, (such as the apostrophe in o'clock), then enter each imbedded quote as double quotes (''). The character string has a maximum of 120 characters, minus the OCL statement characters (slashes, asterisks, single quotes, and blanks) not included on the actual message. However, when the message is displayed on the display screen, only the first 75 characters of the character string are displayed. When the message is printed, all characters in the character string are printed.

The message is always displayed when the statement is processed in the job stream.

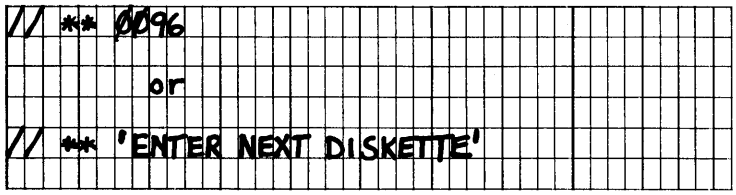
Note: For the ideographic version of the SSP, the message text can contain ideographic characters. If an attempt is made to send ideographic characters to a display station that cannot display ideographic characters, the SSP displays periods for those characters.

Example

In the USER1 message member, the message text associated with MIC 0096 is ENTER NEXT DISKETTE. To cause the following message to appear on the display screen:

```
SYS-1118        OPTIONS (0   )
ENTER NEXT DISKETTE
```

you can enter:



Chapter 2. SSP Procedures

This chapter describes the following procedures, which are supplied as part of the SSP:

ALTERBSC	DEFINEPN	LISTFILE	RENAME
ALTERSDL	DEFINX21	LISTLIBR	REQUESTX
BACKUP	DELETE	LOG	RESPONSE
BLDFILE	DISABLE	ORGANIZE	RESTORE
BLDLIBR	DISPLAY	OVERRIDE	SAVE
BLDMENU	ENABLE	POST	SAVELIBR
BUILD	FORMAT	PRESTOR	SET
CATALOG	FROMLIBR	PRLIST	SETFILE
COMPRESS	HELP	PRMENU	SETRETRY
CONDENSE	HISTCRT	PROF	SPECIFY
COPYI1	HISTORY	PRSAVE	STARTM
COPYPRT	INIT	PRSRC	STOPM
CREATE	JOBSTR	PRSRCID	SYSLIST
CRESTART	KEYSORT	RELOAD	TOLIBR
DATE	LIBRLIBR	REMOVE	TRANSFER
DEFINEID	LINES	RENAME	XREST
			SAVE

Notes:

1. Program product procedures are not described in this manual. Descriptions of these procedures can be found in the respective program product reference manuals.
2. The procedures that are used during program product installation and modification are not described in this manual. Descriptions of these procedures can be found in the *Installation and Modification Reference Manual*.
3. The SSP service procedures and commands (procedures that help you and your IBM service personnel solve system problems that might arise) are not described in this chapter. The SSP service procedures are APAR, APPLYPTF, DFA, DUMP, ERAP, PATCH, SEC, SETDUMP, and TRACE. These service procedures are described in Appendix D of this manual.
4. Some SSP procedures call and use other SSP procedures that you cannot call directly. Although you cannot call these procedures directly, their names might appear on listings you request.
5. The SSP procedures used for data communications (MRJE, DCPRINT, SRJE, and DCFORMS) are not described in this manual. Descriptions of these procedures can be found in the *Data Communications Reference Manual*.

The following information is given for each procedure described in this chapter:

- The function, or functions, of the procedure.
- The format of the command statement that calls the procedure. For a description of the conventions used for statement formats, see *Conventions Used for Describing Statement Formats* at the front of this manual.
- Descriptions of any command statement parameters.
- An example, or examples, of how the procedure is used.

The OCL statements and utility control statements generated when the SSP procedures are run are listed in Appendix E.

ALTERBSC Procedure

Function

The ALTERBSC procedure alters the following communications adapter items for BSC:

Item	Parameters
Bits-per-second (bps) rate	BRATE
Business machine clocking	CLOCK
Error retry count	ERC
Switched network backup	SLINE
IBM modem test	TEST
Non-US answer tone	TONE
Line number	LNUM

Additional BSC operational items that can be altered are included in the OVERRIDE procedure. To identify the current values of these parameters, use the STATUS control command.

Running ALTERBSC alters the values only in the display station communications configuration record and affects only BSC programs that are loaded from the display station from which ALTERBSC was run.

Note: The ALTERBSC procedure *does not* alter the communications adapter items when it is run from a display station in Inquiry mode.

If an SSP-ICF subsystem is to be enabled and the ALTERBSC procedure is required, you must run the ALTERBSC procedure on the system console. If you run ALTERBSC while a subsystem is enabled, the subsystem must be disabled and enabled again before the new values will be used.

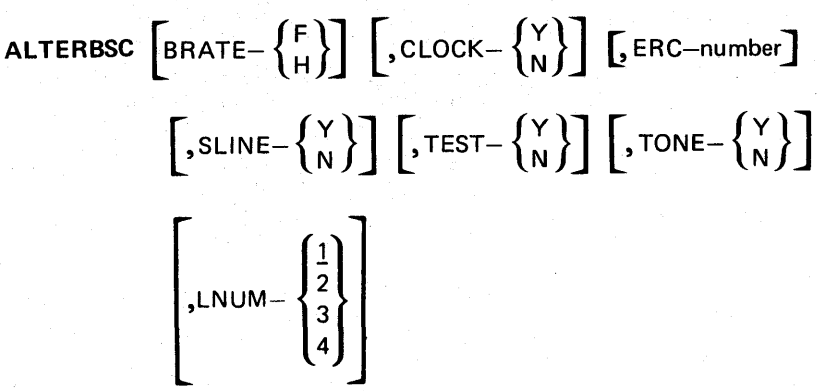
Changes made by ALTERBSC remain in effect until:

- The items are changed again by the $SETCF utility program or the ALTERBSC procedure.
- The system library is reloaded. When the system library is reloaded:
  - CLOCK, TEST, and TONE are set to the values specified during system configuration.
  - The switched network backup line is not used (SLINE-N).
  - The full rated speed of the modem is used (BRATE-F).
  - The error retry count (ERC) from the executing program is used.
- The system is configured again.

The ALTERBSC procedure runs the $SETCF utility program.

Note: The ALTERBSC procedure is intended only for data communications programming that uses BSC. For background information on BSC, see the *General Information—Binary Synchronous Communications Manual.*

Command Statement Format



Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If any parameters are omitted, the corresponding values in the display station communications configuration record are not changed.

Parameters

BRATE: BRATE-F specifies that the full rated speed of the modem is used. BRATE-H specifies that only half the rated speed of the modem is used.

CLOCK: CLOCK-Y specifies that System/34 must provide the programmed clocking facility. CLOCK-N specifies that the modem or another external source has the clocking facility.

ERC: The number of error retries to be attempted. Any decimal number from 1 through 255 can be specified. If the system library is reloaded, the error retry count from the executing user program is used.

When using the SSP-ICF BSC support and the system library is reloaded, ERC defaults to 7. It is possible, however, to override the ERC value with the ENABLE procedure.

SLINE: SLINE-Y specifies that a switched network backup line is used as backup (standby) for the nonswitched primary line. SLINE-N specifies that the switched network backup line is not used.

Note: After specifying SLINE-Y, if you run a batch RPG II or a BSC basic assembler job, the operator can use the OVERRIDE procedure to specify manual call, manual answer, or automatic answer. Otherwise, the connection defaults to manual call or manual answer depending upon the first line operation performed by the user program. If the first operation is a transmit operation, manual call is assumed; if the first operation is a receive operation, manual answer is assumed.

TEST: TEST-Y specifies that an IBM modem is being used. The automatic wrap test includes modem testing when a permanent error occurs. TEST-N specifies that a non-IBM modem is being used. The automatic wrap test does not include modem testing.

TONE: TONE-Y specifies that a non-US special tone is required for manual answer and automatic answer. TONE-N specifies that a non-US special tone is not required.

LNUM: LNUM specifies the line number to which the parameters in the ALTERBSC statement apply. If LNUM is not specified, LNUM-1 is assumed.

Example

Specify an error retry count of 25:

ALTERSDL Procedure

Function

The ALTERSDL procedure alters the following communications adapter items for SDLC:

Item	Parameter
Bits-per-second (bps) rate	BRATE
Business machine clocking	CLOCK
Line number	LNUM
Switched network backup	SLINE
IBM modem test	TEST
Non-US answer tone	TONE

Additional SDLC operational items that can be altered are included in the SPECIFY procedure. To identify the current values of these parameters, use the STATUS control command.

Running ALTERSDL alters the values only in the display station communications configuration record and affects only SDLC programs that are loaded from the display station from which ALTERSDL was run. ALTERSDL does not affect SDLC information for remote work stations or for SSP-ICF.

Note: The ALTERSDL procedure *does not* alter the communications adapter items when it is run from a display station in inquiry mode.

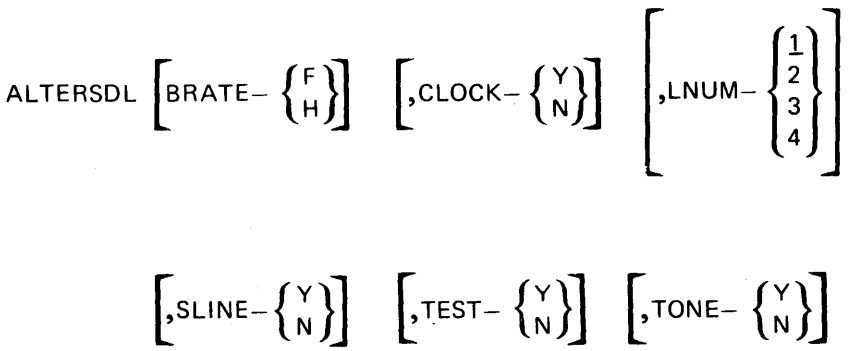
Changes made by ALTERSDL remain in effect until:

- The items are changed again by the $SETCF utility program or the ALTERSDL procedure.
- The system library is reloaded. When the system library is reloaded:
  - CLOCK, TEST, and TONE are set to the values specified during system configuration.
  - The switched network backup line is not used (SLINE-N).
  - The full rated speed of the modem is used (BRATE-F).
- The system is configured again.

The ALTERSDL procedure runs the $SETCF utility program.

Note: The ALTERSDL procedure is intended only for data communications programming that uses SDLC. For background information on SDLC, see the *IBM Synchronous Data Link Control General Information Manual.*

Command Statement Format



Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If any parameters are omitted, the corresponding values in the display station communications configuration record are not changed.

Parameters

BRATE: BRATE-F specifies that the full rated speed of the modem is used. BRATE-H specifies that only half the rated speed of the modem is used.

CLOCK: CLOCK-Y specifies that System/34 must provide the programmed clocking facility. CLOCK-N specifies that the modem or another external source must provide the clocking facility.

LNUM: LNUM specifies the line number to which the parameters in this procedure apply. If not specified, LNUM defaults to 1.

SLINE: SLINE-Y specifies that a switched network backup line is used as backup (standby) for the nonswitched primary line. SLINE-N specifies that the switched network backup line is not used.

Notes:

1. After specifying SLINE-Y, the operator can use the SPECIFY procedure to specify manual call, manual answer, or automatic answer.
2. For information on making a switch line connection for an inoperable point-to-point line to a remote work station, see the description of data communications problem determination in the *Operator's Guide*.

TEST: TEST-Y specifies that an IBM modem is being used. The automatic wrap test includes modem testing when a permanent error occurs. TEST-N specifies that a non-IBM modem is being used. The automatic wrap test does not include modem testing.

TONE: TONE-Y specifies that a non-US special tone is required for manual answer and automatic answer. TONE-N specifies that a non-US special tone is not required.

BACKUP Procedure

Function

The BACKUP procedure copies the system library (#LIBRARY) to a diskette file. The BACKUP procedure cannot be used to copy a user library. For information about backing up user libraries, see the description of the FROMLIBR procedure.The diskette file created by the BACKUP procedure contains:

- A stand-alone program that, together with the RELOAD procedure, can change the size of the system library directory and the size of the system library.

  For information about changing the directory and library sizes, see the description of the RELOAD procedure later in this chapter.

- The reorganized system library. (Unused space between members is collected at the end of the library.)

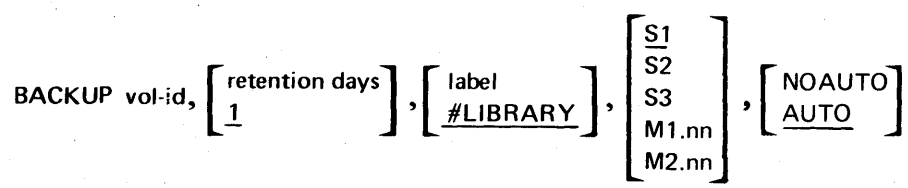
  To return the reorganized system library to the disk, an IPL must be performed from the diskette(s) created by the BACKUP procedure, or the RELOAD procedure must be used. The vol-id of the first diskette created by the BACKUP procedure becomes the vol-id of the disk file during the RELOAD operation.

The BACKUP procedure can be run only from the system console and cannot be run while any other jobs are being run.

The BACKUP procedure runs the $BACK utility program.

For information about determining the number of diskettes required to contain the system library, see the *Installation and Modification Reference Manual*.

Command Statement Format



Note: Parameters 4 and 5 are ignored for systems without a diskette magazine drive.

Parameters

vol-id: The volume ID of the diskette(s); 1 to 6 alphameric characters.

Note: When several diskettes are required for the BACKUP procedure and each diskette has a unique volume ID, the volume ID of the first diskette is the vol-id parameter you must specify for the BACKUP procedure. The vol-id parameter from the BACKUP procedure is used as the PACK parameter in the FILE OCL statement for the $BACK utility program. The SSP checks the volume ID of the second and each succeeding diskette with the volume ID of the diskette that preceded it. If the volume IDs are not the same, the SSP issues error message SYS–1493. The system ignores the error and continues processing if the operator selects option 0.

retention days: The length of the retention period (0 through 999 days) for the diskette file. If the retention period is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file.

label: The label of the single file that is created on the diskette(s). If the label is not specified, the name of the system library (#LIBRARY) is assumed.

CAUTION
The label parameter does not specify the name of a user library to be copied. Only the system library can be copied with the BACKUP procedure.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a fourth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, BACKUP uses only the specified slot. When the end of a diskette is reached, the SSP displays a message; the operator must insert the next diskette in the specified slot.

- If M1.nn or M2.nn is also specified, BACKUP uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine in the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, BACKUP uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, BACKUP uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a fourth parameter is not specified, AUTO is assumed.

Examples

Example A

Back up the system library onto a diskette labeled BCKUP2. The created file will be called #LIBRARY and should be saved for seven days:

```
BACKUP BCKUP2,7
```

Example B

Back up the system library onto diskettes labeled BCKP. The diskettes to receive the copy are in locations M1.07 through M1.10. The file created will be called #LIBRARY and should be saved for 30 days:

```
BACKUP BCKP,30,,M1.07,NOAUTO
```

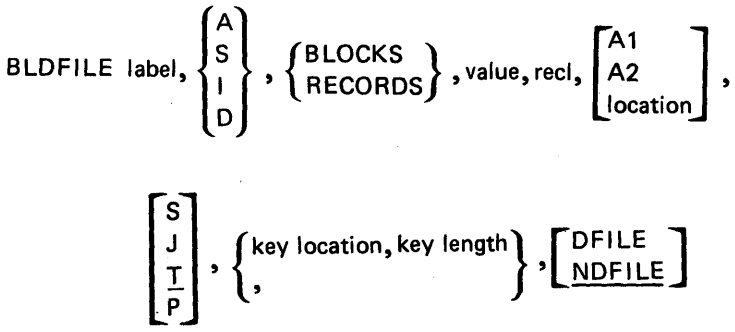
BLDFILE Procedure

Function

The BLDFILE procedure creates a disk file that contains no data records. The file can then be referenced as an existing file by subsequent jobs and job steps.

The BLDFILE procedure runs the $FBLD utility program.

Command Statement Format



Parameters

label: The label of the file to be created.

A: An indexed file is to be created with immediate access to added records (also known as the IFILE characteristic).

S: A sequential file is to be created.

I: An indexed file is to be created.

D: A direct file is to be created.

BLOCKS: Space for the file is to be allocated by blocks.

RECORDS: Space for the file is to be allocated by records.

value: The number of records or blocks to be allocated. value can be any number from 1 through 8388607, but cannot be greater than the number of unused blocks or records available for user files. You can use the CATALOG procedure to determine the amount of space available for your files.

recl: The record length in bytes. recl can be any decimal number from 1 through 4096.

A1: The preferred file location is on disk 1. If you specify disk A1, the file is allocated in the first segment (lowest address) on disk A1 that is large enough to contain the file. If not enough space is available on drive A1, the SSP attempts to allocate the file on disk A2.

A2: The preferred file location is on logical disk A2. Although disks 2 through 4 are physically separate, the system treats these disks as one logical disk: A2 is the logical name. If you specify a preferred placement on disk A2 for a multiple-disk configuration, the SSP allocates the file in the last segment (highest address) of available storage on logical disk A2 that is large enough to contain the file. For example:

- On a three-drive system, the file is placed in the last available segment of disk 3.
- On a four-drive system, the file is placed in the last segment of disk 4.

If not enough space is available on the last physical disk of logical disk A2, the SSP attempts to find space on the other disks that make up A2. If not enough space is available on logical disk A2, the SSP attempts to allocate the file on disk A1.

location: The location (block number) of the first block in the file. The following table gives the beginning block number location for disks 2 through 4:

Disk	Beginning Block Number Location
Disk 2	25203
Disk 3	50406
Disk 4	75609

S: The file is a scratch file.

J: The file is a job file.

T: The file is a temporary file.

P: The file is a permanent file.

For information about file retention types, see the description of the FILE statement for disk files in Chapter 1.

key location: The starting byte of the key area within the record for an indexed file. Key location must be a decimal number from 1 through 999, but cannot be greater than the record length. The sum of the key location and the key length must not exceed the length of the record plus 1. If a key location is specified, the key length must also be specified.

key length: The length of the key area within the record for an indexed file. key length must be a decimal number from 1 through 29, but cannot be greater than the record length. A key length must be specified if a key location is specified. The sum of the key location and the key length must not exceed the length of the record plus 1.

Note: key location and key length must be specified when creating an indexed file. key location and key length are ignored when creating a file other than an indexed file.

DFILE: The file is marked as a delete-capable file.

NDFILE: The file is not marked as a delete-capable file.

If the ninth parameter is not specified, NDFILE is assumed.

Note: Deleting a record within a delete-capable file does not result in the physical compression of the file. Instead, the record is marked as a deleted record and is inaccessible to all users of the file.

Examples

Example A

Create a temporary, delete-capable file that is 13 blocks long. The file is INVOICE, the record length is 50 bytes, preferred placement is on the first disk, and each record contains a 4-byte key beginning at byte 9 in the record.

```
BLDFILE INVOICE,I,BLOCKS,13,50,A1,T,9,4,DFILE
```

Example B

Create a permanent, delete-capable file that is 50 records long. The file label is MYFILE, the record length is 25 bytes, preferred placement is block number location 55100 of disk 3, and each record contains a 5-byte key beginning at byte 1 in the record.

```
BLDFILE MYFILE,P,RECORDS,50,25,55100,P,1,5,DFILE
```

BLDLIBR Procedure

Function

The BLDLIBR procedure creates a new user library and, optionally, copies a disk or diskette file containing one or more library members into the new user library.

If the file containing members to be copied is a sector-mode file, it must have been created by the $MAINT utility or by the FROMLIBR procedure.

Note: If the sector-mode file was created on IBM System/32, only non-SCP members are copied.

If the file containing members to be copied is a record-mode file, each member in the file must begin with a COPY statement and end with a CEND statement. For the formats of the COPY and CEND statements, see *Storing Library Members in Disk or Diskette Files* in Chapter 6.

The COPY and CEND statements are automatically inserted into members created by $MAINT. However, *you* must insert them at the beginning and end of members that are not created by $MAINT.

If the record-mode file is organized as a direct file, you must insert an END statement following the CEND statement that terminates the last member in the file. The format of the END statement is:

```
// END
```

where only one blank must separate the // and the END.

The BLDLIBR procedure runs the $MAINT utility program.

Command Statement Format

BLDLIBR library name, blocks, directory sectors, [A1 / A2], [file label],

[I1 / F1], [mmddyy / ddmmyy / yymmdd], [S1 / S2 / S3 / M1.nn / M2.nn], [NOAUTO / AUTO]

Parameters

library name: The name of the new user library. A library name can be up to 8 characters long and must begin with an alphabetic character (A through Z, #, $, or @). The remaining characters can be any combination of characters (numeric, alphabetic, and special) except commas (,), single quotes ('), blanks, question marks (?), slashes (/), and hyphens (-). Do not use F1, READER, PRINT, or DISK as a library name.

blocks: The size, in blocks, of the new user library (including the directory). One block contains 10 sectors. The maximum library size is 65,535 sectors.

directory sectors: The size, in sectors, of the directory for the new library. The minimum directory size is 2 sectors. The maximum directory size is 256 sectors. The first sector of the directory is the library control sector. Each of the remaining sectors can contain nine entries, except for the last sector, which can contain only eight entries.

A1: Indicates a preference that the new library be placed on the first disk. If space is available, the system places the library on the preferred disk. If you specify disk A1, the library is allocated in the first segment (lowest address) on disk 1 that is large enough to contain the library. If not enough space is available on disk A1, the SSP attempts to allocate the library on disk A2.

A2: Indicates a preference that the new library be placed on the second disk. If space is available, the system places the library on the preferred disk. If the fourth parameter is not specified, A2 is assumed. If you specify a preferred placement on disk A2 for a multiple-disk configuration, the SSP allocates the library in the last segment (highest address) of available storage on logical disk A2 that is large enough to contain the library. Although disks 2 through 4 are physically separate, the system treats these disks as one logical disk: A2 is the logical name. For example:

- On a three-drive system, the library is placed in the last available segment of disk 3.
- On a four-drive system, the library is placed in the last segment of disk 4.

If not enough space is available on the last physical disk of logical disk A2, the SSP attempts to find space on the other disks that make up A2. If not enough space is available on logical disk A2, the SSP attempts to allocate the library on disk A1.

file label: The label of the file containing the member(s) to be copied into the new library.

I1: The file containing member(s) to be copied is on diskette. If file label is specified but I1 or F1 is not, I1 is assumed.

F1: The file containing member(s) to be copied is on disk.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file containing member(s) to be copied. The specified date must be in the same format as the session date. If F1 is specified and a date is not specified, the file with the specified label and the most recent creation date is used. If I1 is specified or defaulted to and a date is not specified, the program processes the first file in the diskette VTOC with the specified file label.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If an eighth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, BLDLIBR uses only the specified slot. When the end of a diskette is reached, the SSP displays a message; the operator must insert the next diskette in the specified slot.

- If M1.nn or M2.nn is also specified, BLDLIBR uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine in the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

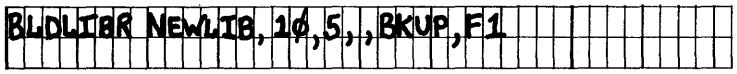
- If S1, S2, or S3 is also specified, BLDLIBR uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, BLDLIBR uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a ninth parameter is not specified, AUTO is assumed.

Example

Create a new library called NEWLIB and copy members from a disk file labeled BKUP into the new library. NEWLIB should be 10 blocks long, the directory should be 5 sectors long, and no disk preference is specified:



BLDMENU Procedure

Function

The BLDMENU procedure creates the library members required to display a menu. A menu allows an operator to start a job by selecting a menu item number instead of entering a control command, a procedure command, or OCL statements. Up to 24 item numbers (1 through 24) can be defined in a menu.

To display a menu, the operator enters the menu name on the Sign On display or uses the MENU control command described in Chapter 3. To terminate a menu display, the operator enters a 0 (zero) instead of an item number.

Input to the BLDMENU procedure is:

- A second-level message source member called a *command source member*. The command source member is required and contains the statements used as system input when the operator selects items from the menu.

- A first-level message member called a *display text source member*. The display text source member is optional for fixed-format menus, but is required for free-format menus. It defines the displayed text that appears on the screen.

More information about the source input members can be found in the descriptions of free-format and fixed-format menus, which follow. The examples at the end of this description provide sample source statements and the correct control statements required to build a menu.

Output from the BLDMENU procedure is:

- A second-level message load member, called a *command load member*, that BLDMENU creates from the command source member. BLDMENU places the command load member in the output library specified on the BLDMENU command statement. This member is one of the two members that must exist to display a menu. The other required member is the screen format load member.

- A first-level message load member, called a *display text load member*, that BLDMENU creates from the optional display text source member. BLDMENU places the display text load member in the output library specified on the BLDMENU command statement. This member is deleted from the library at the end of the job, unless KEEP is specified on the BLDMENU command statement.

- The *screen format load member* for the display. BLDMENU places the screen format load member in the output library specified on the BLDMENU command statement. This member is one of the two members that must exist to display a menu. The other required member is the command load member.

- For a fixed-format menu, a listing that contains the following information for each item number:
  - The corresponding statement from the command source member
  - The corresponding statement from the display text source member
  - The actual display text that will appear on the menu

- For a free-format menu, a listing that contains:
  - The item numbers and the corresponding statements from the command source member
  - The contents of lines 3 through 20 of the menu display

Figure 2-1 summarizes the input to and output from the BLDMENU procedure.

Two types of menus can be created by BLDMENU: fixed-format and free-format. The examples at the end of the procedure description show how to build both types of menus.

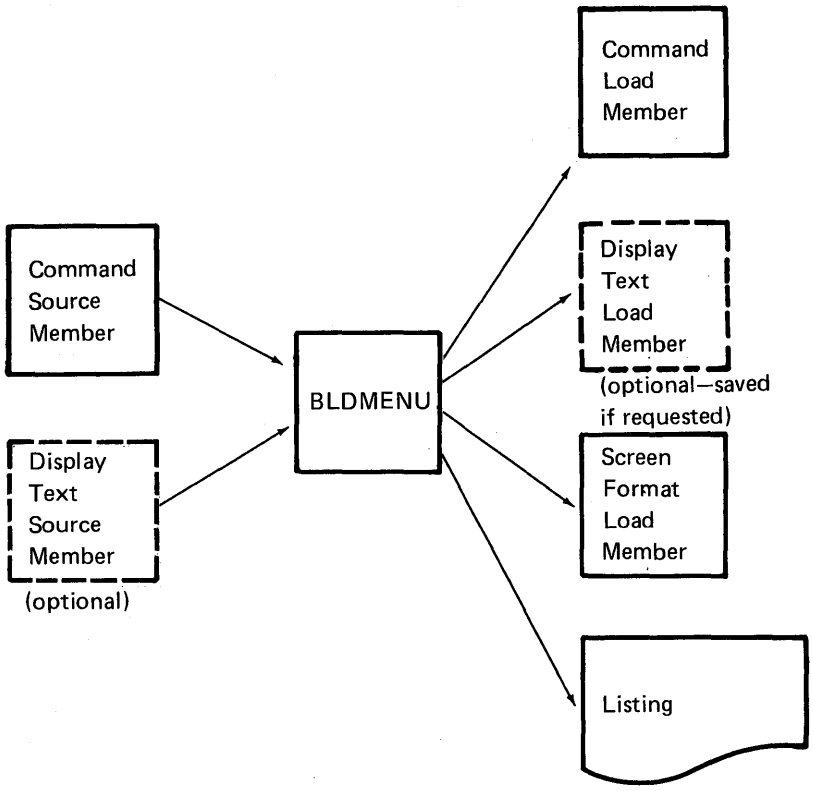


Figure 2-1. Input to and Output from BLDMENU

Fixed-Format Menus

Figure 2-2 shows a typical fixed-format menu. A fixed-format menu always contains two columns of menu items, numbered 1 through 24, with 12 items in each column. For a fixed-format menu, the command source member is required and the display text source member is optional.

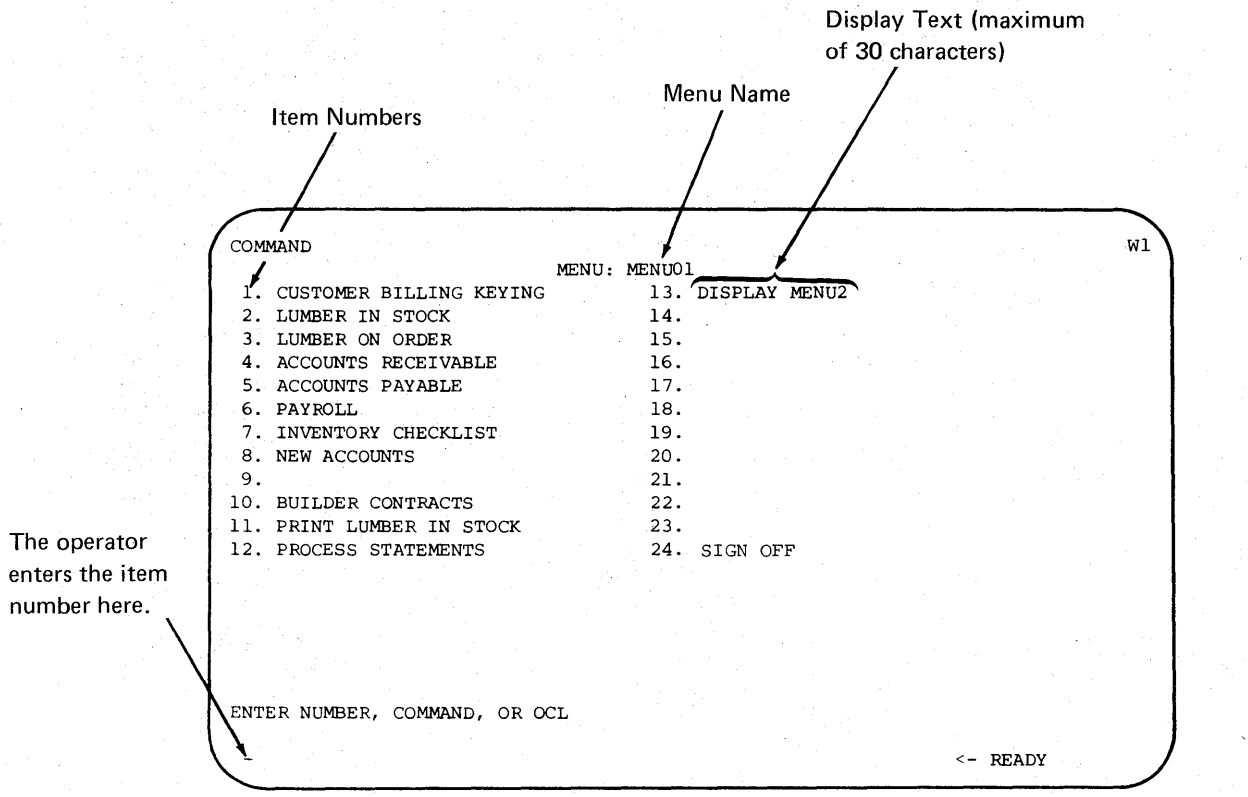
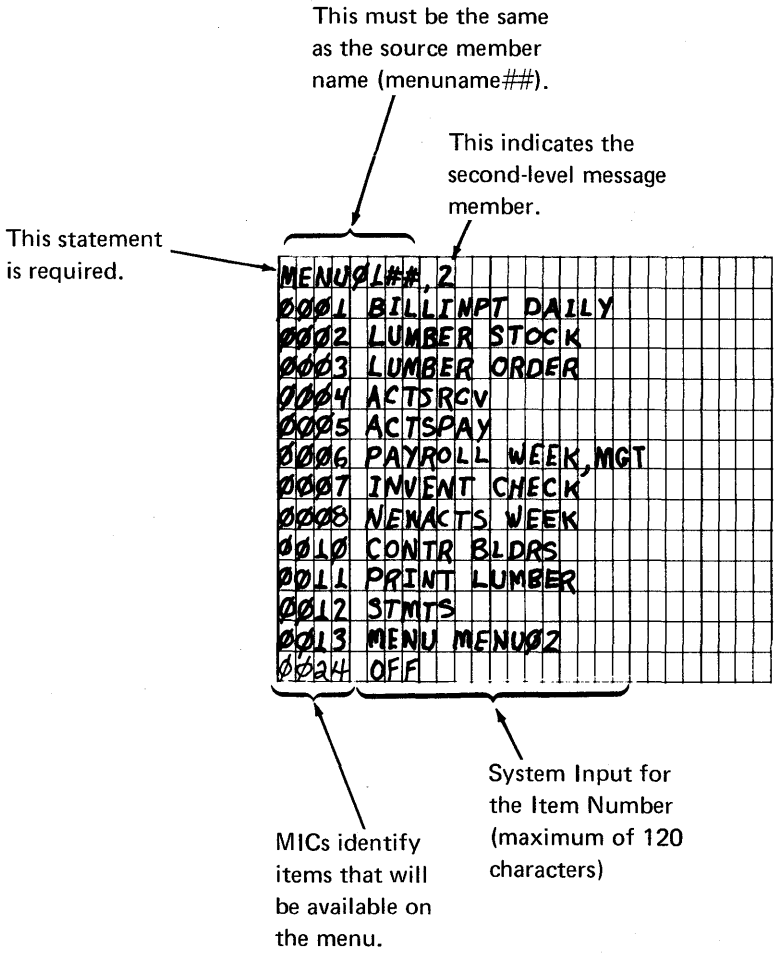
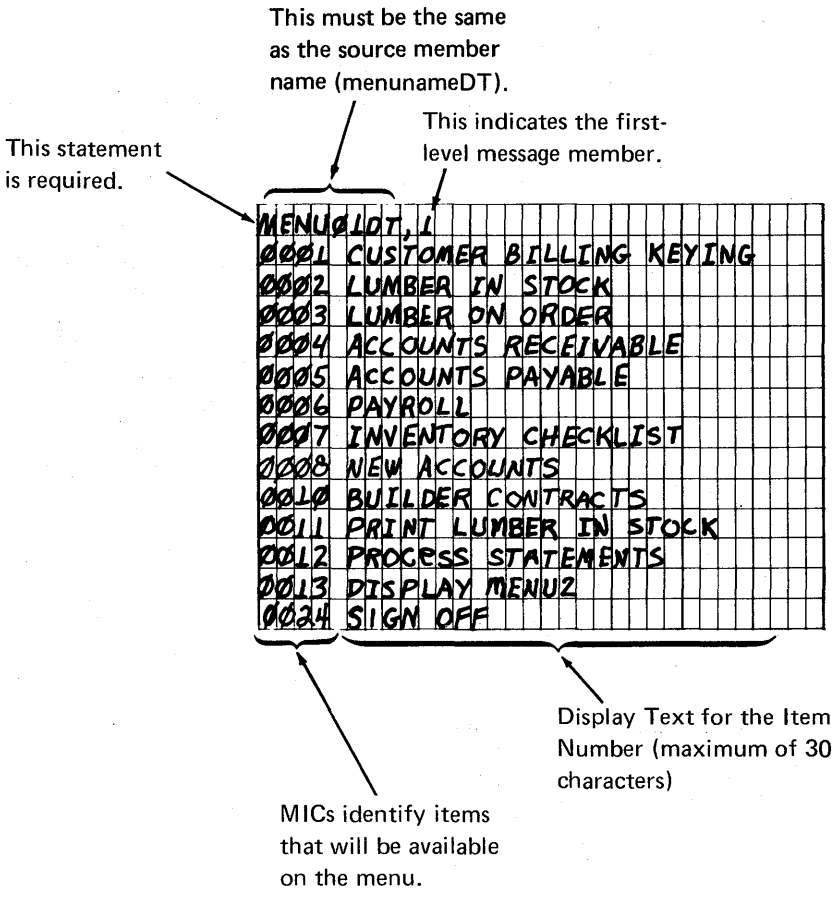


Figure 2-2. Sample Fixed-Format Menu

The name of the *command source member* must be the name of the menu followed by the characters ##. For example, if the menu being created is called MENU3, then the name of the command source member must be MENU3##. Each record in the source member defines a statement to be processed by the system when the operator selects an item from the menu. The MICs (message identification codes) in the member correspond to the menu items that can be selected by the operator. If the member does not contain any MICs from 0001 through 0024 or if all MICs in that range define blank statements, BLDMENU issues an error message and does not build the menu. BLDMENU uses only the first 120 characters of each statement. For the menu shown in Figure 2-2, the command source member could contain the following statements:



Each record in the *display text source member* defines the descriptive text for an item on the menu. The MICs in the display text source member correspond to the menu items that can be selected by the operator. BLDMENU uses only the first 30 characters as the display text. If a display text is not defined for an item that was defined in the command source member, BLDMENU uses the first 30 characters of the record in the command source member as the display text. If a display text source member is not provided as input, BLDMENU uses the first 30 characters of the record in the command source member as the display text for the corresponding menu item. If a display text is defined for an item number that was not defined in the command source member, BLDMENU prints a warning message on the output listing; no text is displayed for the item. For the menu shown in Figure 2-2, the display text source member would contain the following statements:



Free-Format Menus

Figure 2-3 shows a sample free-format menu. When you use BLDMENU to build a free-format menu, you completely define the contents of lines 3 through 20 of the display screen. Both a command source member and a display text source member are required as input to BLDMENU. As with the fixed-format menu, only menu item numbers 1 through 24 can be defined.

If a free-format menu is to be displayed on a 960-character display, lines 15 through 20 will not be displayed. Lines 3 through 8 are displayed in the first half of the menu display, and lines 9 through 14 are displayed in the second half of the menu display.

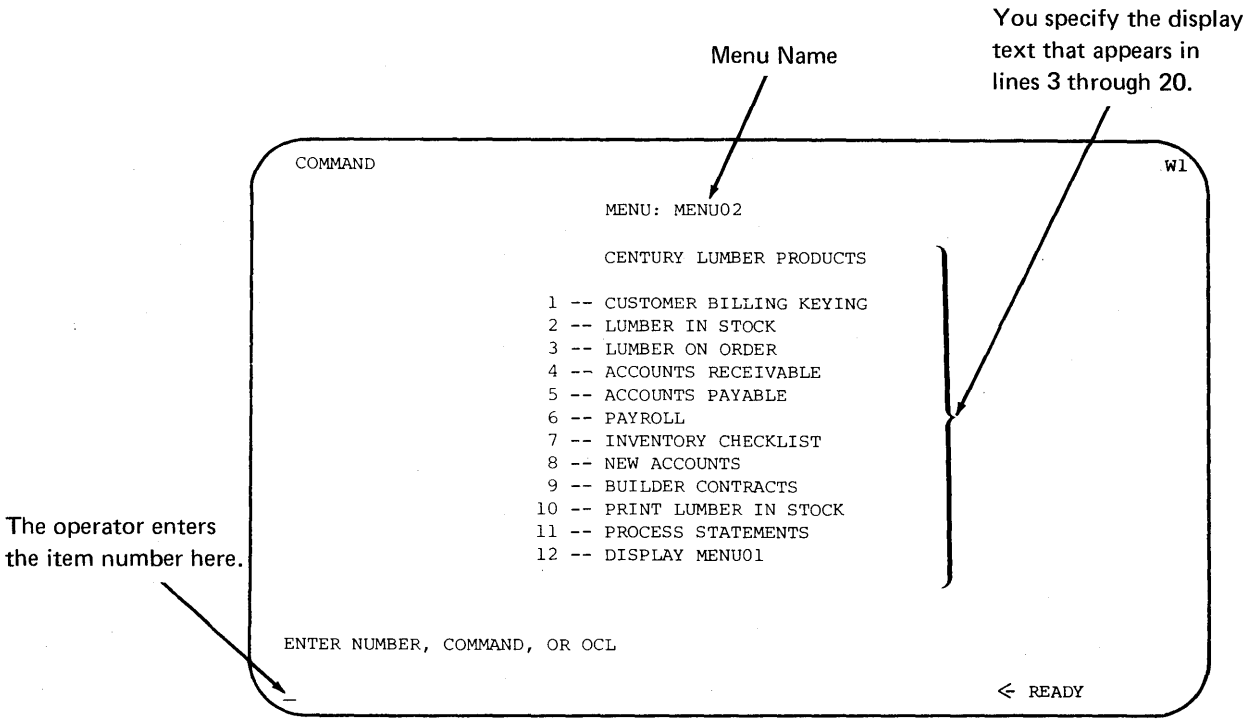
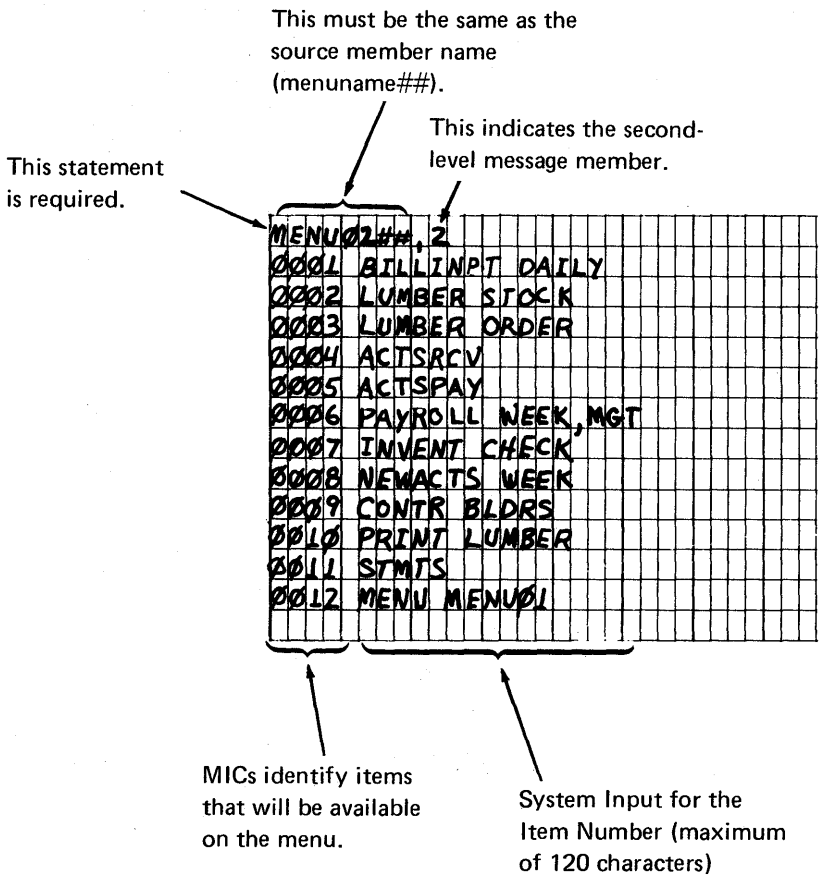
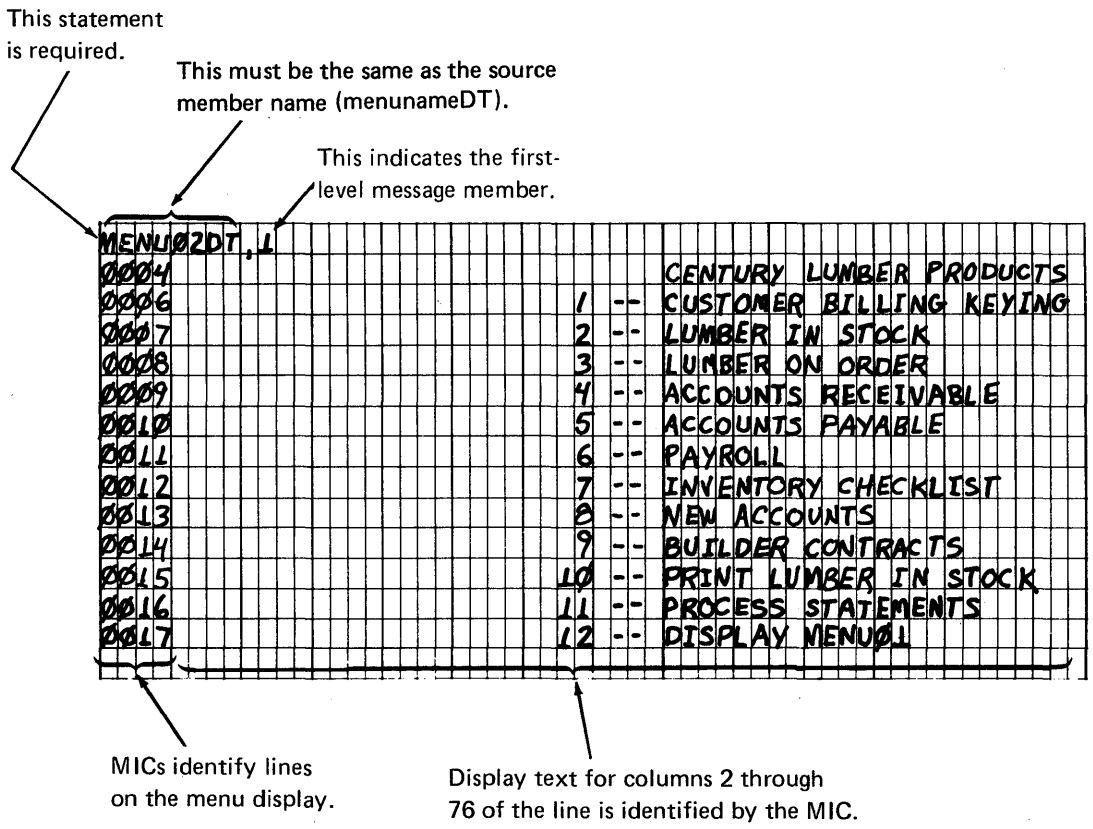


Figure 2-3. Sample Free-Format Menu

Just as for fixed-format menus, each record in the command source member defines the statement to be processed when the menu item identified by the MIC is selected. For the menu shown in Figure 2-3, the command source member would contain the following statements:



Each record in the display text source member for a free-format menu defines the contents of columns 2 through 76 of the display screen line identified by the MIC. For the menu shown in Figure 2-3, the display text source member would contain the following statements:



The BLDMENU procedure requires disk work space to build the screen format load member. If enough space is not available, a diagnostic message is issued.

The BLDMENU procedure runs the $MGBLD, $MAINT, and $BMENU utility programs.

Command Statement Format

BLDMENU menuname, [textname] , [inlib / #LIBRARY] , [outlib / #LIBRARY] ,
[REPLACE] , [KEEP] , [FREEFORM] , [IGC]

Parameters

menuname: The 1- to 6-character name of the menu (the name given to the screen format load member for the menu). The name must be a valid library member name. The names of the command source member and the command load member (the name specified in the first record of the command source member) must be menuname##.

CAUTION
Because ## is concatenated to the menu name by BLDMENU, you will receive an error message (SYS-3807) if parameter 1 (menuname) has more than 6 characters.

textname: The 1- to 8-character name of the display text source member, if one exists. The name must be a valid library member name and cannot be the same as the menu name. The name of the display text load member (the name specified in the first record of the display text source member) must be the same as the name of the source member. Textname is optional only if the FREEFORM parameter is not requested. If textname is not specified, BLDMENU uses the information in the command load member to generate the descriptive text for the items in the fixed-format menu.

inlib: The library that contains the source message member(s). If inlib is not specified, the system library (#LIBRARY) is assumed.

outlib: The library that will contain the screen format load member for the menu, the command load member, and, if KEEP is specified, the display text load member. If outlib is not specified, the system library (#LIBRARY) is assumed.

CAUTION
If the input library and the output library are not the same, BLDMENU copies the input source members to the output library when it begins processing. Consequently, when BLDMENU is run, the output library must have enough unused space to contain the two input source members, as well as the two load members created by BLDMENU. BLDMENU removes the source members from the output library before terminating.

REPLACE: If REPLACE is specified, the following is true for the BLDMENU procedure:

- If a screen format load member already exists in the output library with the same name as the menu being created, BLDMENU automatically deletes the existing member in the output library.

- If a load member that is not a screen format load member already exists in the output library with the same name as the menu being created, BLDMENU displays an error message. The operator must then decide whether to replace the existing member in the output library or to cancel the job.

- If the name of the command source member or the display text source member is the same as the name of a source member or load member in the output library, BLDMENU automatically replaces the existing member(s) in the output library.

If REPLACE is not specified, the following is true for the BLDMENU procedure:

- If a load member already exists in the output library with the same name as the menu being created, BLDMENU displays an error message. The operator must then decide whether to delete the existing member in the output library or to cancel the job.

- If the name of the command source member or the display text source member is the same as the name of a source member in the output library, BLDMENU displays an error message and cancels the job.

CAUTION
If a menu is rebuilt while it is being displayed at a display station, the display station operator should request the rebuilt menu by entering a MENU control command. If this is not done, the old version of the menu is displayed, but it may not correspond to the command input defined for the new menu.

KEEP: If both KEEP and textname are specified, the display text load member created by BLDMENU remains in the output library. If KEEP is not specified, the display text load member is deleted from the output library before BLDMENU terminates.

Note: The display text load member has no function other than to give the user an explanation of the function of the command.

FREEFORM: If FREEFORM is specified, a free-format menu is created. If FREEFORM is not specified, a fixed-format menu is created.

IGC: The IGC parameter is for the ideographic version of the SSP and is ignored for nonideographic systems. IGC specifies that the system-generated text (for example, the ENTER NUMBER, COMMAND, OR OCL prompt) should be displayed as ideographic characters and ideographic data can be entered into the input field of the menu screen.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

CAUTION
If a menu is built using the IGC parameter, the menu cannot be displayed from a nonideographic terminal.

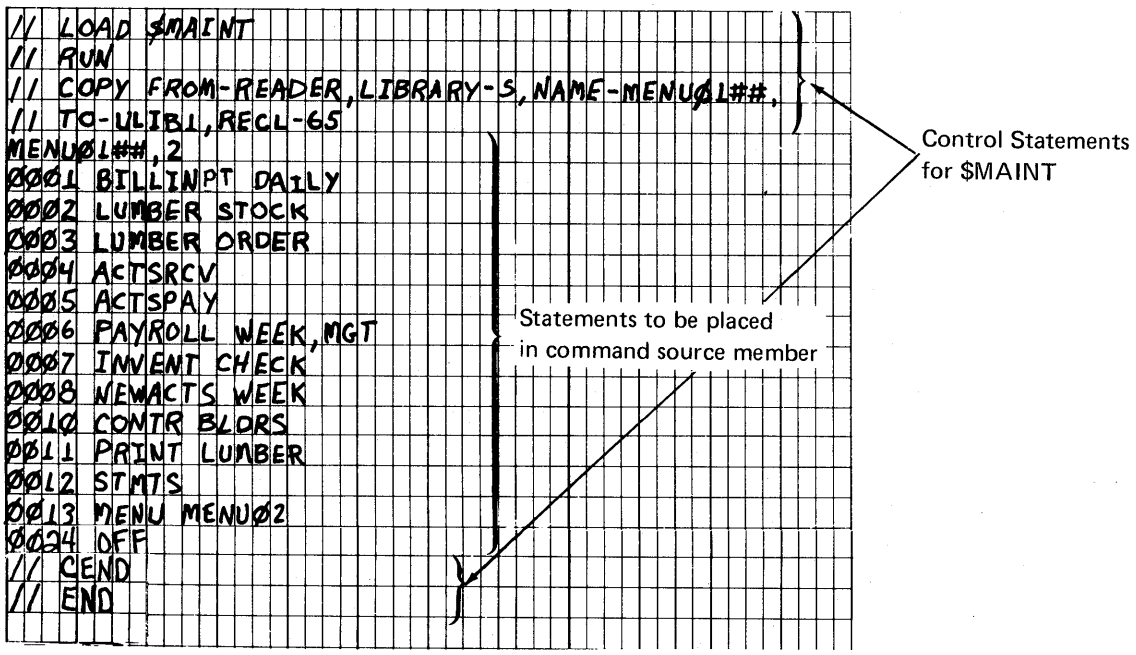
Examples

Example A

Build the fixed-format menu (MENU01) shown in Figure 2-2.

To create MENU01, you must perform three steps:

Step 1: Use the $MAINT utility program or the SEU (source entry utility) portion of the Utilities Program Product to create the command source member (the member that specifies the statement to be executed for each item number). Remember that the member name must be the menu name followed by ##. Therefore, the command source member for this example must be called MENU01##. For this example, place MENU01## in a user library called ULIB1. If you use the $MAINT utility to create the member, enter the following statements:



Step 2: Use the $MAINT utility or SEU to create the display text source member (the member that specifies the text to be displayed along with each item number). For this example, the display text source member is called MENU01DT and is placed in a user library called ULIB1. If you use the $MAINT utility to create MENU01DT, enter the following statements:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-S,NAME-MENU01DT,
//      TO-ULIB1,RECL-80
MENU01DT,1
0001 CUSTOMER BILLING KEYING
0002 LUMBER IN STOCK
0003 LUMBER ON ORDER
0004 ACCOUNTS RECEIVABLE
0005 ACCOUNTS PAYABLE
0006 PAYROLL
0007 INVENTORY CHECKLIST
0008 NEW ACCOUNTS
0010 BUILDER CONTRACTS
0011 PRINT LUMBER IN STOCK
0012 PROCESS STATEMENTS
0013 DISPLAY MENU2
0024 SIGN OFF
// CEND
// END
```

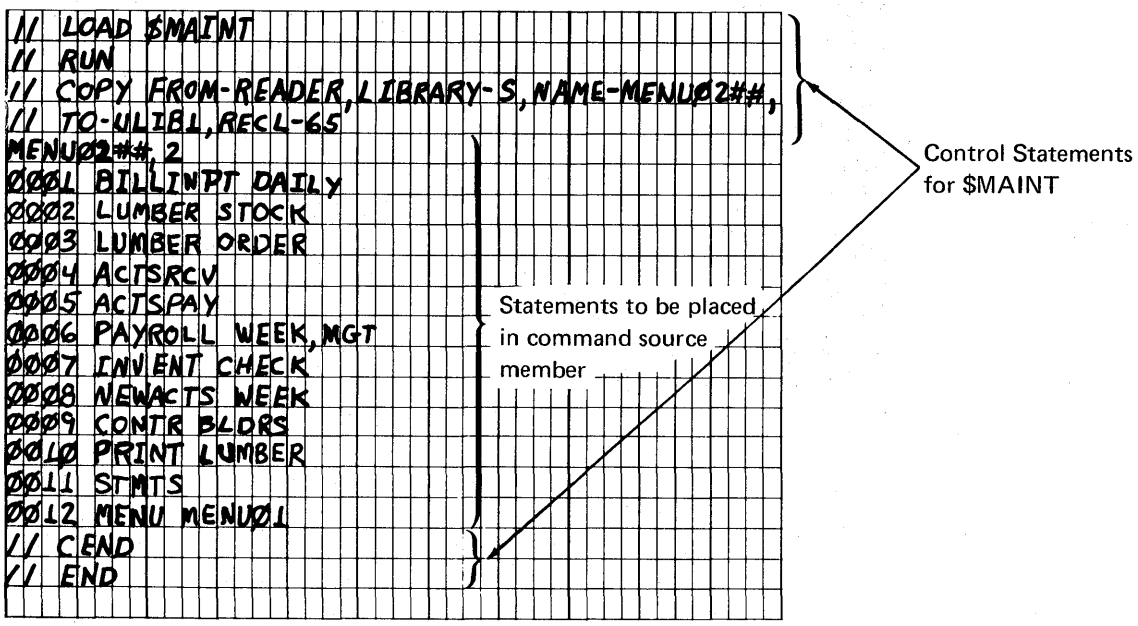
Step 3: Run the BLDMENU procedure. The library members created by BLDMENU are to be placed in a user library called ULIB2, and the display text load member created by BLDMENU is to be deleted from ULIB2 before BLDMENU terminates. The following procedure command is used:

```
BLDMENU MENU01,MENU01DT,ULIB1,ULIB2
```

Example B

Build the free-format menu (MENU02) shown in Figure 2-3. To create MENU02, you must perform three steps:

Step 1: Use the $MAINT utility program or the SEU (source entry utility) portion of the Utilities Program Product to create the command source member (the member that specifies the statement to be executed for each item number). Remember that the member name must be the menu name followed by ##. Therefore, the command source member for this example must be called MENU02##. For this example, place MENU02## in a user library called ULIB1. If you use the $MAINT utility to create the member, enter the following statements:



Step 2: Use the $MAINT utility program or SEU to create the display text source member (for a free-format menu, the member that defines the contents of lines 3 through 20 on the menu display). For this example, the display text source member is called MENU02DT and is placed in a user library called ULIB1. If you use the $MAINT utility to create MENU02DT, enter the following statements:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-S,NAME-MENU02DT,
//    TO-ULIB1,RECL-80
MENU02DT,1
0004                          CENTURY LUMBER PRODUCTS
0006                    1  -- CUSTOMER BILLING KEYING
0007                    2  -- LUMBER IN STOCK
0008                    3  -- LUMBER ON ORDER
0009                    4  -- ACCOUNTS RECEIVABLE
0010                    5  -- ACCOUNTS PAYABLE
0011                    6  -- PAYROLL
0012                    7  -- INVENTORY CHECKLIST
0013                    8  -- NEW ACCOUNTS
0014                    9  -- BUILDER CONTRACTS
0015                   10  -- PRINT LUMBER IN STOCK
0016                   11  -- PROCESS STATEMENTS
0017                   12  -- DISPLAY MENU01
// CEND
// END
```

Step 3: Run the BLDMENU procedure. For this example, the library members created by BLDMENU are to be placed in a user library called ULIB2, and the display text load member created by BLDMENU is to be deleted from ULIB2 before BLDMENU terminates. The following procedure command is used:

```
BLDMENU MENU02,MENU02DT,ULIB1,ULIB2,,,FREEFORM
```

BUILD Procedure

Function

The BUILD procedure allows you to display and correct data on the disk after a disk error occurs. BUILD can be run only from the system console and only when no other programs are running.

When a disk read or write error occurs, the data is written to an *alternative sector*. Alternative sectors are reserved for use in place of defective disk sectors. The BUILD procedure searches the alternative sectors of the disk for data that was unreadable because of a read error. Each sector containing unreadable data is printed, along with the sector logically preceding and the sector logically following the sector containing unreadable data in the file. The operator can then leave the data as it is or correct the data.

For an example of information listed and displayed by BUILD and for information about how the operator should correct the bad data, see the description of the $BUILD utility program in Chapter 4.

The BUILD procedure runs the $BUILD utility program.

Command Statement Format

BUILD

Parameters

None

CATALOG Procedure

Function

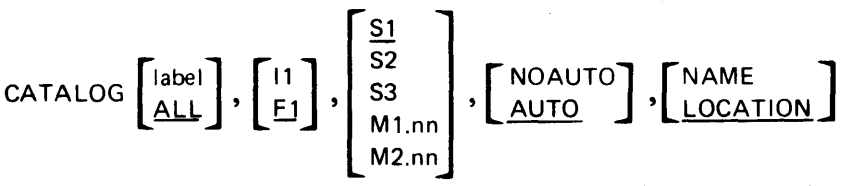
The CATALOG procedure lists, on the system list device assigned to the requesting display station, either the disk VTOC, a diskette VTOC, or a VTOC entry. The disk VTOC contains an entry for each file on the disk, and a diskette VTOC contains an entry for each file on the diskette. Each VTOC entry identifies the related file by label, creation date, and location.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

For information about the VTOC display, see the description of the $LABEL utility program in Chapter 4.

The CATALOG procedure runs the $LABEL utility program.

Command Statement Format



Note: Parameters 3 and 4 are ignored if the system does not have a diskette magazine drive or if F1 is specified for parameter 2.

Parameters

label: The label of the file whose VTOC information is to be displayed. If more than one file exists with the specified label, the VTOC information for all files with the specified label.

ALL: Display all labels in the specified VTOC. If the first parameter is not specified, ALL is assumed.

I1: Display the diskette VTOC entry or entries.

F1: Display the disk VTOC entry or entries. If the second parameter is not specified, F1 is assumed.

S1, S2, or S3: Identifies the first individual diskette slot to be used by CATALOG. If a third parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the first magazine location to be used by CATALOG. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is specified as the third parameter, the CATALOG procedure uses only the specified slot. For CATALOG ALL, CATALOG ends when the diskette in the specified slot is cataloged.
- If M1.nn or M2.nn is specified as the third parameter, CATALOG starts with the specified location and ends after going through the last position of the specified magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is specified as the third parameter, CATALOG starts with the specified slot and ends after cataloging the diskette in slot S3.
- If M1.nn or M2.nn is specified as the third parameter, CATALOG starts with the specified location and ends after cataloging the diskette in M2.10.

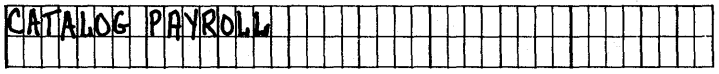
  If a fourth parameter is not specified, AUTO is assumed.

LOCATION: Specifies the VTOC entries are to be sorted and displayed by physical file location sequence. If the fifth parameter is not specified, LOCATION is assumed.

NAME: Specifies the VTOC entries are to be sorted and displayed by filename in alphabetical ascending sequence.

Example

Display VTOC information for the disk file called PAYROLL:



COMPRESS Procedure

Function

The COMPRESS procedure causes all free space within the user area on disk to be accumulated into a single area. For a single-disk configuration, free space is accumulated at the highest block numbers on the disk. For a multiple-disk configuration, the free space on both disks is accumulated at the disk juncture; that is, free space on disk A1 is accumulated at the highest block numbers on disk A1, and free space on disk A2 is accumulated at the lowest block numbers on disk A2.

The COMPRESS procedure can be run only from the system console and cannot be run while (1) any other jobs are being run, (2) a user is signed on at any other display station, (3) remote work stations are varied online, (4) SSP-ICF subsystems are enabled, (5) a communications line is enabled, or (6) while extended trace is active.

The COMPRESS procedure runs the $PACK utility program.

Notes:

1. If LOCATION was specified in the FILE statement for a file moved by the COMPRESS procedure, the location specified is not valid after the COMPRESS procedure moves the file. Use the CATALOG procedure to display the VTOC entries for files moved by the COMPRESS procedure to determine the new locations of the files.
2. If you want to: (1) accumulate free space on only one disk for a multiple-disk configuration, or (2) accumulate free space immediately following the system library and/or accumulate free space in the highest block numbers on disk A2, you can use the $FREE utility program described in Chapter 4.

Command Statement Format

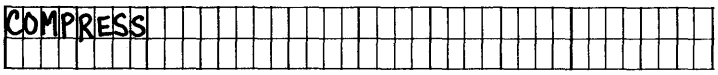
COMPRESS

Parameters

None

Example

Compress the free space on disk:



CONDENSE Procedure

Function

The CONDENSE procedure moves all members in a specified library to the front (low block number) of the library. All unused space is accumulated at the end of the library. No other jobs or display stations can use the specified library when the CONDENSE procedure is running. If an attempt is made to run the CONDENSE procedure for a library that is being used by another job, an error message is displayed and the CONDENSE procedure is not run. Once the CONDENSE procedure begins running, the system does not allow other jobs to use the specified library until the CONDENSE procedure is complete. If the specified library is the system library, the CONDENSE procedure cannot be run while (1) any other jobs are being run, (2) a user is signed on at any other display station, (3) remote work stations are varied online, (4) SSP-ICF subsystems are enabled, or (5) a communications line is enabled.

The CONDENSE procedure runs the $MAINT utility program.

Command Statement Format

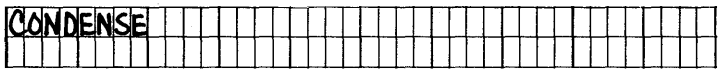
CONDENSE [library name / #LIBRARY]

Parameters

library name: The name of the library to be condensed. If a library name is not specified, the system library (#LIBRARY) is assumed.

Example

Condense the members in the system library:



COPYI1 Procedure

Function

The COPYI1 procedure causes all files on a single diskette or an individual file on a single diskette to be copied to another diskette.

For systems without a diskette magazine drive, the system reads the data to be copied and then instructs the system operator to insert the receiving diskette into the diskette drive.

For systems with a diskette magazine drive, the diskette being copied is placed in slot S1, and the diskette to receive the copy is placed in slot S2. Also, for systems with a diskette magazine drive, the COPYI1 procedure can be used to copy the diskettes in magazine slot M1 onto the diskettes in magazine slot M2.

A work space large enough to contain the file(s) to be copied must be available on the disk. Files, except multivolume files, copied from a diskette are placed contiguously on the receiving diskette. For multivolume diskette files, each file segment occupies the same area on both the copied diskette and the receiving diskette.

Note: The COPYI1 procedure will copy only files that meet the following requirements:

- The file was created on System/34 or the operating system contains blanks.
- The block length is the same as the sector length.
- The record attributes are specified as unblocked and unspanned, or blocked and spanned.
- The output diskette is in the same format as the diskette being copied.

See Appendix C for information about diskette formats and record attributes.

The COPYI1 procedure can be used to create a backup diskette file or to gather all unused space on an input diskette into a single free space on the output diskette.

Diskettes with important permanent files are the diskettes normally copied. Because diskettes can develop surface irregularities as they undergo the wear of continued use, it is a good idea to copy important files soon after they are created.

If the COPYI1 procedure encounters a read error on the input diskette, the COPYI1 procedure displays the bad sector. The operator can then correct the information in the sector or copy it as it is to the output diskette. (If the error was in a load member that was copied from a library or if the error was in file control information used by the system, the file may not be usable even if the operator chooses to bypass the error.) The operator can use the following function and command keys:

Key	Function
Roll Up	Scans right.
Roll Down	Scans left.
Enter/Rec Adv	Updates the sector buffer with the changes keyed by the operator. (The Enter/Rec Adv key has no effect on the data being copied if a sector other than the bad sector is displayed.)
Cmd key 1	Displays the next sector.
Cmd key 2	Displays the preceding sector.
Cmd key 3	If pressed while the bad sector is displayed, causes the sector buffer to be copied to the output diskette. If pressed while a sector other than the bad sector is displayed, causes the bad sector to be displayed.

The COPYI1 procedure runs the $DUPRD utility program.

Command Statement Format

```
COPYI1 [label / ALL] , [mmddyy / ddmmyy / yymmdd] ,vol-id, [DELETE] , [PRESERVE] ,

       [nn / 1] , [M1.01 / S1]
```

Parameters

label: The label of a single file to be copied to another diskette. Label is not allowed if the seventh parameter is M1.01 or M1.

ALL: Copy all files on an individual diskette to another diskette or, if the seventh parameter is M1.01 or M1, copy the contents of the diskettes in magazine slot M1 onto the diskettes in magazine slot M2. If a value is not specified in the first parameter position, ALL is assumed.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file to be copied. The specified date is used to verify that the correct file is on the input diskette. The specified date must be in the same format as the creation date. The output file created by the COPYI1 procedure has the same creation date. If ALL is specified in the first parameter position, a creation date cannot be specified.

vol-id: The volume ID of the output diskette(s); 1 to 6 alphameric characters.

Note: If M1.01 or M1 is specified, vol-id must be the volume ID of all of the diskettes in magazine M2.

DELETE: Do not copy expired files. DELETE can be specified only when ALL is specified.

PRESERVE: Copy the entire file extent (the space from the beginning of the file to the end of the file.) If PRESERVE is not specified, only sectors containing data are copied, and the end of the file immediately follows the last copied sector.

nn: The number of copies to be made. nn can be any number from 1 through 99. If a value is not specified in the sixth parameter position, 1 is assumed.

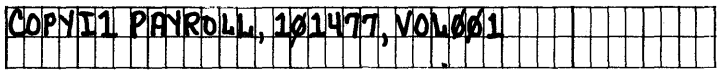
M1.01: Copy the contents of the diskettes in magazine M1 onto the diskettes in magazine M2. Specifying M1 is the same as specifying M1.01.

S1: Copy data from the diskette in slot S1 onto the diskette in slot S2. If a value is not specified in the seventh parameter position, S1 is assumed. For systems without a diskette magazine drive, the seventh parameter is ignored.

Examples

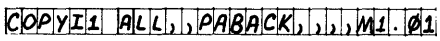
Example A

Copy a diskette file labeled PAYROLL (dated October 14, 1977) onto a diskette with a volume ID of VOL001:



Example B

PA1, a file that occupies four diskettes, is to be copied to four other diskettes, each of which has a volume ID of PABACK. The four diskettes to be copied are in the first four locations of the magazine in slot M1, and the diskettes that will receive the copy are in the first four locations of the magazine in slot M2. The following COPYI1 statement will copy the file:



COPYPRT Procedure

Function

The COPYPRT procedure copies spool file entries from the spool file to a disk file. Selected spool file entries can be displayed.

The COPYPRT procedure runs the $UASF and/or $UASC utility program(s).

Command Statement Format

```
COPYPRT [ALL     ] , [file label] , [CANCEL ] , [CRT]
        [SYSTEM  ]                  [RELEASE]
        [spool-id]
        [Fxxxx   ]
        [NOCOPY  ]
```

Parameters

ALL: Copy all spool file entries, not being processed by spool, that have user IDs that match the user ID of the display station operator. If the first parameter is not specified, ALL is assumed.

SYSTEM: If security is not active, copy all spool file entries not being processed by spool. If security is active, the operator must have a security classification of system operator or higher to copy all spool file entries.

spool-id: The 6-character spool file entry ID of the spool file entry to be copied. If security is not active, the display station operator can copy any spool file entry. If security is active, the user ID associated with the spool file entry must be the same as the user ID of the display station operator, unless the operator has a security classification of system operator or higher.

Fxxx: xxxx is the 1- to 4-character forms ID of the entry or entries to be copied. (For more information about specifying forms numbers, see the CHANGE PRT command.) If security is not active, the display station operator can copy any entry or entries with the specified forms number. If security is active, the user ID associated with the spool file entry or entries must be the same as the user ID of the display station operator, unless the operator has a security classification of system operator or higher. Only entries not being processed by spool can be copied.

NOCOPY: Display a disk file that contains a spool entry or entries previously copied from spool via COPYPRT. If NOCOPY is specified, RELEASE or CANCEL is invalid and parameter 4 defaults to CRT.

file label: The label of the file that will contain the copied spool file entries. If the second parameter is not specified and if a spool-id or forms number is specified, the file label defaults to the spool-id or fxxxx concatenated with the work station ID of the requesting display station. A file label must be specified if ALL, SYSTEM, or NOCOPY is specified as the first parameter or if the first parameter is not specified.

RELEASE: Upon completion of the copy to disk file, the entry or entries specified in parameter 1 are released.

Note: If the entire spool file is held, an individual entry can be released but it will not print until the entire spool file is released.

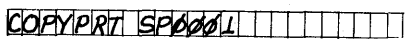
CANCEL: Upon completion of the copy to disk file, the entry or entries specified in parameter 1 are canceled from the spool file.

CRT: Display entry or entries on user's display station.

Examples

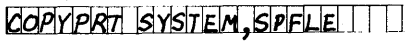
Example A

Copy spool file entry SP0001 into a file labeled SP0001W2.



Example B

Copy all spool file entries into a file labeled SPFLE. The COPYPRT procedure is requested by a user authorized to copy entries belonging to other users.



Example C

Copy all spool file entries with forms number 0017 into a file labeled F0017W3. The COPYPRT procedure is requested from display station W3. Only entries with user IDs matching that of the operator signed on at W3 are copied.

Example D

Copy all spool file entries into a file labeled ALL1, cancel the entries from the spool file, and display the entries on the user's display station. The COPYPRT procedure is requested from display station X2. Only entries with user IDs matching that of the operator signed on at X2 are copied.

```
COPYPRT ALL,ALL1,CANCEL,CRT
```

After the COPYPRT procedure command is entered, the following header selection screen is displayed at display station X2:

```
                              **COMPLETE**
NO.  I/D      PROC      JOBNAME    USER      PRINTER    ID  FORM  PAGES  RECS

001 SP0002              X2083838   LGA       PRINTKEY   P3  0001  1      33
002 SP0010              X2095242   LGA       PRINTKEY   P3  0001  1      33
003 SP0012  BASIC       X2095349   LGA       BASIC255   P3  0001  1      47

-

   SELECTED HEADER -                 PRINT(Y/N) -   N         COPIES -   01
   FROM PAGE            TO PAGE                  ENTER/HELP KEY CMD 7  - END
```

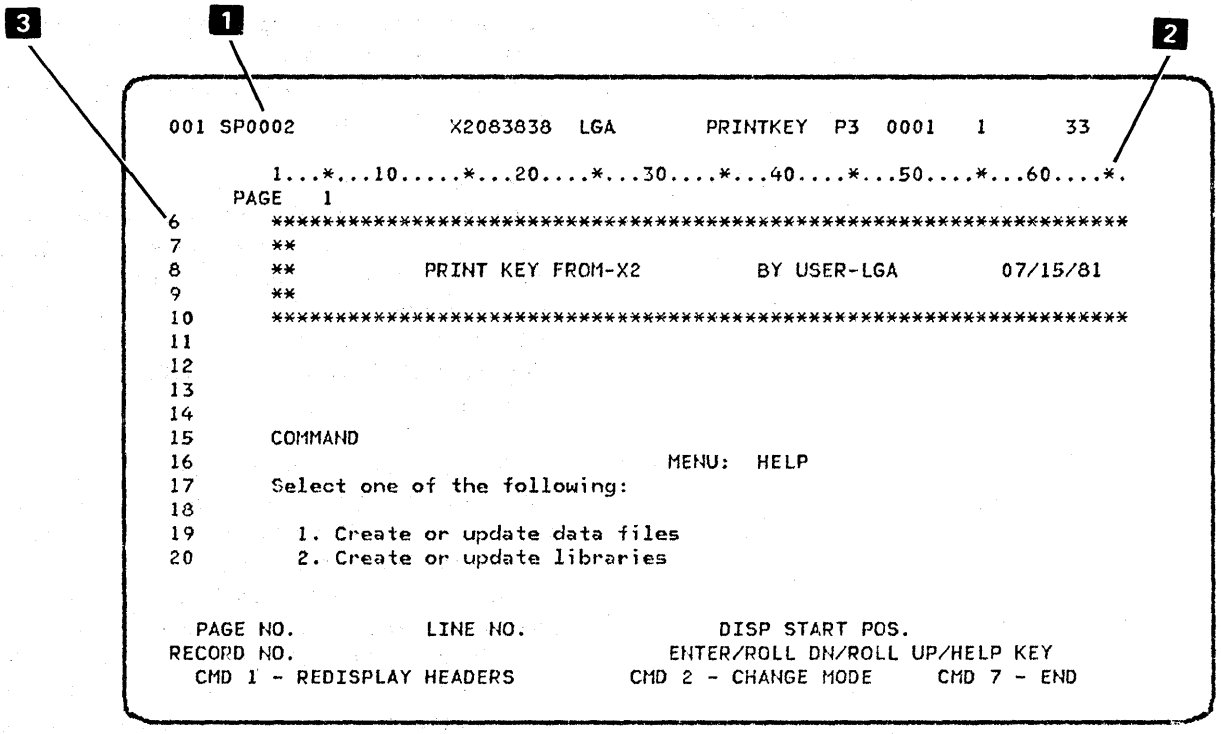
The header selection screen contains the following information about the spool file entries copied into the COPYPRT output file (for Example D, file ALL1):

Item	Description
COMPLETE	Indicates that all the copied spool file entries have been displayed. Additional entries, if any, can be displayed using the Roll Up or Roll Down key.
NO.	The relative header number for each of the spool file entries copied into the COPYPRT output file (in this example, ALL1).
I/D	The spool ID of the spool file entry copied to the COPYPRT output file.
PROC	The procedure name associated with the spool ID.
JOBNAME	The jobname assigned by the system (work station ID and time).
USER	The user ID associated with this entry (in Example D, user LGA).
PRINTER	The printer file name associated with this entry.
ID	The printer ID of the printer to which the output is routed.
FORM	The form number assigned to this entry.
PAGES	The total number of pages of spooled output generated by this entry.
RECS	The number of records the spool file entry occupies in the COPYPRT output file. Each print line of spooled output requires one record.

The following fields of the header selection screen allow operator action:

Field	Description
SELECTED HEADER	To display a specified spool file entry that has been copied into the COPYPRT output file, enter the relative header number (see NO.) in this field. For Example D, to view spool file entry SP0010, enter 2 in this field. If only one spool file entry is copied, this value defaults to the header of that entry (1).
PRINT(Y/N)	To print a copy of the spool file entry identified by the header number entered in the SELECTED HEADER field, enter Y (yes) in this field. The default value is N (no). The copy of the selected spool file entry is printed on the session printer.
COPIES	If Y is specified in the PRINT(Y/N) field, enter the number of copies of the selected entry to be printed. The default value is 1.
FROM PAGE	If Y is specified in the PRINT(Y/N) field, enter the number of the first page to be printed. For example, if an entry contains 45 pages, but you want to print the last 20 pages only, enter 26 in this field. The default value for FROM PAGE is 1 (the first page of the entry).
TO PAGE	If Y is specified in the PRINT(Y/N) field, enter the number of the last page to be printed. For example, if an entry contains 30 pages, but you want to print the first 10 pages only, enter 10 in this field. To print pages 5 through 25, enter 5 in the FROM PAGE field and 25 in the TO PAGE field. The default value for TO PAGE is the page number of the last page of the entry.

For Example D, to view the spooled output for spool ID SP0002, enter the associated header number (1). The following screen is displayed:



This screen is in page mode and displays a portion of the first page of spooled output for the selected spool file entry:

1 The first line of the page mode screen contains the header selection information for the selected entry.

2 A row of numbers on the third line of the screen shows the column positions of the spooled output within the records that make up the displayed entry.

3 A column of numbers along the left side of the screen shows the relative print line numbers for each page of the spooled output. Each print line requires one record of the COPYPRT output file.

The following fields of the page mode screen allow operator action:

Field	Decription
PAGE NO.	To display a specified page of the selected entry, enter the page number in this field. For example, to display page 21 of a 25-page document, enter 21 in the PAGE NO. field.
LINE NO.	To display a specified print line of the currently displayed page, enter the relative print line number in this field. Print line numbers are shown along the left side of the display. For example, to display print line 30 of the currently displayed page of spooled output, enter 30 in the LINE NO. field. To display print line 25 of page 20, enter 20 in the PAGE NO. field and 25 in the LINE NO. field.
DISP START POS.	To display a specified column position and the columns following it, enter the column position in this field. Column positions are shown on the third line of the display. For example, to display columns 80 through 120, enter 80 in this field. To display print line number 20 of page 10, starting in column 50, enter 10 in the PAGE NO. field, 20 in the LINE NO. field, and 50 in the DISP START POS. field. Each starting column position you enter remains in effect until a new starting column position is entered in the DISP START POS. field.
RECORD NO.	To display a specified record number and the records following it, enter the record number in this field. Each print line occupies one record in the COPYPRT output file. To roll up or roll down the display a specific number of records, key in the number of records to roll in the RECORD NO. field, and then use the Roll Up or Roll Down key. The displayed page will roll the selected number of records one time. *Note:* To roll the display a specified number of records, do not press the Enter/Rec Adv key before using the roll keys.

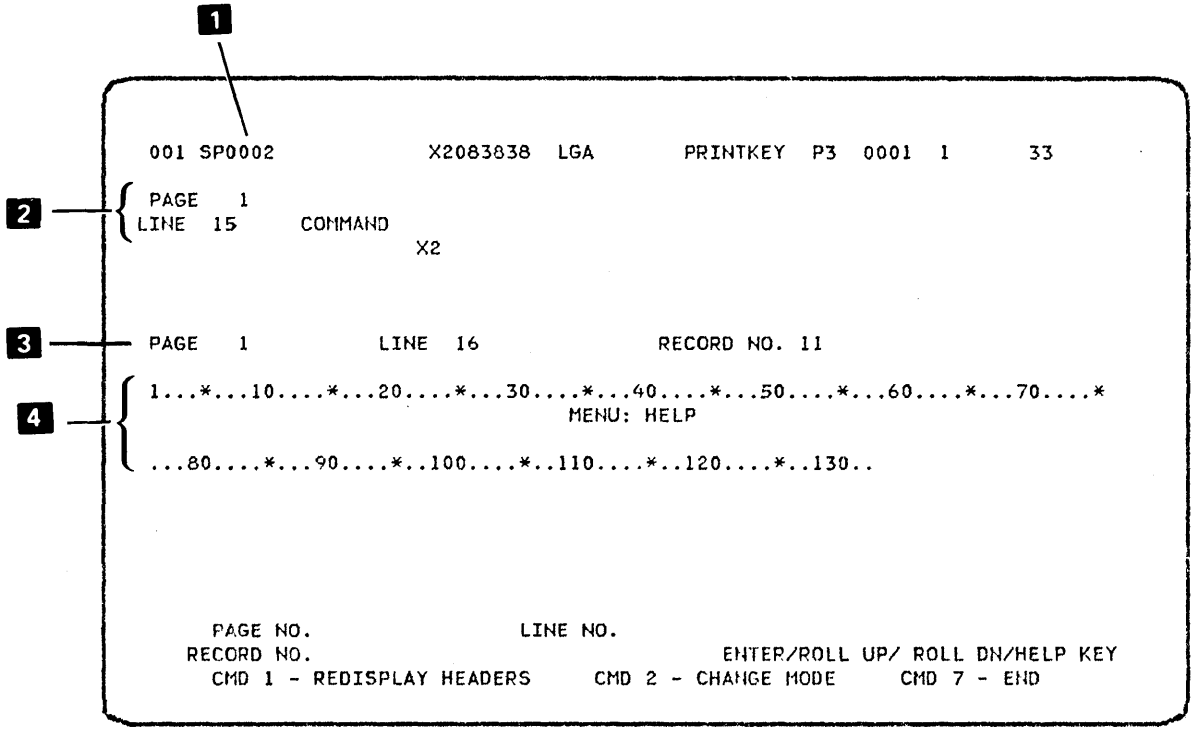
For Example D, to display print line number 16 starting at column position 30, enter 16 in the LINE NO. field, and 30 in the DISP START POS. field. The following screen is displayed:

```
001 SP0002            X2083838  LGA       PRINTKEY  P3  0001   1      33

       0...*...40.....*...50....*...60....*...70....*...80....*...90....*
    PAGE   1
16          MENU:  HELP
17
18
19      iles
20      ies
21      ions
22      rations
23      y
24       to P3 and 66 lines per page
25      to the display station          SYSLIST CRT
26      to the printer                  SYSLIST PRINTER
27      to off                          SYSLIST OFF
28      cility
29
30


  PAGE NO.          LINE NO.              DISP START POS.
RECORD NO.                             ENTER/ROLL DN/ROLL UP/HELP KEY
  CMD 1 - REDISPLAY HEADERS         CMD 2 - CHANGE MODE     CMD 7 - END
```

The following command and function keys are enabled on the page mode screen:

Key	Function
Cmd key 1	Redisplays the header selection screen. For example, after displaying spool file entry SP0010 (header number 2), use command key 1 to return to the header selection screen to select a new entry.
Cmd key 2	Displays the selected entry record by record. When in record mode, command key 2 can be used to return to page mode.
Cmd key 7	Ends the COPYPRT procedure.
Roll Up/Roll Down	Rolls up or rolls down the currently displayed page. To roll up or roll down the display a specific number of records, key in the number of records to roll in the RECORD NO. field, and then use the Roll Up or Roll Down key. The displayed page will roll the selected number of records one time. *Note:* To roll the display a specified number of records, do not press the Enter/Rec Adv key before using the roll keys.

To display the currently selected entry in record mode, use command key 2. For Example D, the following record mode screen is displayed:



1. The second line of the record mode screen contains information about the selected entry.

2. Lines 4 through 6 display the record that precedes the selected record.

3. Line 10 lists the page number, the line number, and the record number of the selected record.

4. Lines 12 through 16 display the currently selected record, in addition to the record column positions.

The PAGE NO., LINE NO., and RECORD NO. fields are used to select a particular page number, line number, or record number.

The Roll Up and Roll Down keys are used to move forward and backward through the records of the currently displayed entry. Cmd key 1 is used to return to the header selection screen, and Cmd key 2 is used to return to the page mode screen. Cmd key 7 is used to end the COPYPRT procedure.

CREATE Procedure

Function

The CREATE procedure creates a message load member from a message source member. A message load member contains messages that can be retrieved by user programs or IBM programs.

For information about creating a message source member, see the description of the $MGBLD utility program in Chapter 4.

The CREATE procedure runs the $MGBLD utility program.

Command Statement Format

CREATE sourcename, [REPLACE] , [library name / #LIBRARY]

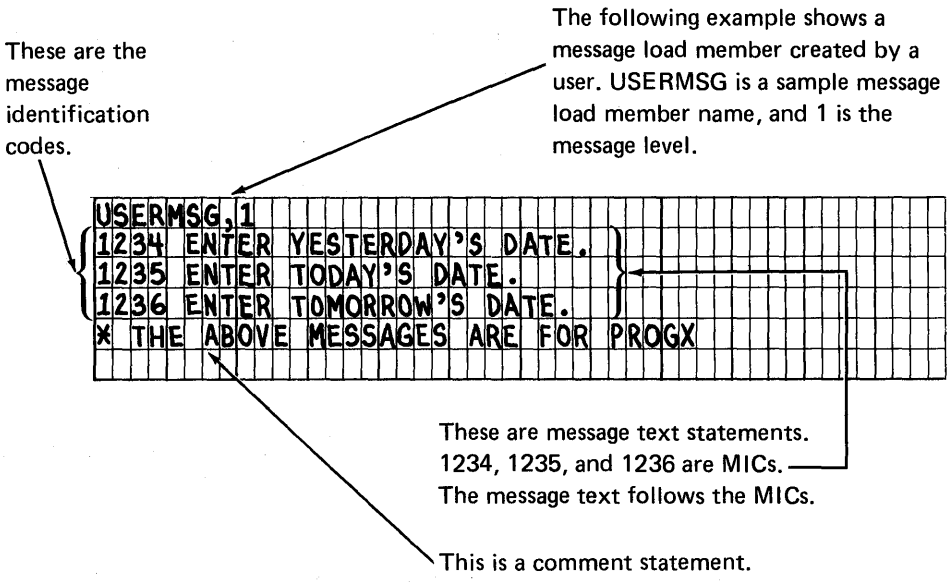
Parameters

sourcename: The name of the existing message source member that contains a control statement and message text statement(s).

REPLACE: The created message load member will replace an existing load member with the same name. The name of the load member is specified by the control statement in the source member.

library name: The name of the library that contains the source member and into which the load member is placed. If a library name is not specified, the system library (#LIBRARY) is assumed.

Example Assume a message source member contains the following statements:



If the above source member is named USERMI, the CREATE command statement appears as follows:

```
CREATE USERMI
```

The CREATE procedure causes the MICs and their associated text to be formatted into a message load member named USERMSG. The comment statement (* THE ABOVE MESSAGES ARE FOR PROGX) does not become part of the message load member (USERMSG).

CRESTART Procedure

Function

The CRESTART procedure is an optional procedure that restarts a program after a system failure if the program was checkpointed before the failure occurred. If you do not want to restart the program, the CRESTART procedure removes the checkpoint record file and checkpoint active status from the system.

The CRESTART procedure cannot be run from within another procedure.

If resource security is active, the operator who issues the CRESTART procedure must be on the list of users for the user files and libraries for the job step being restarted, as well as all job steps that follow the checkpointed job step.

The checkpoint and restart facilities can be used only for assembler programs, assembler subroutines, and COBOL programs. For a description of the checkpoint and restart facilities on System/34, see the *Concepts and Design Guide*.

Command Statement Format

CRESTART label , [DELETE / RESUME]

Parameters

label: The label of the file that contains the checkpoint records. This label is included as part of the informational message that is issued to the operator when a checkpoint is requested. The file is also displayed as a checkpoint record file when the disk VTOC is displayed by the CATALOG procedure.

DELETE: Do not restart the checkpointed job, but remove the checkpoint active status from the system. This action frees the checkpoint active files so they may be allocated, frees the libraries so they may be condensed, and removes the checkpoint record file from disk.

Note: If DELETE is selected and the checkpointed job has written data to a spool file, that spool file is not removed. The console operator should cancel the spool file if it is not needed. If the spool file is not canceled, only the records up to the last checkpoint are printed.

RESUME: Return a checkpoint job to the status saved in the checkpoint record and resume execution of the task. This parameter is the default.

Example

Restart a checkpointed job step contained in the checkpoint file CHKFIL.

DATE Procedure

Function

If you enter the DATE procedure and it is not between a LOAD statement and a RUN statement, the session date is changed. (A session begins when the display station operator signs on and, normally, ends when the display station operator enters the OFF control command.) If the DATE procedure is not used to establish a session date, the system date specified during IPL is used as the session date.

If you enter the DATE procedure between a LOAD statement and a RUN statement, it specifies the program (job step) date. When the program ends, the program date is used as the session date. If the DATE procedure is not run between a LOAD statement and a RUN statement, the session date is used as the program date. If you run two or more DATE procedures between a LOAD statement and a RUN statement, the last DATE procedure is used.

Notes:

1. The program date is used to determine the file retention period for diskette files used by the program and is printed on the printed output. The program date is also used as the creation date of any disk or diskette files created by the program.
2. If a job is placed on the input job queue, the program date when the job was placed on the queue is assigned to the job.
3. If 2400 hours (midnight) occurs, the system date is automatically updated, but the session date and the program date are not.

Command Statement Format

DATE {mmddyy / ddmmyy / yymmdd}

Parameters

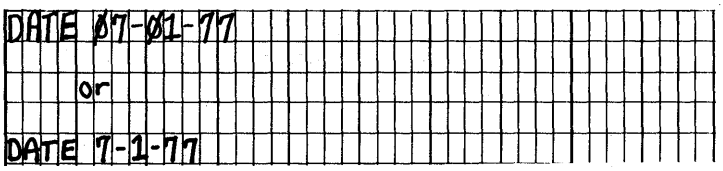
The date specified on the DATE command statement must be in the current session date format. The session date can be in any of three formats: month-day-year (mmddyy), day-month-year (ddmmyy), or year-month-day (yymmdd). The STATUS control command can be used to determine the current session date format, and the SET procedure can be used to change the current session date format. mm, dd, and yy must each be a 2-digit number, but leading zeros in month and day can be omitted when punctuation is used. A date of 000000 (all zeros) is invalid. The date can be entered with or without punctuation. For example, July 24, 1977 can be specified in any of the following ways:

7-24-77	mm-dd-yy
24-7-77	dd-mm-yy
77-7-24	yy-mm-dd
072477	mmddyy
240777	ddmmyy
770724	yymmdd

In the punctuated form, any characters except commas (,), quotes ('), numbers, and blanks can be used as punctuation. However, if the DATE procedure command is within another procedure, question marks (?), slashes (/), and hyphens (-) cannot be used as punctuation.

Example

Specify a DATE procedure command for July 1, 1977:



DEFINEID Procedure

Function

The DEFINEID procedure is used to specify a list of the remote IDs for a switched communication line that is using the SSP-ICF BSCEL subsystem. When a linkup is made on a switched line with a remote system, this list is scanned for a comparison with the ID received from the remote system. If the ID is found, line initialization completes successfully. If it is not found, the link is disconnected.

The DEFINEID procedure runs the $IDSET utility program. The $IDSET utility can also be loaded by the CNFIGICF and ENABLE procedures.

The DEFINEID procedure allocates a system file named #IBSRID, which is four sectors long and can contain up to 55 remote IDs.

The DEFINEID procedure displays two screen types. The initial screen is the command screen, which defines the operating instructions for conducting a successful data entry session. The second screen permits the entry of the initial ID list or modification of the existing IDs, including addition, deletion, deactivation, and reactivation. At end of job, all deleted IDs are removed and the file is condensed.

The following editing functions are provided to simplify data entry:

- Page function: The screen can be paged forward and backward.
- Command screen display: At any time during data entry, the command screen can be redisplayed.

The editing functions are specified by the following keys:

Key	Function
Enter/Rec Adv	Page forward
Cmd key 1	Redisplay the command screen
Cmd key 4	Page backward
Cmd key 9	End job
Cmd key 19	End job without update

The editing functions are valid in either update or display modes.

The DEFINEID procedure must be run from the system console and will reference only the system file #IBSRID. If the BSCEL subsystem has already been enabled, and you want to modify the list of remote IDs, perform the following:

1. Disable the BSCEL subsystem via the DISABLE procedure.

2. Modify the file #IBSRID via the DEFINEID procedure.

3. Reenable the subsystem via the ENABLE procedure.

It is recommended that you run the DEFINEID procedure before the BSCEL subsystem requiring remote IDs is enabled.

For the BSCEL subsystem to use the remote IDs defined by the DEFINEID procedure, the multiple remote IDs option msut be selected when you configure the BSCEL subsystem via the CNFIGICF or ENABLE procedures. For further information, see the *Interactive Communications Feature Reference Manual.*

Command Statement Format

The format of the DEFINEID procedure is:

```
DEFINEID [UPDATE ]
         [DISPLAY]
         [DELETE ]
```

Parameters

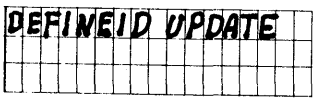
UPDATE: The DEFINEID procedure will permit the user to initially specify a list of remote IDs, modify existing IDs, add new IDs, remove old IDs, deactive IDs without deletion from the file, or reactivate previously deactivated IDs.

DISPLAY: The DEFINEID procedure displays the existing remote ID list. No modification is permitted. The display option is invalid if no IDs are defined.

DELETE: This option indicates that the system file #IBSRID is to be deleted. The delete option is invalid if no IDs are defined.

Example

Update #IBSRID:



DEFINEPN Procedure

Function

The DEFINEPN procedure provides a way to create, update, or delete one or more phone number lists, each containing up to 120 phone numbers. A phone list contains the phone number or numbers you want System/34 to call automatically. Each list is stored as a load member in either #LIBRARY or a user library, and as many lists can be defined as there is space available. Refer to the *Interactive Communications Feature Reference Manual* and to the *Data Communications Reference Manual* for additional information on how the phone list can be used.

The DEFINEPN procedure runs the $PNLM utility program. A source member containing the phone list is required for the DEFINEPN procedure. If directed to do so, DEFINEPN first calls SEU so that you can create the source member. The following SEU screen format (DEFPN) can be used to create or change the phone list source member:

```
16      084   DEFPN          120                1      ENTER     PHONTEMP



              PHONE NUMBER. . . . . . . . . . . .
              ERROR RETRY COUNT . . . . . . .1-255
              WAIT TIME TO CONNECT. . . . . .3-126




0001.00    -ENTER/UPDATE STATEMENT NUMBER
```

The DEFPN screen can be selected by pressing Cmd key 3 when in SEU, and then selecting option 40. If you do not use SEU to create this source member, a source member containing the phone list must be given to the DEFINEPN procedure.

The phone list source member contains the following information for each phone number:

- The phone number, including separator (SEP) and end of number (EON) characters. The phone number can be 1 through 22 characters, including any SEP or EON characters. The phone number must be placed in columns 1 through 22 and must be left-justified.

  An apostrophe (X'7D') represents an SEP character. The SEP character, if included, is used with some autocall units to indicate that the hardware waits for the next dial tone before continuing to dial the rest of the phone number. For example, to call a number outside of your private exchange, you might have to dial 9 and then wait for a dial tone before dialing the rest of the phone number. In this case, you would put a 9 in column 1, an apostrophe (the SEP character) in column 2, and the rest of the number in columns 3 through 22.

  An asterisk (X'5C') represents an EON character. The EON character, if included, must be the last character of the phone number. The EON character tells the autocall unit that the preceding characters make up the entire number as it is to be dialed.

  Note: Use of the separator (SEP) and end of number (EON) characters may be restricted by the autocall unit you are using. Either or both of these characters may not be allowed. Some countries may require that the SEP or EON characters appear in a certain order within a phone list. The user should be aware of such restrictions when creating a phone list. Check with your autocall supplier to find out which characters are allowed and what special requirements your autocall unit may have.

- A retry count specifying the total number of times an attempt to call this number should be made. Allowable values are 1 through 255; the default value is 1. The retry count must be placed in columns 23 through 25, right-justified, and zerofilled.

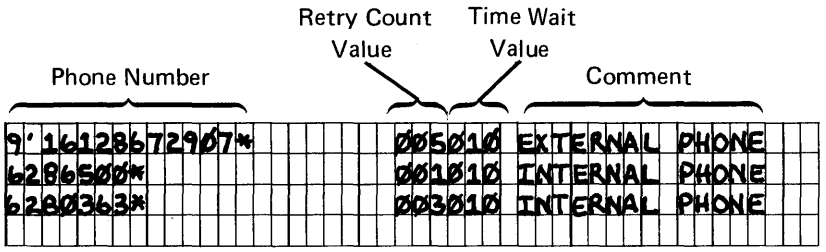
  Note: Many countries limit the number of attempts to the same telephone number in the case of unsuccessful calls. Where this maximum limitation exists, it is typically in the range of 2 through 12 attempts. You should contact the PTT (telephone company) for the maximum value that applies.

- A timer (or connection) wait value indicating the time to wait for the distant station to be connected after the last digit has been dialed. Allowable values are 3 through 126 seconds; the default value is 20 seconds. The timer wait value must be placed in columns 26 through 29, right-justified, and zerofilled.

  Note: You should be aware of any PTT (telephone company) or autocall unit restrictions on the timer wait value. For example, the timer wait value for autocall units conforming to the CCITT standard V.25 is typically restricted to the range of 10 through 40 seconds.

Using the source member, the DEFINEPN procedure prints each entry in the phone list, flagging any statements in error. A load member is built in the user-specified library if no errors occur.

System/34 calls the numbers in the order listed. When a number is called, the call may be unsuccessful. The retry count specified during the DEFINEPN procedure is associated with each number in the list. For example, suppose the following phone list is created:



When phone list PHONEL1 is specified, by either the COMM statement or the SESSION statement, or during the CNFIGICF or ENABLE procedures, System/34 calls the first number in the phone list. When a call is unsuccessful, the retry count is decremented by 1, and the next number in the list is called:

Phone number	**Retry Count**
916128672907	5 - 1 = 4 (The retry count is decremented
6286500	1 and the next number is called.)
6280363	3

The first number in the list will not be called again until the other numbers in the list have been called, or until the phone list is reinitialized using the RESTORE parameter of the COMM OCL statement, the REFRESH or RESTORE parameters of the SESSION OCL statement, the REFRESH parameter of the CNFIGICF procedure, or the SHOW parameter of ENABLE procedure.

If the retry count for a number reaches 0, a message is displayed on the system console indicating that the call was unsuccessful, and the number is marked as unsuccessfully called:

Phone number	**Retry Count**
916128672907	5 - 1 = 4
6286500	1 - 1 = 0 (Unsuccessfully called)
6280363	3

When a number is marked as unsuccessfully called, it is not called again until the phone list is reinitialized.

If a number is successfully called, an operator message is displayed on the system console, the line connection is established, and the number is marked as successfully called:

Phone number	Retry Count
916128672907	5 - 1 = 4
6286500	1 - 1 = 0 (Unsuccessfully called)
6280363	3 - 0 = 3 (Successfully called)

The number is not called again until the list is reinitialized.

In this example, the first number in the list has not been marked as successfully or unsuccessfully called. Therefore, if an application program uses the phone list without reinitializing it, the first number is the only number that can be called.

- If the call is successful, the number is marked as successfully called:

Phone number	Retry Count
916128672907	4 (Successfully called)
6286500	0 (Unsuccessfully called)
6280363	3 (Successfully called)

  The phone list must now be reinitialized to be used again. If the list is not reinitialized and an application program attempts to use the list, a message PHONE LIST EXHAUSTED is displayed on the system console for some ICF subsystems, and a minor return code of 86 (for ICF subsystems) is returned to the application program.

- If the call to the first number in the list is unsuccessful and the retry count reaches 0, the number is marked as unsuccessfully called, a message NO NUMBERS REACHED is displayed on the system console for some ICF subsystems, and a minor return code of 85 (for ICF subsystems) is returned to the application program:

Phone number	Retry Count
916128672907	0 (Unsuccessfully called)
6286500	0 (Unsuccessfully called)
6280363	3 (Successfully called)

  The phone list must be reinitialized to be used again. When the list is reinitialized, all retry counts are set to the counts specified during the DEFINEPN procedure and calling begins with the first number in the list when the list is used again.

For more information about the reinitializing phone lists, see the *ICF Reference Manual* and the *Data Communications Reference Manual.*

It might be desirable for security purposes to restrict the use of the DEFINEPN procedure to one person (for example, the system operator or some other key operator). The creation and updating of phone lists can be restricted using the following steps:

1. Define resource security for a user library and restrict owner access to that library to a single user. Information about defining resource security is in the *Installation and Modification Reference Manual.*

2. Copy the DEFINEPN procedure and the $PNLM utility program from #LIBRARY to the restricted library, using the $MAINT utility program. Use COPY utility control statements, one each for DEFINEPN and $PNLM.

3. Delete the DEFINEPN procedure and the $PNLM utility program from #LIBRARY using the $MAINT utility. Use DELETE utility control statements with RETAIN-S specified, one each for DEFINEPN and $PNLM.

4. Use SEU or the $MAINT utility to change the first statement in the new copy of the DEFINEPN procedure. Change the statement // LIBRARY NAME-0 to // LIBRARY NAME-'library name', where the library name is the name of the restricted library.

The owner of the restricted library is now the only one allowed to create or modify phone lists using the DEFINEPN procedure. Output from the DEFINEPN procedure can be directed to any library, whether protected or not, at the option of the owner of the protected library.

Command Statement Format

DEFINEPN source member name,[source library name / #LIBRARY],

[load member name / source member name],[load library name / #LIBRARY],[NODELAY / DELAY]

Parameters

source member name: The name of the library source member that contains the phone numbers.

source library name: The name of the library that contains the source member. If source library name is not specified, the system library (#LIBRARY) is assumed.

load member name: The name of the phone list to be built. If load member name is not specified, the source member name is used.

load library name: The name of the library that contains or will contain the phone list. If the load library name is not specified, the system library (#LIBRARY) is assumed.

DELAY: DELAY must be specified for installations in all countries except the United States and Canada. When the phone list is specified, autocall will process the phone list in the following manner:

- Autocall initially waits 60 seconds before placing a call to the first number in the list.
- On each *retry* to a number previously unsuccessfully called, autocall waits 60 seconds before placing the call.
- On each attempt to a number *not* previously called, autocall waits 3 seconds before placing the call.

NODELAY: When the phone list is specified, autocall will process the phone list without intercall delays. NODELAY is the default.

Example

Create a phone list, using the $MAINT utility, in a library source member named PHONEL2 to be stored in a library named COMMLIBR. Phone number 1-612-555-1111 has a retry count of 3, with a timer wait value of 10 seconds. Phone number 123-4567 has a retry count of 5, with a timer wait value of 10 seconds. Phone number 1-123-7645 has a retry count of 1 (the default), with a timer wait value of 20 seconds (the default):

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,TO-COMMLIBR,LIBRARY-S,NAME-PHONEL2
16125551111*                    003010 PHONE NUMBER 1
1234567*                        005010 PHONE NUMBER 2
11237645*                              PHONE NUMBER 3
// CEND
// END
```

Compile the library source member named PHONEL2 to be stored in a library called COMMLIBR:

```
DEFINEPN PHONEL2,COMMLIBR
```

DEFINX21 Procedure

Function

The DEFINX21 procedure provides a way to interactively build or change a list of phone numbers for an X.21 public data network. Each list can contain up to 84 numbers for the public data network. Each list is stored as a library load member. For information about how System/34 uses a list of phone numbers for an X.21 public data network, see the *Interactive Communications Feature Reference Manual* and the *Data Communications Reference Manual.*

The DEFINX21 procedure runs the $XNLM utility program.

When the DEFINX21 procedure is run, you can choose either to build a new list of phone numbers or to change an existing list. You are then prompted for the name of the list and name of library in which the list is to be stored. The default library is #LIBRARY. After the list name and the library name are entered, a screen is displayed allowing you to build a list or to change numbers within an existing list:

```
                    3.0 DEFINX21 PHONE LIST

                         Build a phone list

  Connection number      Retry value       Delay value
  1-18 digits,DC,I,D        1-255             0-16









Enter-Update   CMD7-EOJ   Help-More information
```

The following information can be specified for each number in the list:

- A number consisting of 1 through 18 characters. Valid characters are the numeric digits 0 through 9, slashes (/), hyphens (-), periods (.), and commas (,). The phone number must be left-justified.

  The characters "DC"in positions 1 and 2, with no characters following, indicate a direct call. The direct call capability is offered by public data networks to simplify the call procedure for a subscriber who is always connected to the same remote system. On calling, System/34 presents to the network the direct call sequence instead of the remote system's number. The network connects the calling System/34 to the remote system.

- A retry value specifying the total number of times an attempt to call the number should be made. Allowable values are 001 through 255; the default value is 001. The retry value must be right-justified and zerofilled.

- A delay value in seconds indicating the time to wait before attempting a retry to a number that could not be reached. Allowable values are 00 through 16; the default value is 00 seconds. The delay value must be right-justified and zerofilled.

  Note: The delay function is invoked only if the call failed to connect on the last attempt because of a CCITT Group Code 2 (no connection, number busy) error or a CCITT Group Code 6 (network congestion) error.

The DEFINX21 procedure verifies that allowable values have been entered into the above fields. The load member is built or updated using the specified list of numbers and is saved in the specified library.

When building or changing a list of phone numbers for an X.21 public data network, you can delete or insert numbers. To delete a number within the list, move the cursor to the first position of the number to be deleted. Place a D in the first position of the number, and press the Enter/Rec Adv key. The number is deleted.

To insert a number within the list, move the cursor to the first position of the first number that precedes the number to be inserted, and press the Enter/Rec Adv key. A number with the characters DC, a retry value of 001, and a delay value of 00 is inserted. Replace the direct call characters with the proper number and the retry and delay values with the proper values.

Note: The DEFINX21 procedure will not accept an attempt to insert a number after the last number defined in the list.

The following command keys can be used during the DEFINX21 procedure:

Key	Function
Cmd key 3	Returns the screen to the previous display. If a build or change operation was being performed, and Cmd key 3 is used, the newly specified numbers are lost, or the list is not updated. This command key cannot be used on the operation selection screen.
Cmd key 7	Ends the DEFINX21 procedure. If used during a build operation, the list of phone numbers is created. If used during a change operation, the list is updated.
Cmd key 19	This command key can be used during a build or change operation. If this command key is used, the DEFINX21 procedure is ended, and any changes made to the list are ignored. The list is not created or updated.
Roll Up	Displays the next number, if one exists, in the list of phone numbers.
Roll Down	Displays the previous number, if one exists, in the list of phone numbers.

Command Statement Format

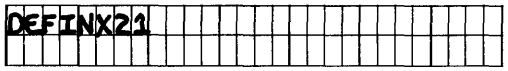
DEFINX21

Parameters

None

Example

To build a list of phone numbers, named PHONEX21 and stored in library COMMLIB, for an X.21 public data network, enter:



Then do the following:

- Select option 1 to build the list.
- Enter the name of the list, PHONEX21, and the name of the library in which the list is to be stored, COMMLIB.
- Enter the numbers, retry values, and delay values for the list as shown in the following screen:

```
                    3.0 DEFINX21 PHONE LIST

                         Build a phone list

 Connection number        Retry value       Delay value
 1-18 digits,DC,I,D          1-255             0-16

 15075553131                 001               00

 16125554745                 003               05

 DC                          001               00



Enter-Update   CMD7-EOJ   Help-More information
```

- Press Cmd key 7 to end the build operation and the DEFINX21 procedure.

DELETE Procedure

Function

The DELETE procedure causes the space occupied by the named user library, diskette, or disk file(s) to be made available and, optionally, erases the contents of the deleted file. The DELETE procedure cannot be used to delete a file while the file is being used by another job or while the file is being used by this job as a job (RETAIN-J) file. The system library (#LIBRARY), checkpoint active files, or checkpoint active libraries cannot be deleted with this procedure.

The DELETE procedure runs the $DELET utility program.

Command Statement Format

```
DELETE label, [F1|I1] , [SCRATCH|REMOVE|ERASE] , [mmddyy|ddmmyy|yymmdd] , [LIBR] , [S1|S2|S3|M1.nn|M2.nn]
```

Parameters

label: The label of the user library or file to be deleted from the disk, or the label of the file to be deleted from the diskette(s). ALL cannot be used as a label.

F1: The file to be deleted is a disk file.

I1: The file to be deleted is on one or more diskettes. If the file is a multivolume file, the operator is prompted to insert each required diskette. If a value is not specified in the second parameter position, I1 is assumed.

SCRATCH: If the file or the library is on a diskette, the expiration date is set to the current program date. If the file or library is on the disk, the VTOC entry for the file is removed. If the third parameter is not specified, SCRATCH is assumed.

REMOVE: The VTOC entry for the file or library is removed.

ERASE: The VTOC entry for the file or library is removed. Data that was in the deleted file or library also is removed; that is, all bytes within the physical extent of the file or the library are replaced with binary zeros. A system warning message is displayed if ERASE is specified.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file to be deleted. For a disk file, the specified date must be in the same format as the session date. For a diskette file, the specified date must be in the same format as the creation date of the diskette file.

Notes:

1. If no date is specified and more than one file with the specified label exists on the disk, the operator can delete all of the files with that label or cancel the job.
2. If no date is specified and more than one file with the specified label exists on a diskette, only the first file with the specified label is deleted.
3. If LIBR is specified, the date parameter cannot be specified.

LIBR: If LIBR is specified, a user library is to be deleted. If LIBR is not specified, a file is to be deleted.

Notes:
1. LIBR and date cannot both be specified.
2. LIBR and I1 cannot both be specified.
3. Whenever a display station or a job is using any members from a library, you are restricted from deleting that library until the display station has signed off or the job has gone through termination.

S1, S2, or S3: Identifies the individual diskette slot that contains the diskette from which the file or the library is to be deleted.

Note: If the file is a multivolume file, the DELETE procedure can use all three individual slots. The volumes to be deleted should be placed in order in the slots. The first volume to be deleted must be in the slot specified by this parameter. Each time a volume in slot S3 is deleted, a message is displayed if more volumes remain to be deleted. The system operator can then:

- Insert the next volumes beginning in slot S1 and select the option that causes the DELETE procedure to resume deleting at slot S1
- Insert the next diskette in slot S3 and select the option that causes the DELETE procedure to resume deleting at slot S3

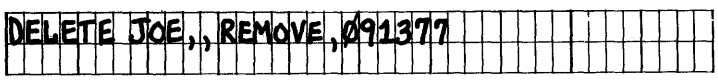
M1.nn or M2.nn: Identifies the magazine location that contains the diskette from which the file or the library is to be deleted.

Note: If the file is a multivolume file, the volumes to be deleted must be placed in order in the magazine. The first volume to be deleted must be in the location specified by this parameter. When the last volume in a magazine is deleted, a message is displayed if more volumes remain. The system operator can then:

- Insert the next magazine into the current magazine slot and select the option that causes the DELETE procedure to resume deleting at the first location in the current magazine slot.
- Ensure that the next magazine is in the other magazine slot and select the option that causes the DELETE procedure to resume deleting at the first location in the other magazine slot.

Example

Delete the diskette file labeled JOE (dated September 13, 1977):



DISABLE Procedure

Function

The DISABLE procedure terminates an enabled subsystem configuration and can be run only from the system console. If there are active sessions, DISABLE issues a halt with four options. The options and their functions are:

0 Hold the DISABLE procedure; no new sessions are to start. The subsystems will continue the DISABLE procedure when the current sessions terminate.

1 Retry the DISABLE procedure and check again for active sessions.

2 Immediately terminate the active sessions and continue the DISABLE procedure.

3 Terminate the DISABLE procedure and ignore the DISABLE request. The DISABLE procedure goes to end of job; however, the subsystem remains enabled.

DISABLE frees the subsystem queue space in main storage (if any) associated with the subsystem provided the same subsystem is not using it for another configuration. If there are no other interactive communications feature subsystems running, DISABLE also frees the main storage space required for SSP-ICF common queue space and SSP-ICF control. If there are no other enabled configurations of this type, the subsystem task is terminated.

The DISABLE procedure runs the $IEDS utility program.

Command Statement Format

DISABLE subsystem name, [location name]

Parameters

subsystem name: The member name specified during configuration of the subsystem to be disabled.

location name: The location specified is terminated. The subsystem stays up as long as there are active locations.

Example

Disable the subsystem configuration SUB1:

DISPLAY Procedure

Function

The DISPLAY procedure lists, on the system list device assigned to the requesting display station, all or part of a disk file. The DISPLAY procedure cannot be used to display the contents of a file that is being used by another job.

For information about assigning the system list device, see the description of the SYSLIST procedure later in this chapter.

Note: When the system list device is the display screen, you control the record display in the following manner:

- To display the next record, press Cmd key 1.
- To display the previous record, press Cmd key 2. (Cmd key 2 is ignored if the displayed record is the first record in the file.)
- To terminate the display, press Cmd key 3.
- To display the next bytes of the displayed record, press the Shift and Roll Up keys.
- To display the previous bytes of the displayed record, press the Shift and Roll Down keys.

  Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

The DISPLAY procedure runs the $COPY utility program.

Command Statement Formats

For displaying an entire file:

```
DISPLAY label,[mmddyy]
              [ddmmyy],,,,[IGC]
              [yymmdd]
```

For displaying records by relative record numbers:

```
DISPLAY label,[mmddyy]
              [ddmmyy],RECORD,value1,[value2],[IGC]
              [yymmdd]
```

Parameters

label: The label of the file to be displayed or printed.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file to be displayed or printed. If a date is not specified, and if more than one file with the same specified label exists, the file with the most recent creation date is displayed or printed.

RECORD: The records from the file are to be displayed or printed based on their relative record number.

value1: The number of the first record to be displayed or printed. value1 is valid for sequential, indexed, and direct files.

value2: The number of the last record to be displayed or printed. value2 is valid for sequential, indexed, and direct files. If value2 is not specified, records are listed until the end of the file is reached. If value2 is the same as value1, one record is displayed or printed.

IGC: The IGC parameter is used for the ideographic version of the SSP and is ignored for nonideographic systems. IGC specifies that the file might contain ideographic characters and, if possible, DISPLAY should display those characters.

Note: When nonideographic characters are printed, 100 characters are printed on each line. The characters are printed in print positions 1 through 100. When a mixture of ideographic and nonideographic characters are printed, the number of characters printed on a line varies. Each ideographic character represents 2 bytes of information and requires two print positions on the line. If an ideographic character begins in print position 100, position 100 is left blank and the character is printed in positions 0 and 1 on the next line. (Position 0 is staggered one position to the left of the normal starting position, which is position 1.) To determine the position of a character within the record, you can use the following algorithm:

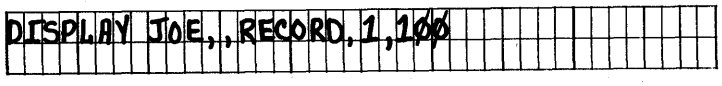
$$\begin{pmatrix}\text{position in}\\ \text{record}\end{pmatrix} = 100 \times \begin{pmatrix}\text{number of}\\ \text{previous lines}\end{pmatrix} + \begin{pmatrix}\text{print position of}\\ \text{the character}\end{pmatrix}$$

For example, if an ideographic character begins in position 0 of the third line of printout, the character begins in byte 200 of the record (100 x 2 + 0 = 200).

If IGC is not specified, the file contents are displayed just as they would be for a nonideographic system. Any ideographic data in the file is treated as one-byte alphameric characters.

Example

Display or print the first 100 records of the most recent file created with the label JOE:



ENABLE Procedure

Function

The ENABLE procedure initializes an interactive communications feature (SSP-ICF) subsystem and it must be run before an SSP-ICF subsystem can be used. This procedure can be run only from the system console. When an enable occurs with no SSP-ICF tasks currently running, the ENABLE procedure loads the SSP-ICF control routines and allocates the SSP-ICF common queue space. ENABLE also defines the subsystem environment, loads the requested subsystem and interrupt handlers (if they are not already loaded), and gives the subsystem control. When the subsystem is loaded, additional subsystem queue space is also allocated if required.

The enable function may allocate nonswappable main storage for SSP-ICF control routines, SSP-ICF common queue space, SSP-ICF subsystem queue space, and for the subsystem code. Allocating nonswappable storage decreases the user-available swappable storage. Enable will not allow the available swappable user storage to become less than the current maximum region size (including the region size for the enable function).

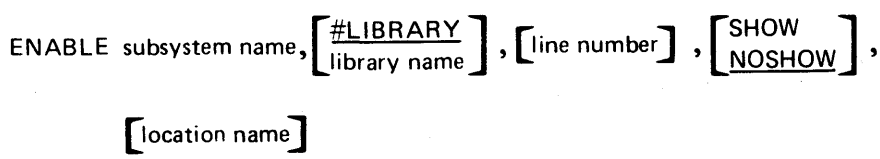
The subsystem environment is defined by a composite of the following:

- ENABLE parameters
- Subsystem configuration variables
- System console display station communications variables (that is, $SETCF parameters)

After you issue the ENABLE command, communications via the subsystem may take place.

The ENABLE procedure runs the $IENBL utility program.

Command Statement Format



Parameters

subsystem name: The member name given to the subsystem configuration when it was configured.

library name: The name of the library containing the subsystem configuration. The default is #LIBRARY.

line number: The number of the communications line to use (1, 2, 3, or 4).

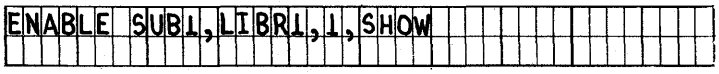
SHOW: The SHOW option displays some of the parameters configured for a particular subsystem. The operator has the option of modifying some of these parameters. For a description of the parameters displayed, refer to the *Interactive Communications Feature Reference Manual.*

NOSHOW: The parameters configured for a particular subsystem are not displayed. This option is the default.

location name: Specifies the location to be activated. The SSP-ICF subsystem is activated if it is not already enabled.

Example

In the following example, the ENABLE procedure activates the subsystem specified by the configuration member SUB1 found in the user library LIBR1. SUB1 uses communications line 1, and the parameters for this configuration are to be displayed.



EXTRACT Procedure

The EXTRACT procedure is not documented in this manual. For information about this procedure, see the *WSU Reference Manual*.

FORMAT Procedure

Function

The FORMAT procedure processes user-generated source information called display screen format specifications to perform one the following functions:

- Create a new display screen format load member containing the formats defined by the source specifications. All formats used by an application program can be placed in one load member (up to a maximum of 32 formats in one member) or they can be placed in more than one member. The application program must open each load member containing formats used by the program.
- Add one or more formats to an existing display screen format load member.
- Replace one of the formats in an existing display screen format load member.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

For information about display screen format specifications, see the description of the $SFGR utility program in Chapter 4.

In addition, the FORMAT procedure can be used to delete a format from an existing display screen format load member. No source input is required when deleting a format.

The FORMAT procedure runs the $SFGR utility program.

Command Statement Formats

For creating, adding to, or updating a screen format load member:

```
FORMAT [CREATE]
       [ADD   ] ,load member name, [output library name] ,
       [UPDATE]                    [#LIBRARY           ]

        source member name, [input library name] , [number of formats] ,
                            [#LIBRARY           ]   [1              ]

        [REPLACE] , [HALT  ] , [PRINT  ]
                    [NOHALT]   [NOPRINT]
                               [PARTIAL]
```

For deleting a format from a screen format load member:

```
FORMAT DELETE,load member name, [user library name] ,format name
                                [#LIBRARY        ]
```

Parameters

CREATE: Create a new display screen format load member. If a parameter is not entered in the first parameter position, CREATE is assumed.

ADD: Add one or more formats to an existing display screen format load member.

UPDATE: Replace a format in an existing display screen format load member.

DELETE: Remove a format from an existing display screen format load member. If the removed format was the only one in the member, the display screen format load member is removed from the library.

load member name: The name of the display screen format load member.

output library name: The name of the library that contains or will contain the display screen format load member. If output library name is not specified, the system library (#LIBRARY) is assumed.

source member name: The name of the library source member that contains the display screen format specifications.

input library name: The name of the library that contains the source member. If input library name is not specified, the system library (#LIBRARY) is assumed.

number of formats: The number of formats defined in the display screen format specifications source member. This number determines the amount of temporary workfile space required by the $SFGR utility program. The larger the number, the more temporary disk file space will be required. If number of formats is not specified, 1 is assumed. The number of formats should not be specified if UPDATE is specified.

REPLACE: If a load member already exists with the same name as the display screen format load member being created, the existing member is deleted by FORMAT. REPLACE has meaning only if CREATE is specified (or the default is taken) for the first parameter on the FORMAT statement. If REPLACE is not specified and a duplicate member exists, a message is displayed. The operator must then decide whether to delete the existing member or to end the job.

HALT: SFGR issues a halt message indicating that warning and/or terminal errors were encountered during the processing of a screen source member. If a warning error is issued, the operator is given the option of either canceling or continuing the processing. If a terminal error is issued, the operator is given the option of canceling the processing. If a parameter is not entered in the eighth parameter position, HALT is assumed.

NOHALT: The $SFGR utility program does not issue a halt message indicating that warning and/or terminal errors were encountered during the processing of a screen source member. For warning errors, the job step is completed and the screen format generation is performed. For terminal errors, the job step is ended and no screen format generation is performed.

Note: For terminal errors, the ?CD? OCL substitution expression is set to 1008; refer to *Substitution Expressions* in Chapter 5 for more information.

PRINT: Causes the $SFGR utility program to print the following:

- The screen source member name
- The screen format S- and D-specifications
- Any informational, warning, or terminal errors
- The input and output buffer descriptions
- A list of the screen format indicators used
- The input and output library names
- The screen format load member name
- The storage requirements for each format
- The data stream length for each format

If a parameter is not entered in the ninth parameter position, PRINT is assumed.

NOPRINT: If a terminal error is encountered in the source specifications, only the statement in error and the error message are printed.

PARTIAL: Causes the $SFGR utility program to print the following:

- The screen source member name
- Any warning or terminal messages together with the statement causing the message, or any informational messages
- The input and output library names
- The screen format load member name

user library name: The name of the library that contains the display screen format load member containing the format being deleted. If user library name is not specified, the system library (#LIBRARY) is assumed.

format name: The name of the format being deleted.

Example

Create a new display screen format load member called FORMAT2 in the system library. The display screen format specifications define three display screen formats and are in a source member called INPUT. The source member is in a user library called ULIB1:

```
FORMAT CREATE,FORMAT2,,INPUT,ULIB1,3
```

FROMLIBR Procedure

Function

The FROMLIBR procedure copies library members into a new disk or diskette file or adds library members to an existing disk or diskette file containing library members. Members in files created by the FROMLIBR procedure can be placed back in the library by the TOLIBR procedure.

The FROMLIBR procedure runs the $MAINT utility program.

Note: If you use the FROMLIBR procedure to copy library members from a library to a file, you can copy the members from the file back to the library only by using the TOLIBR procedure or the $MAINT utility.

Command Statement Formats

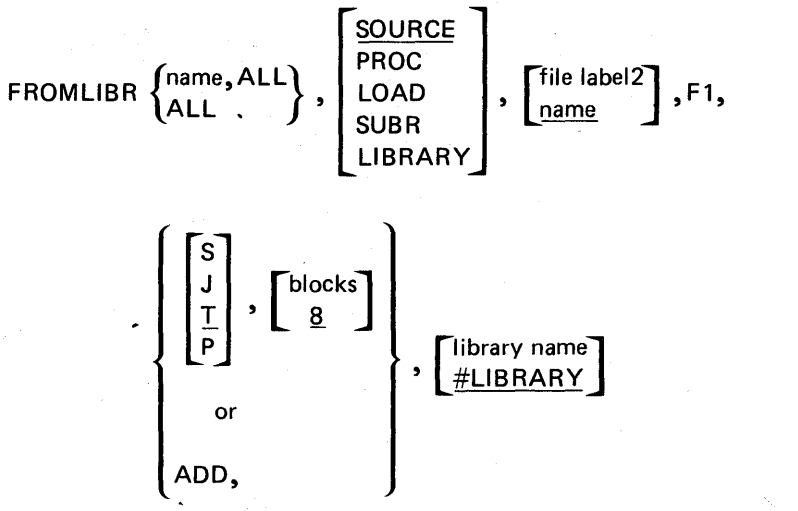
For copying a non-SSP library member or adding a non-SSP member to a sequential disk file:

```
                                  [SOURCE]
                                  [PROC  ]
FROMLIBR member name,             [LOAD  ]  , [file label1   ],
                                  [SUBR  ]    [member name  ],
                                  [LIBRARY]

                                  [S]                 ]
                                  [J]                 ]
                                  [T]  , [blocks]     ]
                                  [P]    [  8  ]      ]
                                                      ]
                                   or                 ]
                                                      ]
                                  ADD,                ]
```

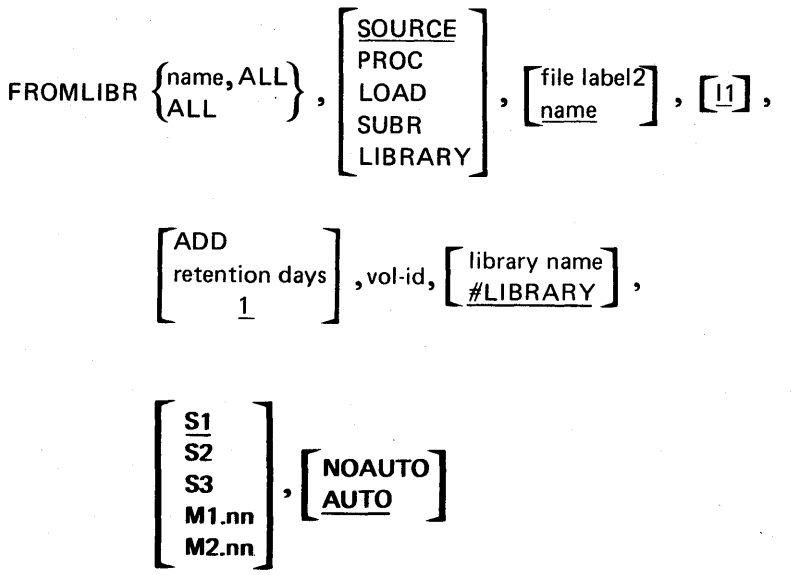
For copying a non-SSP library member or adding a non-SSP member to a sequential diskette file:

```
                                  [SOURCE]
                                  [PROC  ]
FROMLIBR member name,             [LOAD  ]  , [file label1   ] , [I1] ,
                                  [SUBR  ]    [member name  ]
                                  [LIBRARY]

                                  [ADD                ]
                                  [retention days     ] , vol-id, [library name ] ,
                                  [     1             ]           [#LIBRARY ]

                                  [S1  ]
                                  [S2  ]
                                  [S3  ]  , [NOAUTO]
                                  [M1.nn]   [AUTO  ]
                                  [M2.nn]
```

For copying or adding all non-SSP members or for copying or adding all non-SSP members beginning with specified characters to a disk file:



For copying or adding all non-SSP members or for copying or adding all non-SSP members beginning with specified characters to a diskette file:



Parameters

member name: The name of the non-SSP library member to be copied from the library.

name,ALL: All non-SSP members with names beginning with the indicated characters (name) are to be copied. A maximum of 7 characters can be used for name. For example: PAYR,ALL refers to non-SSP members having names that begin with PAYR.

ALL: All non-SSP members are to be copied from the library.

Note: All non-SSP members include IBM-supplied, non-SSP members (such as members for other program products) as well as members you have created.

SOURCE: Source members are copied. If a second parameter is not specified, SOURCE is assumed.

PROC: Procedure members are copied.

LOAD: Load members are copied.

SUBR: Subroutine members are copied.

LIBRARY: All types of members (SOURCE, PROC, LOAD, and SUBR) are copied.

file label1: The label of the file to be created. If file label1 is not specified, the name specified for member name is assumed.

file label2: The label of the file to be created. If file label2 is not specified, name is assumed. If ALL is specified (all non-SSP members are copied from the library) and file label2 is not specified, the operator is prompted for the label of the file being created.

F1: The output file is created on the disk.

I1: The output file is created on a diskette. If the fourth parameter is not specified, I1 is assumed.

ADD: Library member(s) will be added to an existing file that contains library members.

Note: When adding a member to a disk file, the file must contain enough unused space to hold the member. When adding a member to a diskette file, the file must be the last active file on the diskette. ADD cannot be specified if retention days is specified.

retention days: The length of the retention period (0 to 999 days) for the diskette file. If I1 is specified or assumed and retention days is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file. Retention days cannot be specified if ADD is specified. For more information on diskette file retention, see *File Statement (for Diskette Files)*, RETAIN parameter, in Chapter 1.

S: The disk file is a scratch file. (The file does not exist after FROMLIBR terminates).

J: The disk file is a job file.

T: The disk file is a temporary file. If F1 is specified and P, T, or S is not specified, T is assumed.

P: The disk file is a permanent file.

vol-id: The volume ID of the diskette; 1 through 6 alphameric characters. If the volume ID is not specified on the procedure command, the operator is prompted for the volume ID.

blocks: The number of blocks to allocate for the output file. blocks is ignored if ADD is specified. If blocks is not specified, 8 is assumed.

library name: The name of the library containing the member(s) to be copied. If library name is not specified, the system library (#LIBRARY) is assumed.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be used for output. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be used for output. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, FROMLIBR uses only the specified slot. When the end of a diskette is reached, the SSP displays a message; the operator must insert the next diskette in the slot being used.

- If M1.nn or M2.nn is also specified, FROMLIBR uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes are required, the SSP displays a message; the operator must insert the next magazine in the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, FROMLIBR uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes are required, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

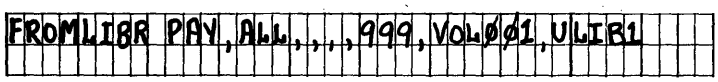
- If M1.nn or M2.nn is also specified, FROMLIBR uses both magazine slots. Processing begins with the diskette in the specified location and continues through the diskette in location M2.10. If more diskettes are required, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a parameter is not specified, AUTO is assumed.

Examples

Example A

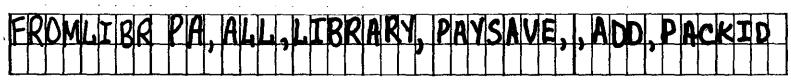
Assume that all payroll application source programs are in a library called ULIB1 and begin with the characters PAY. To copy all payroll source programs to a permanent diskette file, specify:



All payroll application programs would be placed into a sequential diskette file named PAY.

Example B

To add all system library members whose names begin with the characters PA to a diskette file named PAYSAVE, specify:



HELP Procedure

Function

The HELP procedure is an optional procedure that:

- Aids the operator in executing System/34 procedures
- Displays quick-reference information for many System/34 functions

HELP can be used to run the following procedures:

- SSP procedures
- Procedures that are part of the Utilities Program Product
- Procedures that run the program products and the COBOL PRPQ (ASM, COBOL, FORTRAN, OLINK, BASIC, and RPG)
- Data communications procedures
- Service aid procedures

To run a procedure using HELP, the operator can progress through a series of menus until he selects the name of the procedure to be run or he can enter the name of the procedure directly on the HELP command statement. The HELP procedure then prompts the operator for the parameters used by the procedure. After entering the parameters, the operator can press the Enter/Rec Adv key to run the procedure immediately or press Cmd key 4 to place the procedure on the input job queue. A message is displayed showing a maximum of 75 characters of the information entered. The entire procedure and its parameters are logged to the history file. Examples at the end of this section show how the operator can use HELP to run a procedure.

Note: The HELP procedure uses bytes 249 through 256 of the display station local data area. When the HELP procedure is run, any user data in those bytes is destroyed. For further information, see the description of the LOCAL OCL statement in Chapter 1.

The HELP procedure can be used to display quick-reference information about:

- Operator control commands
- OCL statements
- Procedure control expressions

The operator can progress through a series of menus until he selects the required function, or he can enter the name of the function directly on the HELP command statement.

Instead of using the HELP command statement, the operator can initiate the HELP procedure by pressing the Help key. The system will accept a Help key request even from a mandatory menu; however, if the operator tries to use the help function to run a job, an error message is displayed. See the *Installation and Modification Reference Manual* for information about menu security and mandatory menus.

During the execution of the HELP procedure, the following function keys and command keys can be used:

Key	**Function**
Cmd key 1	Displays a description of the operator options available
Cmd key 2	Displays a list of HELP categories
Cmd key 3	Displays the HELP list for the current category
Cmd key 4	Causes the requested procedure to be placed on the input job queue
Cmd key 7	Cancels the HELP procedure
Enter/Rec Adv	Causes the requested procedure to be executed immediately
Roll Up	Displays the next display in a series of displays
Roll Down	Displays the previous display in a series of displays

During the execution of the HELP procedure, some error messages are displayed without message numbers. The following table lists those messages, their meanings, and the action required to recover from the error:

Message	Meaning	Recovery
INVALID NAME SPECIFIED FOR HELP	A name not supported by HELP was entered.	Press the Error Reset key and enter a valid name.
REQUIRED PARAMETER MISSING–ENTER REQUIRED PARAMETER	A parameter required to execute the requested function was not entered.	Press the Error Reset key and enter the required parameter(s).
I/O ERROR–PRESS ERROR RESET TO CONTINUE	A display station input/output error occurred.	Press the Error Reset key to resume processing of the HELP procedure.
INVALID NUMBER SPECIFIED–ENTER VALID NUMBER	An invalid response number was entered.	Press the Error Reset key and enter a valid number.
CANNOT PLACE JOB ON INPUT JOBQ NOW–JOBQ FULL	The input job queue is full.	Press the Error Reset key. Use command key 4 again when you determine that space is available on the job queue.
CANNOT PLACE JOB ON INPUT JOBQ–JOBQ NOT ACTIVE	The input job queue is not active.	Press the Error Reset key to resume processing of the HELP procedure. Your request to put the job on the input job queue is ignored.
CANNOT PLACE JOB ON INPUT JOBQ NOW–JOBQ DISK ERROR	A disk error occurred while the SSP was reading from or writing to the input job queue.	Press the Error Reset key to resume processing of the HELP procedure. Use the CANCEL command to cancel the job on the input job queue. The canceled job must be resubmitted.
CANNOT RUN CMD USING HELP–MANDATORY MENU ACTIVE	You have attempted to use the HELP procedure to run a job, but a mandatory menu is active at your display station.	Press the Error Reset key to resume processing of the HELP procedure.

The HELP procedure runs a utility program called $HELP. $HELP can be run only by the HELP procedure. You cannot run $HELP by entering LOAD and RUN OCL statements from the keyboard.

Command Statement Format

HELP [procedure name | command name | OCL | PCE]

Parameters

procedure name: The name of the procedure to be run.

command name: The name of the control command for which reference information should be displayed; for example, HELP STATUS displays reference information about the STATUS control command.

OCL: Displays the list of System/34 OCL statements. From that list, the operator selects the statement for which reference information will be displayed.

PCE: Displays the list of System/34 procedure control expressions. From that list, the operator selects the expressions for which reference information will be displayed.

Examples

Example A

To run the FROMLIBR procedure, the operator enters:

HELP FROMLIBR

The HELP procedure then prompts the operator for parameters used by FROMLIBR:

```
                          FROMLIBR PROCEDURE                    OPTIONAL-(O)
      Copies or adds library members to a disk or diskette file.
Member Name, Partial Name Or All ..............................
  If Partial Name Or All ......................................
Library Type (PROC/LOAD/SUBR/SOURCE/LIBRARY) ................... SOURCE
Disk/Diskette File Label ....................................          (O)
Unit (F1/I1) ................................................... I1
```

In this case, a source member called DMSRC is to be copied to diskette. After entering DMSRC in response to the first prompt, the operator presses the Enter/Rec Adv key. The HELP procedure then prompts for the remaining parameters:

```
                          FROMLIBR PROCEDURE                    OPTIONAL-(O)
      Copies or adds library members to a disk or diskette file.
Member Name, Partial Name Or All ..............................  DMSRC
  If Partial Name - Enter ALL .................................
Library Type (PROC/LOAD/SUBR/SOURCE/LIBRARY) ..................  SOURCE
Disk/Diskette File Label ....................................          (O)
Unit (F1/I1) ................................................... I1
Retention Days Or ADD .......................................  1
Volume ID ......................................................
Library Name ................................................... #LIBRARY
Location (S1/S2/S3/M1.nn/M2.nn) ................................ S1
Automatic Advance To Next Slot/Magazine (AUTO/NOAUTO) ..........  AUTO
```

In this example, DMSRC is in #LIBRARY and is to be copied to the diskette in slot S1; therefore, the operator can use the displayed defaults, and he can press the Enter/Rec Adv key to run FROMLIBR.

Example B

The operator wants to run the BLDMENU procedure. To use the HELP procedure, the operator presses the Help key; the first HELP menu then appears:

```
                    SYSTEM/34 HELP CATEGORIES

1. SSP Procedure Commands

2. Control Commands

3. Operation Control Language - OCL

4. Procedure Control Expressions

5. Utilities - DFU,SDA,SEU,SORT,WSU

6. Languages & Compilers - ASM,BASIC,COBOL,FORTRAN,OLINK,RPG

7. Data Communications Procedures

8. Service Aid Procedures


ENTER NUMBER OF CATEGORY--->                    CMD KEY 1 - FUNCTIONS
                                                CMD KEY 7 - CANCEL
```

The operator enters 1 to display the list of SSP procedures:

```
                      SYSTEM/34 SSP PROCEDURES

ALTERBSC - Alter BSC Parameters          CONDENSE - Free Library Space
ALTERSDL - Alter SDLC Parameters         COPYII   - Copy Diskette
BACKUP   - Backup System Library         COPYPRT  - Copy Spool File Entries
BLDFILE  - Create Disk File              CREATE   - Generate Message Member
BLDLIBR  - Create User Library           CRESTART - Restart Checkpoint Task
BLDMENU  - Create Menu                   DATE     - Change Session Date
BUILD    - Correct Disk Data             DELETE   - Delete File Or Library
CATALOG  - Display VTOC                  DISABLE  - Disable Subsystem
COMPRESS - Free Disk Space               DISPLAY  - Display Data File

ENTER PROCEDURE NAME TO BE EXECUTED --->                  Roll To Continue
```

The operator then enters the name BLDMENU, and the HELP procedure prompts for the BLDMENU parameters:

```
                          BLDMENU PROCEDURE                    OPTIONAL-(O)

          Creates the library members required to display a menu.

Menu Name ..................................................

Name Of The Display Text Source Member .....................          (O)

Library That Contains The Source Message Member ..... #LIBRARY

Library To Contain The Format Load Member ..... #LIBRARY

If The New Member Will Replace An Existing Load Member,
     Enter REPLACE .........................................          (O)

If The Load Member Is To Remain In The Output Library,
     Enter KEEP ............................................          (O)

If Creating A Free-Format Menu, Enter FREEFORM .............          (O)
```

In this example, a menu called MENU01 is to be built. Display text source member MENU01DT is in a library called ULIB1.The new menu is to be placed in a library called ULIB2. The operator enters the parameters as follows and presses the Enter/Rec Adv key to run the BLDMENU procedure:

```
                          BLDMENU PROCEDURE                    OPTIONAL-(O)

          Creates the library members required to display a menu.

Menu Name ..................................................... MENU01

Name Of The Display Text Source Member ..................... MENU01DT  (O)

Library That Contains The Source Message Member ..... ULIB1

Library To Contain The Format Load Member ..... ULIB2

If The New Member Will Replace An Existing Load Member,
     Enter REPLACE .........................................          (O)

If The Load Member Is To Remain In The Output Library,
     Enter KEEP ............................................          (O)

If Creating A Free-Format Menu, Enter FREEFORM .............          (O)
```

HISTCRT Procedure

Function

The HISTCRT procedure displays selected entries from the history file on the display screen at the requesting display station. The HISTCRT procedure display is similar to the HISTORY procedure display with SYSLIST CRT specified.

The HISTCRT procedure cannot be used to display entries logged to the history overflow file. To list entries logged to the history overflow file, you must use the HISTORY procedure described later in this chapter.

For information about the entries logged to the history file and the history overflow file, see the description of the HISTORY procedure.

When the HISTCRT procedure is run, the most recent entries logged to the history file are displayed first. The following is an example of the initial HISTCRT display:

```
HISTORY SCROLL                                WKSTN   USER      JOB NAME    TIME
     // RUN                                     X2    LGA       X2095834  09.59.28
                                                X2    LGA       X2095834  09.59.30
     // DEFINEPN INAME-PHONTEMP,INLIB-#LIBRARY,
     // OUTLIB-#LIBRARY,MODE-NODELAY            X2    LGA       X2095834  09.59.31
     // END                                     X2    LGA       X2095834  09.59.32
 SYS-8621  OPTIONS (   3) PNLM                  X2    LGA       X2095834  09.59.34
                                                X2    LGA       X2095834  09.59.34
 SOURCE MEMBER PHONTEMP NOT FOUND IN SPECIFIED LIB
 3                                              X2    LGA       X2095834  09.59.39
                                                X2    LGA       X2095834  09.59.41
*EJ  09.58.34  09.59.41  00.01.07  07/15/81  LGA       X2  DEFINEPN
 HISTCRT                                        X2    LGA       X2100218  10.02.19
     // LIBRARY NAME-0                          X2    LGA       X2100218  10.02.19
     // MEMBER USER1-##MSG2                     X2    LGA       X2100218  10.02.19
     // * 5039                                  X2    LGA       X2100218  10.02.19
 HISTCRT PROCEDURE EXECUTING                    X2    LGA       X2100218  10.02.20
     // LOAD $HSML                              X2    LGA       X2100218  10.02.20
     // RUN                                     X2    LGA       X2100218  10.02.21
     // DISPLAY SYSTEM-NO                       X2    LGA       X2100218  10.02.22
     // END                                     X2    LGA       X2100218  10.02.22
                NO MORE ENTRIES IN HISTORY FILE
      CMD7-END OF JOB     CMD8-DISPLAY OLDEST     CMD9-DISPLAY NEWEST
 DIRECTION- B   SCAN DATA-                      WKSTN-     USER-
```

Scan values may be entered at the bottom of the HISTCRT display. If scan values are entered at the bottom of this display, the HISTCRT procedure searches the history file for entries matching the scan values. A scan direction of forward (F) or backward (B) can be specified; the default value is B. SCAN DATA applies only to the text portion of the history file entries; WKSTN applies only to the control information portion of the entries. The following scan values can be specified:

DIRECTION	One alphabetic character, either F for a forward scan, or B for a backward scan; the default value is B.
SCAN DATA	A string of 1 through 20 alphameric characters to be searched for within the text portion of the history file entries.
WKSTN	A 2-character, alphameric work station ID.
USER	A 1- to 8-character, alphameric user ID can be entered only if SYSTEM was specified in the HISTCRT procedure. USER does not appear if HISTCRT USER was entered.

When the Enter/Rec Adv key is pressed after keying in a scan value or values in the above fields, the HISTCRT procedure searches the history file for the specified values. If multiple scan values are specified, all of the specified conditions must be met for the scan to be successful.

If a forward scan is specified, the scan begins with the first history file entry on the currently displayed screen. If the scan is successful, a screen of history file entries is displayed with the first entry containing the specified characters. All entries that satisfy the scan are shown in high intensity and are undefined. To continue the scan operation with the same values, press the Enter/Rec Adv key. The scan continues with the next entry after the last currently displayed entry. If the specified characters are not found in the history file, a message is displayed.

If a backward scan is specified, the scan begins with the last entry on the currently displayed screen and searches backward through the history file. If the scan is successful, a screen of history file entries is displayed with the last entry containing the specified scan characters. To continue the scan in a backward direction, press the Enter/Rec Adv key. The scan continues with the entry previous to the first currently displayed entry. If the specified characters are not found in the history file, a message is displayed.

The HISTCRT procedure provides the following functions through the use of the command and function keys:

Key	Function
Roll Up	Pages forward through the history file entries. The roll factor is initially set to the number of entries that can be displayed on the screen at one time. When the last, or the most recent, entry in the history file is displayed, a "last entry" message is displayed. If the Roll Up key is used when the last entry is displayed, the current screen is redisplayed. The roll factor can be changed using Cmd key 3. If SYSTEM was specified in the HISTCRT procedure, history file entries are displayed as they are logged to the history file. The newly logged entries are marked with the symbol > when displayed.
Roll Down	Pages backward through the history file entries. The roll factor is initially set to the number of entries that can be displayed on the screen at one time. When the first, or the oldest, entry in the history file is displayed, a "first entry" message is displayed. If the Roll Down key is used when the first entry is displayed, the current screen is redisplayed. The roll factor can be changed using Cmd key 3.

Key	Function
Cmd key 3	Used to change the roll factor of the Roll Up and the Roll Down keys. The following screen is displayed when Cmd key 3 is pressed:

```
 HISTORY SCROLL                                WKSTN   USER      JOB NAME    TIME
      // RUN                                    X2    LGA       X2095834  09.59.28
                                                X2    LGA       X2095834  09.59.30
      // DEFINEPN INAME-PHONTEMP,INLIB-#LIBRARY,
      // OUTLIB-#LIBRARY,MODE-NODELAY           X2    LGA       X2095834  09.59.31
      // END                                    X2    LGA       X2095834  09.59.32
  SYS-8621  OPTIONS (   3) PNLM                 X2    LGA       X2095834  09.59.34
                                                X2    LGA       X2095834  09.59.34
  SOURCE MEMBER PHONTEMP NOT FOUND IN SPECIFIED LIB
  3                                             X2    LGA       X2095834  09.59.39
                                                X2    LGA       X2095834  09.59.41
*EJ  09.58.34  09.59.41  00.01.07  07/15/81  LGA        X2  DEFINEPN
  HISTCRT                                       X2    LGA       X2100218  10.02.19
      // LIBRARY NAME-0                         X2    LGA       X2100218  10.02.19
      // MEMBER USER1-##MSG2                    X2    LGA       X2100218  10.02.19
      // * 5039                                 X2    LGA       X2100218  10.02.19
  HISTCRT PROCEDURE EXECUTING                   X2    LGA       X2100218  10.02.20
      // LOAD $HSML                             X2    LGA       X2100218  10.02.20
      // RUN                                    X2    LGA       X2100218  10.02.21
      // DISPLAY SYSTEM-NO                      X2    LGA       X2100218  10.02.22
      // END                                    X2    LGA       X2100218  10.02.22
                    NO MORE ENTRIES IN HISTORY FILE
    -ROLL FACTOR      PRESS ENTER, ROLL UP, ROLL DOWN
```

To change the roll factor for future roll up or roll down operations, key in the new number of entries to roll, then press the Enter/Rec Adv key. Valid entries are 1 through 999. If a value is not keyed into this field, the HISTCRT procedure assumes the number of entries that can be displayed on a full screen.

To change the number of entries to roll for one operation of the roll keys only, key in the new number of entries to roll, and then use the Roll Up or Roll Down key. This option does not change the roll factor for future roll up or roll down operations. Valid entries are 1 through 999. If a value is not keyed into this field, the HISTCRT procedure assumes the number of entries that can be displayed on a full screen.

Key	Function
Cmd key 4	Used to specify whether the following control information for each entry to the history file is displayed: • Work station ID • User ID • Job name • Time If the control information is currently displayed, use Cmd key 4 to remove the control information. If the control information is not currently displayed, use Cmd key 4 to display the control information.
Cmd key 5	Used to specify whether nonviewed entries to the history file are displayed. Viewed entries are those that were displayed prior to entry into the history file. Nonviewed entries include *E entries and all OCL statements generated via procedures. If nonviewed entries are currently displayed, use Cmd key 5 to remove the nonviewed entries. If only viewed entries are currently displayed, use Cmd key 5 to also display nonviewed entries.

Key	Function
Cmd key 6	Used to define what values must be met before an entry to the history file is selected to be displayed. For example, the following screen is displayed if Cmd key 6 is used:

```
                    CHANGE SELECTION CRITERIA                    Optional-*

          Cancels or changes the scroll display selection criteria

Work station ID . . . . . . . . . . . . . . . . . . . . . . . .          *

User ID . . . . . . . . . . . . . . . . . . . . . . . . . . . .          *

Jobname . . . . . . . . . . . . . . . . . . . . . . . . . . . .          *

Start time  . . . . . . . . . . . . . . . . . . . . . . hh.mm.ss         *

Stop time . . . . . . . . . . . . . . . . . . . . . . . hh.mm.ss         *




CMD6-To cancel above selection and return to scroll display.
CMD7-To return to scroll display without any change to selection criteria.
```

Specify the following values by which the system selects the entries to be displayed:

Work station ID: A 2-character, alphameric work station ID.

User ID: A value can be entered in this field only if SYSTEM was specified in the HISTCRT procedure. If a user ID is entered in this field, the ID must consist of 1 to 8 alphameric characters.

If SYSTEM was not specified in the HISTCRT procedure, the default value for the user ID is the user ID of the requesting display station. In this case, the user ID cannot be changed, and the operator cannot display entries with other user IDs.

Job name: An 8-character, alphameric job name.

Start time: All entries made to the history file including and after the specified time will be displayed. The start time must be entered in the form hh.mm.ss, where hh is hours ($00 \leq hh \leq 23$), mm is minutes ($00 \leq mm \leq 59$), and ss is seconds ($00 \leq ss \leq 59$).

Stop time: All entries made to the history file including and before the specified time will be displayed. The stop time must be entered in the form hh.mm.ss, where hh is hours ($00 \leq hh \leq 23$), mm is minutes ($00 \leq mm \leq 59$), and ss is seconds ($00 \leq ss \leq 59$).

The above fields are initially set to null values, except for user ID. All history file entries matching the values entered above are displayed until you again use Cmd key 6 to change these values. You may enter multiple selection values; in this case, the entries must match all of these values in order to be displayed.

Key	Function
Cmd key 7	Ends the HISTCRT procedure.
Cmd key 8	Displays the oldest, or the first, history file entry.
Cmd key 9	Displays the newest, or the last, history file entry. History file entries will also be displayed as they are logged to the history file.
Cmd key 10	Used to control whether only the history file entries beginning with *EJ (end-of-job) or with *EP (end-of-print) are displayed. For information about the *E entries, refer to the description of the $HIST utility program in Chapter 4. If all history file entries, including the *E entries, are currently displayed, use Cmd key 10 to display only the *E entries. If only the *E entries are currently displayed, use Cmd key 10 to display all history file entries, including the *E entries.

The HISTCRT procedure runs the $HSML utility program. The $HSML utility program creates a workfile containing the history file entries to be displayed. The label assigned to the workfile is HSFWKxx, where xx is the work station ID of the requesting display station. The workfile has a retention of J, which specifies that the workfile will be scratched at end of job.

Command Statement Format

HISTCRT [USER / SYSTEM]

Parameters

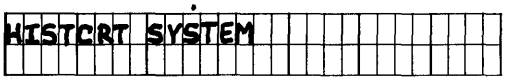
USER: Display only those entries with the user ID of the operator currently assigned to the requesting display station. If this parameter is not specified, USER is assumed.

SYSTEM: Display all entries in the system history file. If security is active, SYSTEM can be entered only by an operator with a security classification of system operator or higher. As entries are logged to the history file after the HISTCRT procedure is entered, the new entries are marked with the symbol > when displayed. This allows the operator to view the history file entries as the system processes the corresponding jobs.

Examples

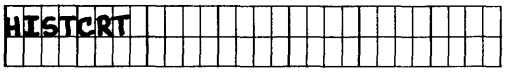
Example A

Display all history file entries:



Example B

Display only those history file entries with the user ID of the requesting display station:



HISTORY Procedure

Function

The HISTORY procedure displays selected entries from the history file or the history overflow file on the system list device assigned to the requesting display station.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

For information about assigning the system list device, see the description of the SYSLIST procedure later in this chapter.

Recorded in the history file are:

- All OCL statements, utility control statements, control commands, and procedures executed by the SSP
- All messages displayed at the display station
- All operator responses to messages and prompts

In addition to the actual text, each entry in the history file listing contains the following information:

- The user ID of the operator associated with the entry. If the entry comes from a job on the input job queue, no user ID is associated with the entry.
- The work station ID associated with the entry. If the entry comes from a job on the input job queue, no work station ID is associated with the entry.
- The job name of the job associated with the entry. If the entry is a procedure command or control command, a job name is not associated with the entry.
- The time when the entry was placed in the history file. (The time is based upon the time specified by the system operator during IPL.)

Because the history file is limited in size (the size is specified during system configuration), the number of events that can be reflected in the history file at a particular time varies with the length of the file and the length of the entries in the file.

To avoid losing history file entries after the history file has been filled, you can request during IPL or system configuration that a history overflow file be used. The overflow file contains from one to eight segments, each segment being the same size as the history file. During IPL or system configuration, you specify the size (number of segments) of the overflow file; the overflow file is allocated at that time. When less than 25 sectors remain in the history file, the system issues an informational message to the system console and copies the contents of the history file into an overflow file segment. The system operator should then use the HISTORY procedure to print or display the contents of the overflow file. When all entries in a segment are displayed and reset, the system can again use that segment for a copy of the history file.

The informational message indicating when only 25 sectors remain is not issued if an overflow file is not configured.

To reset the entire history file, the system operator can perform the following:

Enter HISTORY SYSTEM,NOLIST,RESET

If an overflow file is not used, history file entries can be lost because once the file has been filled, each new entry causes the oldest entry to be dropped from the file. When the file is listed, the oldest entry is displayed or printed first.

When the System/34 security function is active, the system console parameters (SYSTEM, OVERFLOW, USER and WORKSTN) can be used from the system console or from any display station by an operator who is classified as a system operator or higher. When the security function is not active, any or all of the HISTORY parameters can be used from any display station.

CAUTION
Care should be exercised when the RESET parameter is used because it is possible that one display station's history entries can be reset from another display station.

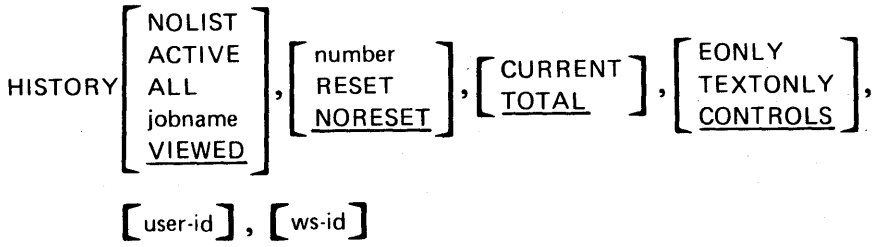
Indentation of OCL statements is as follows:

Print Pos 1: Any entries beginning with *E
Print Pos 3: Any entries that were displayed prior to entry into history file
Print Pos 7: All OCL generated via procedures
Print Pos 5: All other entries

The HISTORY procedure runs the $HIST utility program; refer to $HIST in Chapter 4 for a description of the fields listed for the *E entries.

Command Statement Formats

If run from any display station (see Note):



If run from the system console (see Note) and if entries for any display station and operator are required:

```
HISTORY SYSTEM, [NOLIST | ALL | jobname | VIEWED] , [number | RESET | NORESET] , [CURRENT | TOTAL] , [EONLY | TEXTONLY | CONTROLS] ,
[user-id] , [ws-id]
```

or

```
HISTORY OVERFLOW, [ALLOCATE,RESET | DELETE,RESET | [ALL | jobname | VIEWED | NOLIST] , [number | RESET | NORESET]] , [CURRENT | TOTAL] , [EONLY | TEXTONLY | CONTROLS] ,
[user-id] , [ws-id]
```

Note: SYSTEM, OVERFLOW, user-id, and ws-id are system parameters that can be run from any display station if security is active and the operator is the system console operator or higher. It could also be run from any display station whose security is not active.

Parameters

NOLIST: Prevents display of history data.

Notes:
1. The execution of the HISTORY procedure with the NOLIST parameter appears the same as a HISTORY execution with syslist off.
2. The NOLIST parameter is only valid with RESET.

ACTIVE: List history file entries for all active jobs associated with the requesting display station.

To list the history file for an active job, press the Attn key (Inquiry mode) and then enter HISTORY ACTIVE.

ALL: List all types of history file entries, including OCL statements generated by procedures. If ALL is not specified, only items previously displayed to the operator during execution are listed. Items previously displayed to the operator include OCL statements and messages that were logged and displayed as they were entered or issued.

jobname: If a job name is specified, only history file entries for that job are displayed.

ALLOCATE: Display and reset the entries in the overflow file (RESET must be specified with ALLOCATE). The overflow file will then be deleted and reallocated. The length of the new overflow file will be the length requested during IPL. The ALLOCATE parameter is used if you change the size of the overflow file during IPL, but the existing file still contains entries that have not been reset.

DELETE: Display and reset the entries in the overflow file (RESET must be specified with DELETE). The overflow file will then be deleted. The DELETE parameter is used if, during IPL, you request that the overflow file be deleted, but the existing file still contains entries that have not been reset.

Note: CURRENT and TEXTONLY cannot be specified with ALLOCATE or DELETE.

VIEWED: List only history file entries that are identified as having been displayed during execution.

SYSTEM: If SYSTEM is specified, history file information for the whole system can be printed, instead of just history file entries for a single display station or user.

OVERFLOW: OVERFLOW specifies that only entries from the overflow file are to be displayed. If VIEWED is not specified with OVERFLOW, entries for any display stations can be displayed. If VIEWED is specified with OVERFLOW, only the entries viewed at the system console are displayed.

Note: SYSTEM, OVERFLOW, user-id and/or ws-id can be specified from any display station if one of the following conditions is true:

- The System/34 security function is not active.
- The security function is active, and the operator is classified as a system operator or higher.

RESET: If SYSTEM is also specified, RESET specifies that any entries displayed cannot be displayed again by the HISTORY procedure. If OVERFLOW is also specified, the overflow file entries displayed cannot be displayed again by the HISTORY procedure. When all entries in an overflow file segment have been reset, that segment can again be used for a copy of the history file.

If SYSTEM is not specified, RESET specifies that any entries displayed are unavailable for display by the HISTORY procedure when it is run from the requesting display station. Those entries can still be displayed at the system console if SYSTEM is specified on a HISTORY procedure command.

NORESET: Entries that are displayed can be displayed again. If a parameter is not specified, NORESET is assumed.

number: Specifies the number of entries to be displayed. For example, if 23 is specified, the 23 most recent entries are displayed. The smallest number that can be specified is 1; the largest number, 999. If a number larger than 999 is specified, the entire history file is processed.

CURRENT: Display only those entries placed in the history file or the overflow file since the last time CURRENT was specified on a HISTORY procedure command.

TOTAL: Display all specified entries. If a parameter is not specified, TOTAL is assumed.

TEXTONLY: Display only the text of each entry. Control information (such as user ID, work station ID, job name, and date) is not displayed.

CONTROLS: Display the control information along with the text for each entry. If a parameter is not specified, CONTROLS is assumed.

EONLY: Display only end-of-job (*E) entries. This parameter should only be used when you are compiling job accounting information.

user-id: Display only those entries that have the specified user ID.

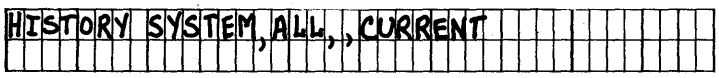
ws-id: Display only those entries that have the specified work station ID.

Note: The user-id and ws-id parameters should be used only if the history is desired for a work station other than the work station being used.

Examples

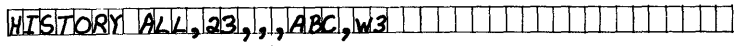
Example A

To display all history file entries that were added to the file since the last time the file was displayed, the system operator can enter the following statement:



Example B

To display the 23 most recent entries of user ABC on display station W3, an operator can enter the following statement:



INIT Procedure

Function

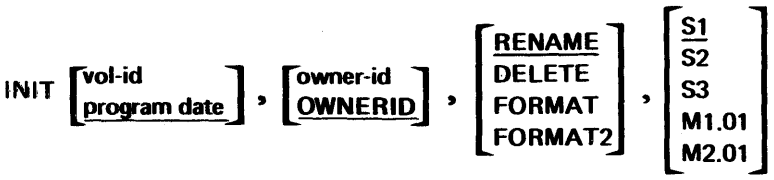
The INIT procedure prepares (initializes) a single diskette or all the diskettes in a magazine so that they can be used. The INIT procedure performs some or all of the following functions:

- Writes sector addresses on the diskette
- Checks for defective cylinders
- Assigns new cylinder numbers when a cylinder has a defective sector
- Writes an identifying name on each diskette
- Formats cylinder 0

Note: A diskette 1 diskette can be initialized on a diskette drive designed for the diskette 2D diskette, but a diskette 2D diskette cannot be initialized on a diskette drive designed for a diskette 1 diskette.

The INIT procedure runs the $INIT utility program.

Command Statement Format



Note: Parameter 4 is ignored for systems without a diskette magazine drive.

Parameters

vol-id: The volume ID of the diskette; 1 to 6 alphameric characters.

If DELETE is not specified, the vol-id is left-justified, padded with blanks, and written in the volume ID field of the diskette volume label. If vol-id is not specified, the program date is written in the volume ID field. The date is written in year-month-day format.

If DELETE is specified on the INIT statement, the vol-id is checked against the existing volume ID of the diskette to ensure that the correct diskette is inserted.

owner-id: The owner ID of the diskette; 1 through 8 alphameric characters. If RENAME, FORMAT, or FORMAT2 is specified on the INIT statement, the owner ID is left-justified, padded with blanks, and written in the owner ID field of the diskette volume label. If owner-id is not specified, OWNERID is written in the owner ID field.

RENAME: The diskette is renamed (the volume ID and owner ID fields are rewritten). Files and their labels are not affected. If a parameter is not specified in the third parameter position, RENAME is assumed.

DELETE: Delete active files, thereby making space available on the diskette (initialize cylinder 0 on the diskette).

FORMAT: For a diskette 1 diskette, the surface of the diskette is formatted in the 128-byte format. For a diskette 2D diskette, the surface of the diskette is formatted in the 256-byte format.

For information about diskette formats, see Appendix C.

Each cylinder with a surface defect is renumbered. If cylinder 0 (index track) or more than two cylinders have defects, a message is displayed and the diskette is not initialized (the diskette is not usable).

Note: All diskettes in a multivolume file must be initialized in the same format.

FORMAT2: For a diskette 1 diskette, the surface of the diskette is formatted in the 512-byte format. For a diskette 2D diskette, the surface of the diskette is formatted in the 1024-byte format.

For information about diskette formats, see Appendix C.

Each cylinder with a surface defect is renumbered. If cylinder 0 (index track) or more than two cylinders have defects, a message is displayed and the diskette is not initialized (the diskette is not usable).

Note: If FORMAT2 is specified, a basic data exchange file cannot be placed on the diskette. Also, all diskettes in a multivolume file must be initialized in the same format.

S1, S2, or S3: The individual diskette slot that contains the diskette to be initialized. If a fourth parameter is not specified, S1 is assumed.

M1.01 or M2.01: The magazine that contains the diskettes to be initialized. Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01. The diskettes in the magazine are initialized beginning with the first diskette in the magazine and continuing through position 10, or until an empty position is found in the magazine.

Examples

Example A

Rewrite the volume ID and owner ID fields in a diskette volume label:

The new volume ID is 934613, and the new owner ID is JOEDISK.

Example B

Initialize a diskette 1 diskette in 128-byte format:

The volume label is 843163, and the owner ID is OWNERID.

INSTCOPY Procedure

The INSTCOPY procedure is not documented in this manual. For information about this procedure, see the *Installation and Modification Reference Manual: Program Products and Physical Setup*.

INSTINIT Procedure

The INSTINIT procedure is not documented in this manual. For information about this procedure, see the *Installation and Modification Reference Manual: Program Products and Physical Setup*.

JOBSTR Procedure

Function

The JOBSTR procedure copies, to a specified library, a $MAINT-compatible diskette file that contains one or more procedure members and/or source members. Refer to *Storing Library Members in Diskette Files* in Chapter 6. In addition, you can specify the name of a procedure to be executed after the members in the diskette file are copied.

The JOBSTR procedure runs the $MAINT utility program.

Command Statement Format

```
JOBSTR| label, [procname] , [SAVE / NOSAVE] , [library name / #LIBRARY] , [Q / jobq prty] ,
[S1 / S2 / S3 / M1.nn / M2.nn] , [NOAUTO / AUTO]
```

Parameters

label: The label of the $MAINT-compatible diskette file.

procname: The name of a procedure to be executed after the members are copied. The procedure need not be one of the members just copied, but must be in the same library as the copied members.

SAVE: The procedure specified by procname is not deleted from the library after execution. If a third parameter is not specified, SAVE is assumed.

NOSAVE: After the procedure specified by procname is executed, it is deleted from the library.

library name: The name of the library into which the members are copied. If a library name is not specified, the system library (#LIBRARY) is assumed.

Q: The procedure specified by procname is run from the input job queue, and has a job queue priority of 3; see the description of *jobq prty*, which follows, for a description of the job queue priority. If a fifth parameter is not specified but a procname is specified, the procedure is run as part of the job containing the JOBSTR procedure.

jobq prty: Specifies the job queue priority for the job; that is, the job's order of execution from the input job queue. The job queue priority can be any decimal number from 1 through 5. The system begins running jobs in order of decreasing job queue priority. For example, all jobs with a job queue priority of 5 are run before any other jobs in the job queue. Jobs with the same job queue priority are run in the order they were placed in the job queue. Jobs with a job queue priority of 1 are the last jobs run by the system. If a fifth parameter is not specified but a procname is specified, the procedure is run as part of the job containing the JOBSTR procedure.

S1, S2, or S3: Identifies the diskette slot containing the first diskette from which members are to be copied. If a sixth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette from which members are to be copied. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, JOBSTR uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, JOBSTR uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location within the magazine and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine in the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, JOBSTR uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, JOBSTR uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a seventh parameter is not specified, AUTO is assumed.

Examples

For the following examples, a diskette file labeled JBS contains the following job stream:

```
// COPY NAME-P1,LIBRARY-P
  •
  •
  •
// CEND
// COPY NAME-P2,LIBRARY-P
  •
  •
  •
// CEND
// COPY NAME-S1,LIBRARY-S
  •
  •
  •
// CEND
```

Example A

Copy the procedure members (P1 and P2) and the source member (S1) into the system library:

```
JOBSTR JBS
```

Example B

Copy the procedure members (P1 and P2) and the source member (S1) into a user library called ULIB; then, execute P1 (P1 should be saved):

```
JOBSTR JBS,P1,,ULIB
```

KEYSORT Procedure

Function

The KEYSORT procedure sorts the index keys for a specified disk file. No other jobs can be using the disk file to be sorted. For information about sorting index keys, see the *Concepts and Design Guide*.

The KEYSORT procedure runs the $DDST utility program.

Command Statement Format

KEYSORT label, [mmddyy / ddmmyy / yymmdd] , [J / T / P]

Parameters

label: The label of the file whose index keys are to be sorted.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file. If a date is not specified, the index keys in the file with the specified label and the most recent creation date are sorted.

J, T, or P: The retention of the file. By specifying J, T, or P, the user is defining a specific file, and no further searching is done. When retention is not specified, the order of search is J first, then T, and finally P; the KEYSORT procedure is done on the first file found with name and/or date that match. The KEYSORT procedure goes to end of job after sorting one file.

Note: To keysort a T file that has the same label as a J file, T must be specified.

Example

Sort the index keys in a file called PAYRCD with a creation date of November 11, 1978:

```
KEYSORT PAYRCD,111178
```

LIBRLIBR Procedure

Function

The LIBRLIBR procedure copies non-SSP members from one library to another, optionally renaming the members, or copies non-SSP members within a library by renaming the members.

The LIBRLIBR procedure runs the $MAINT utility program.

Command Statement Format

```
LIBRLIBR from library name, [to library name / from library name] , [SOURCE / PROC / LOAD / SUBR / LIBRARY] ,
{name / name,ALL / ALL} , [new name] , [REPLACE]
```

Parameters

from library name: The name of the library from which member(s) are to be copied.

to library name: The name of the library to which member(s) are to be copied. If the second parameter is not specified, the from library name is assumed.

Note: If the from library name and the to library name parameters are the same, the new name parameter must be specified.

SOURCE: Source member(s) are to be copied. If the third parameter is not specified, SOURCE is assumed.

PROC: Procedure member(s) are to be copied.

LOAD: Load member(s) are to be copied.

SUBR: Subroutine member(s) are to be copied.

LIBRARY: All types of members (SOURCE, PROC, LOAD, and SUBR) are to be copied.

name: The name of a non-SSP member to be copied.

name,ALL: All non-SSP members with names beginning with the indicated characters (name) are to be copied. Up to 7 characters can be specified.

ALL: All non-SSP members are to be copied. ALL is not allowed if the to library and the from library are the same.

new name: Specifies the new name(s) of the copied member(s). new name must be specified if the to library and the from library are the same. new name must not be specified if ALL was specified as the fourth parameter. If name,ALL was specified as the fourth parameter, new name must contain the same number of characters as name (the remaining characters in each member name are unchanged).

REPLACE: Replace the library member specified, if one exists.

If REPLACE is not specified, members are placed in the library until a duplicate is found, at which time the system displays a message telling the operator that a duplicate exists. In response to the message, the operator can either cancel the job or continue processing. If the job is continued, the new member replaces the existing member in the library. If other duplicates are found during the job, then existing members are automatically replaced and no messages are displayed regarding the duplicate members.

If REPLACE is specified, new members replace existing duplicate members in the library, and no messages regarding the new members are displayed.

Example

Copy all source members whose names begin with TEST from a library called TESTLIB into a library called ACTLIB. The new members should be given names beginning with GOOD:

```
LIBRLIBR TESTLIB,ACTLIB,,TEST,ALL,GOOD
```

LINES Procedure

Function

The LINES procedure modifies the number of lines per page, the horizontal print density, and the vertical print density for printed output from a display station session. A job placed on the input job queue from a display station uses the lines-per-page and the characters-per-inch specifications that were in effect when the job was placed on the queue.

The LINES procedure generates a FORMS OCL statement.

Command Statement Format

LINES [number / 66] , [cpi value] , [lpi value]

Parameters

number: The number of lines per page. The maximum number of lines that can be specified per page is 112. The lines-per-page specification remains in effect until a FORMS statement is used, the LINES procedure is used again, the SET procedure or $SETCF utility program is used to change the number of lines per page, or the display station session ends. If number is not specified, 66 is assumed.

For SSP utility programs, the printer skips (overflows) to a new page when six less than the number of lines specified are printed. For example, if LINES-84 is specified, the printer skips to a new page after printing line 78. If LINES-13 is specified, one line is printed per page. When 12 or less lines are specified, printing is on every line (no overflow).

For user programs, the SSP indicates that an overflow condition occurred when six less than the number of lines specified were printed.

Note: RPG II programs can specify an overflow value that overrides the value specified by the LINES procedure. If a line counter specification is used in an RPG II program, the specification remains in effect only for the duration of that program.

cpi value: The cpi (characters per inch) value parameter specifies the horizontal print density used for printed output from the display station session. The values that can be specified are 10 or 15. If a value of 15 is used, the output is identified as printer dependent and should be printed on a 5225 (Models 1 through 4) Printer or a 5224 (Models 1 and 2) Printer. If a value of 15 is attempted on another printer, an error message will appear.

The cpi value parameter remains in effect until changed by a FORMS statement, a PRINTER statement, or the LINES procedure; or until the display station session is ended. If the cpi value is not specified, the horizontal print density is not changed.

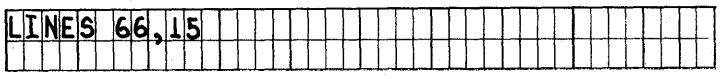
lpi value: The lpi (lines per inch) value parameter specifies the vertical print density to use for printed output from the display station session. The values that can be specified are 4, 6, and 8. If the output is printed on a printer other than a 5225 Printer or a 5224 Printer, the lpi value is ignored.

The lpi value remains in effect until changed by another FORMS statement or the LINES procedure, or until the display station session is ended. The PRINTER statement can be used to change the lpi value for a particular print job, but it stays in effect only for that print job. If the lpi value is not specified and the lines per inch was not previously set during the session, the system uses the value that was set during work station configuration (either 4, 6, or 8). If no lpi value was specified during work station configuration, the system uses an lpi value of 6.

Note: For 5225 Printer Models 11 and 12 and the 5224 Printer Model 12, an lpi value of 8 will cause characters to vertically overlap from one line to another.

Example

Change the number of lines per page to 66, and the horizontal print density to 15:



LISTFILE Procedure

Function

The LISTFILE procedure lists the contents of a specified file. The format of the listing depends upon the type of the file being processed.

The listing for a COPYFILE, an EXCHANGE, an IFORMAT, a LIBRARY, or a LIBRFILE type of file is displayed on the system list device assigned to the requesting display station. For information about assigning the system list device, see the description of the SYSLIST procedure later in this chapter.

The listing for an APARFILE, a BACKUP, or a PROFILE type of file is displayed either on the system printer or on the display screen at the requesting display station. After entering the LISTFILE procedure, the operator is prompted for the destination of the procedure's listing, either PRINTER or CRT.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

The LISTFILE procedure runs the $BACK, $BICR, $COPY, $FEDMP, or $MAINT utility program. (The $FEDMP utility is run by the DUMP SSP service procedure, which is described in Appendix D.)

Command Statement Format

```
                                  [APARFILE]                  [NOAUTO]
                                  [BACKUP  ]                  [AUTO   ]
                                  [COPYFILE]                  [       ]
                                  [EXCHANGE]                  [       ]
                                  [IFORMAT ]                  [       ]
                                  [LIBRARY ]                  [       ]
                                  [LIBRFILE]                  [       ]
                                  [PROFILE ]                  [       ]
                                  [       ]                   [S1     ]
                                  [       ]                   [S2     ]
                                  [       ]                   [S3     ]
                                  [       ]                   [M1.nn  ]
                                  [       ]                   [M2.nn  ]
                     [mmddyy]     [F1]                        [       ]
                     [ddmmyy]     [I1]                        [       ]
LISTFILE label,      [yymmdd]     [  ],                       [  ],
```

Note: The fifth and sixth parameters are ignored if F1 is specified or assumed as the third parameter or if the system does not have a diskette magazine drive.

Parameters

label: The label of the file to be displayed.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file to be displayed. For a disk file, the specified date must be in the same format as the session date. If the creation date for a disk file is not specified, and more than one file exists with the specified label, the file with the latest creation date is listed. For a diskette file, the specified date must be in the same format as the creation date of the diskette file. If the creation date for a diskette file is not specified, and more than one file exists with the specified label, the first file in the diskette VTOC is processed.

F1: A disk file is listed. If no parameter is specified in the third parameter position and either LIBRARY, LIBRFILE, or COPYFILE is specified in the fourth parameter position, F1 is assumed.

I1: A diskette file is listed. If no parameter is specified in the third parameter position and any parameter other than LIBRARY, LIBRFILE, or COPYFILE is specified in the fourth parameter position, I1 is assumed.

APARFILE: List a diskette file created by the APAR SSP service procedure. If APARFILE is specified, I1 must be specified in the third parameter, and the fifth and sixth parameters are ignored. The diskette containing the file must be in diskette slot S1.

The listing is a series of displays in the formats produced by the following series of DUMP SSP service procedure commands:

```
DUMP CONFIG,,I1
DUMP PTF,,I1
DUMP TRACE,,I1
DUMP MAIN,,I1
DUMP CONTROL,,I1
DUMP IOC,,I1
```

BACKUP: List a diskette file created by the $BACK utility. If BACKUP is specified, I1 must be specified in the third parameter, and the fifth and sixth parameters are ignored. The diskette containing the file must be in diskette slot S1.

The listing is in the format produced by the following system service procedure command:

```
DUMP DISK,,I1
```

COPYFILE: List a disk file that is not a library, or list a diskette file created by the $COPY utility. If a parameter is not specified in the fourth parameter position, COPYFILE is assumed.

Note: When the system list device is the display screen, you control the record display in the following manner:

- To display the next record, press Cmd key 1.
- To display the previous record, press Cmd key 2. (Cmd key 2 is ignored if the file being displayed is a diskette file or if the displayed record is the first record in the file.)
- To terminate the display, press Cmd key 3.
- To display the next bytes of the displayed record, press the Shift and the Roll Up keys.
- To display the previous bytes of the displayed record, press the Shift and the Roll Down keys.

EXCHANGE: List a basic data exchange diskette file. The third parameter must be I1 if EXCHANGE is specified. The listing is in the same format as a sequential file listing produced by the $COPY utility.

IFORMAT: List an I exchange diskette file. The third parameter must be I1 if IFORMAT is specified. The listing is in the same format as a sequential file listing produced by the $COPY utility.

LIBRARY: List a library on disk. The listing is in the same format as a listing produced by the following SSP procedure command:

```
LISTLIBR DIR,LIBRARY, library name
```

LIBRFILE: List a disk or diskette file that (1) contains a library member (or members) copied to the file by the FROMLIBR procedure or by the $M^INT utility program, or (2) contains a user-created record mode file. The listing indicates the type (source, procedure, load, or subroutine) and name of each member listed. This listing is the same as the listing produced by the $MAINT utility.

PROFILE: List a copy of the system security profile that was created by the $PRSV utility program. If PROFILE is specified, I1 must be specified in the third parameter, and the fifth and sixth parameters are ignored. The diskette containing the file must be in diskette slot S1.

The listing is in the format produced by the following system service procedure command:

```
DUMP DISK,,I1
```

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a fifth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, LISTFILE uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed. The operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, LISTFILE uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, LISTFILE uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

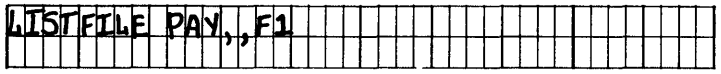
- If M1.nn or M2.nn is also specified, LISTFILE uses both magazine slots. Processing begins with the diskette in the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a sixth parameter is not specified, AUTO is assumed.

Examples

Example A

List the contents of a disk file called PAY:



Example B

List the contents of a user library called ULIB1:

LISTLIBR Procedure

Function

The LISTLIBR procedure lists, on the system list device assigned to the requesting display station, the contents of a specified library. Either directory entries or the contents of individual members can be listed.

For information about assigning the system list device, see the description of the SYSLIST procedure later in this chapter.

The LISTLIBR procedure runs the $MAINT utility program.

Notes:

1. If the display screen is used for listing the library, only the first 80 bytes of each LISTLIBR output line are displayed. To ensure that all the information in a library member or directory entry is listed, use a printer to list the output. The STATUS control command can be used to determine where output from SSP utility programs is being listed for a display station (that is, what the current system list assignment is), and the SYSLIST procedure can be used to change the current system list assignment.
2. If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

Command Statement Formats

For listing non-SSP directory entries:

```
                 [SOURCE  ]
                 [PROC    ]
LISTLIBR DIR,    [LOAD    ] , [library name]
                 [SUBR    ]   [#LIBRARY]
                 [LIBRARY ]
```

For listing status information from a library directory:

```
LISTLIBR DIR,SYSTEM, [library name]
                     [#LIBRARY]
```

For listing non-SSP library members and their directory entries:

```
                                [SOURCE  ]
                                [PROC    ]
LISTLIBR {member name}          [LOAD    ] , [library name]
         {name,ALL   }          [SUBR    ]   [#LIBRARY]
         {ALL        }          [LIBRARY ]
```

Note: For a description of a library status and a library directory listing, see *Copy Functions* under the $MAINT utility program in Chapter 4.

Parameters

DIR: A directory entry is to be listed.

member name: The name of the non-SSP library member to be listed.

name,ALL: Specifies the beginning characters of the non-SSP library member names to be listed. Up to 7 characters can be used.

ALL: Specifies that all non-SSP members of the specified type(s) are to be listed.

SYSTEM: List status information for a library. SYSTEM is valid only with DIR.

SOURCE: List source directory entries if DIR is specified; otherwise, list the contents of source member(s). If the second parameter is not specified, SOURCE is assumed.

PROC: List procedure directory entries if DIR is specified; otherwise, list the contents of procedure member(s).

LOAD: List load directory entries if DIR is specified; otherwise, list the contents of load member(s).

SUBR: List subroutine directory entries if DIR is specified; otherwise, list the contents of subroutine member(s).

LIBRARY: List status information and all types of directory entries (SOURCE, PROC, LOAD, and SUBR) if DIR is specified; otherwise, list the contents of all member types (SOURCE, PROC, LOAD, and SUBR).

library name: The name of the library containing the member(s) to be listed. If library name is not specified, the system library (#LIBRARY) is assumed.

Examples

Example A

List the procedure member JOE in the user library called USER1:

```
LISTLIBR JOE,PROC,USER1
```

Example B

List all system library non-SSP procedure members that have names beginning with PA:

```
LISTLIBR PA,ALL,PROC
```

Example C

List status information and the source, procedure, load, and subroutine directory entries for non-SSP members in the system library:

```
LISTLIBR DIR,LIBRARY
```

LOG Procedure

Function

If single-program mode without print spooling was specified during system configuration, the LOG procedure tells the system where to display messages and OCL statements and whether to skip to line 1 of the next form at end of job. The LOG procedure can be run only from the system console. If the LOG procedure (or a LOG OCL statement) is not used, the system displays messages and OCL statements only on the system console display screen.

If single-program mode was not specified during system configuration or if spooling was specified, the LOG procedure is ignored, and the system displays messages and OCL statements only on:

- The display screen at the requesting display station, unless the display station is released, the executing program is an MRT (multiple requestor terminal) program, or the job is run from the input job queue
- The system console display screen, if the requesting display station was released, the program is an MRT program, or the job is run from the input job queue

Command Statement Format

LOG [CRT / PRINTER] , [EJECT / NO EJECT]

Parameters

CRT: Use only the system console display screen. If the first parameter is not specified, CRT is assumed.

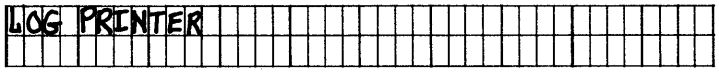
PRINTER: Use the system printer and the system console display screen.

EJECT: Skip to line 1 of the next page at end of job. If the second parameter is not specified, EJECT is assumed.

NOEJECT: Do not skip to line 1 of the next page at end of job.

Example

Display messages and OCL statements on both the system console display screen and the printer:



ORGANIZE Procedure

Function

The ORGANIZE procedure performs one of the following functions:

- Copies a disk file to another area on the disk
- Copies a disk file to another area on the disk and deletes specified records
- Copies a disk file to a diskette
- Copies a disk file to a diskette and deletes specified records

If the input file is sequential, the output file is sequential. However, if a sequential input file is reorganized, records should be specified for deletion. If the input file is indexed, the output file is indexed, and the data records in the output file are in the same sequence as the keys in the index. A direct input file cannot be reorganized. The ORGANIZE procedure cannot process a file that is being used by any other job on the system.

The ORGANIZE procedure runs the $COPY utility program.

Command Statement Formats

For reorganizing a disk file as another disk file:

```
                                                  [S]   [position ,character]
                                                  [J]   [SYSDEL ,
ORGANIZE label1, [mmddyy] ,F1,label2,            [T]   [          ,SYSDEL]
                 [ddmmyy]                         [P]   [SYSDEL ,SYSDEL]
                 [yymmdd]
```

For reorganizing a disk file as a diskette file:

```
                                                                  [retention days]
ORGANIZE label1, [mmddyy] , [I1] ,vol-id,                         [      1       ] ,
                 [ddmmyy]
                 [yymmdd]

                                                  [S1    ]
                                                  [S2    ]
                                                  [S3    ]
                                                  [M1.nn ]
                                                  [M2.nn ]
                                                                  [NOAUTO]
                                                                  [AUTO  ]
```

Note: Parameters 8 and 9 are ignored for systems without a diskette magazine drive.

Parameters

label1: The label of the disk file to be reorganized (and the label of the diskette file created if it is reorganized as a diskette file).

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the input file. If this parameter is omitted, and if more than one file exists with the specified label 1, the most recently created file is reorganized.

F1: The disk will contain the reorganized copy.

I1: The diskette will contain the reorganized copy. If a parameter is not specified in the third parameter position, I1 is assumed.

label2: The label of the disk file to contain the reorganized copy.

vol-id: The volume ID of the diskette that will contain the reorganized copy; 1 to 6 alphameric characters.

S: The new disk file is a scratch file. (The file does not exist after ORGANIZE terminates.)

J: The new disk file is a job file.

T: The new disk file is a temporary file. If F1 is specified and S, J, T, or P is not specified, T is assumed.

P: The new disk file is a permanent file. For more information on diskette file retention, see *File Statement (for Diskette Files)*, RETAIN parameter, in Chapter 1.

retention days: The length of the retention period (0 to 999 days) for the new diskette file. If I1 is specified or assumed and retention days is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file. For more information on diskette file retention, see *File Statement (for Diskette Files)*, RETAIN parameter in Chapter 1.

position: Delete records having the specified character (character) in the position specified. position can be any position in the record (first position is 1, second position is 2, and so on) to a maximum of 999. Deleted records are not copied to the reorganized file.

Note: If the input file is sequential, record deletion should be specified. If record deletion is not specified, an error message is displayed and the operator can continue the job or cancel the job.

character: Can be any one or two of the standard characters, or the characters Xdd or Xdddd, where X is a constant and dd or dddd is the hexadecimal equivalent of the character or characters. Records containing the specified character in the position specified by the position parameter are not copied to the reorganized file.

SYSDEL: Permits the removal of deleted records during file reorganization. You can specify SYSDEL only for a delete-capable file on a system configured with extended disk data management.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be used for output. If an eighth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be used for output. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, ORGANIZE uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes are required; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, ORGANIZE uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes are required, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, ORGANIZE uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes are required, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, ORGANIZE uses both magazine slots. Processing begins with the diskette in the specified location and continues through the diskette in location M2.10. If more diskettes are required, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

Each time the ORGANIZE procedure is run, the first diskette to receive output must be in the specified location. If the diskette in that location is not capable of receiving the output, the system will not search the following slots for a valid diskette. Instead, an error message will be displayed and the operator will have to insert a diskette that is capable of receiving the output into the specified slot.

If a ninth parameter is not specified, AUTO is assumed.

Examples

Example A

Reorganize the indexed file, PAYROLL, into a permanent disk file called PAYR:

```
ORGANIZE PAYROLL,,F1,PAYR,P
```

Example B

Reorganize the file called JOE and place the organized copy (except records containing a D in record position 13) on diskette volume 123456:

```
ORGANIZE JOE,,,123456,999,13,D
```

In this example, neither F1 nor I1 is specified in the third parameter position, so the default is I1. Also, the file is to be retained permanently, so a retention period of 999 days is specified.

Note: A date is not specified in either example. Consequently, if more than one file labeled JOE or PAYROLL exists on the disk, the most recently created of the files labeled JOE or PAYROLL is reorganized.

OVERRIDE Procedure

Function

The OVERRIDE procedure is used to (1) override BSC parameters specified in the user program without recompiling the program, or (2) specify parameters that cannot be specified by the user program:

Item	Parameter
Local BSC switched line ID	LOCID
Remote BSC switched line ID	REMID
Tributary station address	ADDR
Line type	LINE
Switch type	SWTYP
Wait time	WAIT
Blank compression	BLANK
Record separator	RCSP
3740 multiple file support	MLTFL
Line number	LNUM

Note: The OVERRIDE procedure *does not* override the parameters listed when it is run from a display station in inquiry mode.

Additional communications adapter items for BSC that can be altered are included in the ALTERBSC procedure. To identify the current values of these parameters, use the STATUS control command.

Running OVERRIDE alters the values only in the communications configuration record for the requesting display station and affects only the BSC programs that are loaded from the display station from which OVERRIDE is run.

If the line type to be used by an SSP-ICF BSC subsystem is different from the line type specified during hardware configuration, the OVERRIDE procedure must be run on the system console to override the hardware line type before the SSP-ICF BSC subsystems. However, if LINE-S is specified, the SWTYP parameter is required. The SWTYP parameter is actually used only by batch BSC and not by SSP-ICF BSC.

Changes made by OVERRIDE remain in effect until:

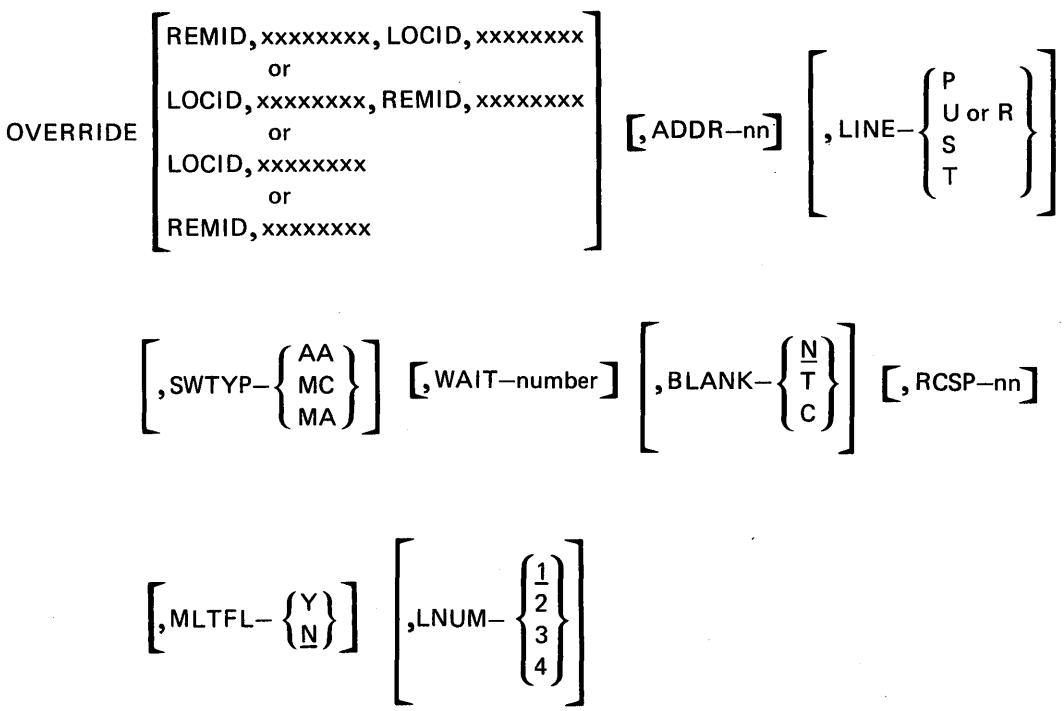
- The items are changed by the $SETCF utility program or the OVERRIDE procedure.

- The system library is reloaded. When the system library is reloaded:
  - LINE is set to the value specified during system configuration.
  - REMID, LOCID, ADDR, SWTYP, WAIT, and RCSP are all set to use the values specified in the executing program.
  - Blank compression is not used (BLANK-N).
  - 3740 multiple files are not used (MLTFL-N).

Note: The OVERRIDE procedure is intended only for data communications programming that uses BSC. For information about BSC, see the *General Information—Binary Synchronous Communications Manual.*

The OVERRIDE procedure runs the $SETCF utility program.

Command Statement Format



Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If LINE-S is specified, SWTYP must be specified; if SWTYP is specified, LINE-S must be specified unless SLINE-Y was previously specified on an ALTERBSC procedure command or a SETB utility control statement of the $SETCF utility. Although SWTYP is required with LINE-S, SSP-ICF BSC does not use its value.

Parameters

REMID,xxxxxxxx,: xxxxxxxx is the hexadecimal equivalent of the remote station switched line ID. Either 2, 4, 6, or 8 hexadecimal digits can be specified. If the remote station's ID is longer than 4 characters, it must be specified in the user program. If REMID is not specified, the value in the user program is used.

Note: If the LOCID or REMID parameter is specified, it must appear before any other parameters.

LOCID,xxxxxxxx,: xxxxxxxx is the hexadecimal equivalent of the local station switched line ID. Either 2, 4, 6, or 8 hexadecimal digits can be specified. If the remote station's ID is longer than 4 characters, it must be specified in the user program. If LOCID is not specified, the value in the user program is used.

Note: If the LOCID or REMID parameter is specified, it must appear before any other parameters.

ADDR: The hexadecimal equivalent of one of the pair of tributary station addressing characters. If ADDR is not specified, the value in the user program is used.

For the polling and addressing characters for System/34 tributary stations, see Appendix G.

LINE: Specifies the line facility:

- LINE-P specifies point-to-point nonswitched line.
- LINE-U or LINE-R indicates that the line type specified in the user program should be used.
- LINE-S specifies point-to-point switched line.
- LINE-T specifies tributary station on a multipoint line.

If the LINE parameter is not specified, the value in the communications configuration record remains unchanged.

SWTYP: The switch type:

- SWTYP-AA specifies that if a switched line (LINE-S or SLINE-Y) is specified and the modem is in automatic-answering mode, then System/34 automatically answers the call.
- SWTYP-MC specifies that if a switched line (LINE-S or SLINE-Y) is specified, the System/34 operator initiates the call manually.
- SWTYP-MA specifies that if a switched line (LINE-S or SLINE-Y) is specified, the System/34 operator answers the call manually.

If the SWTYP parameter is not specified, the value specified in the user program is used.

If you are using SSP-ICF BSC and if LINE-S is specified, the SWTYP parameter is required; however, its value is not used.

WAIT: The number of seconds that BSC will wait with no message being sent or received before it indicates that a permanent error occurred. Any decimal number from 1 through 999 can be specified. If this parameter is omitted, zero is written in the display station communications configuration record. Zero indicates that the value for wait time is taken from the user program.

BLANK: Specifies if 3780-format blank compression or truncation is used:

- BLANK-N specifies no blank compression or truncation. If BLANK is not specified, BLANK-N is assumed.
- BLANK-C specifies blank compression.
- BLANK-T specifies blank truncation only.

RCSP: RCSP specifies the record separator.

If RCSP is not specified and if the user program is an RPG II program, the following statements are true:

- If BLANK-C or BLANK-T is specified, a record separator of hex 1E is used.
- If BLANK-C or BLANK-T is not specified, no record separator is used.

If RCSP is not specified and if the user program is an assembler program, the following statements are true:

- If a record separator is specified in the user program, that record separator is used.
- If BLANK-C or BLANK-T is specified and a record separator is not specified in the user program, a record separator of hex 1E is used.
- If BLANK-C or BLANK-T is not specified and a record separator is not specified in the user program, no record separator is used.

MLTFL: MLTFL-Y specifies that 3740 multiple files can be transmitted or received with null records separating the files. MLTFL-N specifies that 3740 multiple files are not supported. If the MLTFL parameter is not specified, MLTFL-N is assumed.

LNUM: Specifies the communications line to which this override applies.

Example

Override the current line and switch type. The new line type is point-to-point switched and the new switch type is manual call:

Note: This example also indicates blank compression and that 3740 multiple files are not used. Record separator, wait time, address, local ID, and remote ID must be specified in the user program if they are to be used.

POST Procedure

Function

The POST procedure copies disk files to basic data exchange diskette files or copies special E format diskette files (created by the IBM 5260 Retail System) to disk files. POST can:

- Add a diskette file that is in special E format to an existing sequential disk file
- Convert a special E format diskette file to a disk sequential or indexed file
- Convert a disk file to a basic data exchange diskette file (basic data exchange files are sequential files), or add records from a disk file to an existing basic data exchange diskette file

For information about the basic data exchange format and the special E format, see Appendix C.

Note: Because POST moves only special E format diskette files, POST cannot be used to move files between the disk and diskette 1 diskettes (one-sided diskettes) initialized in the 512-byte format or to move files between the disk and diskette 2D diskettes (two-sided, double-density diskettes) initialized in the 1024-byte format. If the diskette format is not known, you can use the CATALOG procedure to list the diskette VTOC. This listing shows whether the diskette was initialized in the 128-, 256-, 512-, or 1024-byte format.

When a special E format diskette file is added to an existing disk sequential file, the record length of the disk file is used for all records added to the file. When a special E format diskette file is converted to a disk sequential or indexed file, records are placed in the disk file sequentially, using the record length of the diskette file.

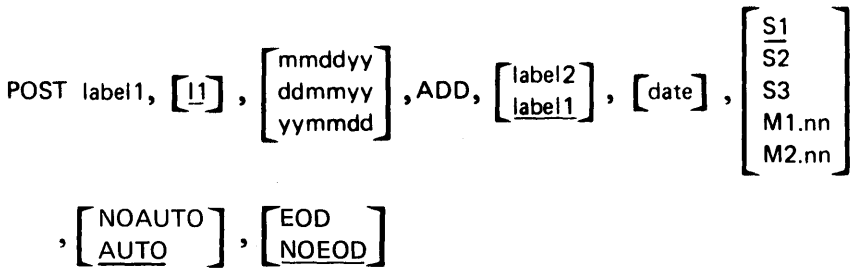
A disk file to be converted by POST to a basic data exchange diskette (always sequential) can be a sequential, indexed, or direct file. If the record length of the disk file is greater than 128 bytes (or 256 bytes for the two-sided, double-density diskettes), all records are truncated to 128 (or 256) bytes.

For an example of converting a source member or a procedure member to a diskette file in 128-byte basic data exchange format, see the description of the $MAINT utility program in Chapter 4.

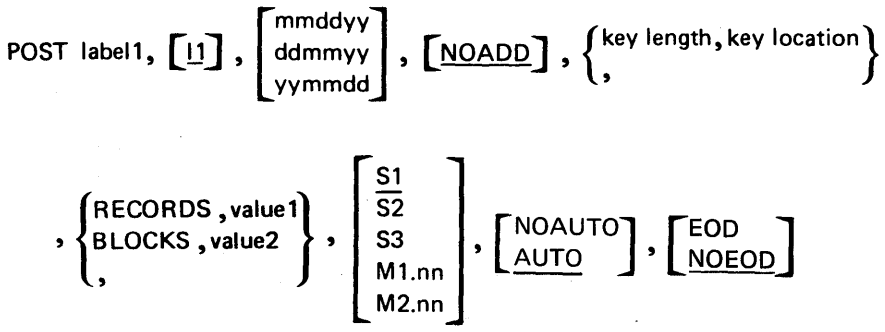
The POST procedure runs the $POST utility program.

Command Statement Formats

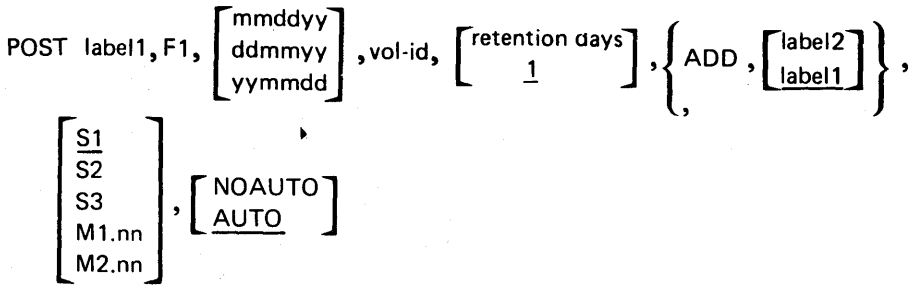
For copying a file from diskette to an existing disk file:



For copying a file from diskette to a new disk file:



For copying a disk file to a new diskette file or for adding a disk file to an existing diskette file:



Note: The location and AUTO/NOAUTO parameters are ignored for systems without a diskette magazine drive.

Parameters

label1: The label of the file being copied. If a new file is being created, it is given the label specified by label1.

I1: A special E format diskette file is being copied to a disk sequential or indexed file. If the second parameter is not specified, I1 is assumed.

F1: A disk file is being copied to a basic data exchange diskette file.

mmddyy or ddmmyy or yymmdd: The creation date of the file being copied. If more than one file with the specified label 1 resides on disk and no date is specified, the file with the specified label and the most recent date is copied to diskette.

ADD: If the second parameter is I1 and the fourth parameter is ADD, the records in a diskette file are added to the records in an existing disk sequential file. The first record from the diskette file is placed after the last record in the disk file.

If the second parameter is F1 and the sixth parameter is ADD, the records in a disk file are added to an existing basic data exchange file on diskette. The first record from the disk file is placed after the last record in the diskette file.

label2: The label of the existing disk file to which a special E format diskette file is to be added or the label of a basic data exchange diskette file to which records from a disk file are to be added. label2 is valid only if ADD is specified. If label2 is omitted, label1 is assumed.

date: The creation date of the existing disk file. date is valid only if ADD is specified. The date must be specified in the same format as the program date.

NOADD: The basic data exchange diskette file being transferred will become a new disk file with label1 as the label. NOADD is assumed whenever a file is copied from diskette to disk.

key length: The key length for the disk indexed file that is being created. key length can be any decimal number from 1 through 29. It must be specified with key location, and the sum of key length and key location must not exceed the record length plus 1.

key location: The relative displacement of the start position of the record keys for an indexed disk file that is being created. key location can be any decimal number from 1 through 128 for diskette 1 or from 1 through 256 for diskette 2D. It must be specified with key length, and the sum of key length and key location must not exceed the record length plus 1.

RECORDS,value1: Specifies that the disk file being created must be large enough to contain the number of records specified by value1.

Note: Either RECORDS,value1, or BLOCKS,value2 (see following) is required if (1) a multivolume file is being copied or (2) the created disk file is to be larger than the file being copied.

BLOCKS,value2: Specifies that the disk file being created must be large enough to contain the number of blocks specified by value2.

Note: Either BLOCKS,value2 or RECORDS,value1 (see preceding) is required if (1) a multivolume file is being copied or (2) the created disk file is to be larger than the file being copied.

vol-id: The volume ID for the diskette; 1 through 6 alphameric characters.

retention days: The length of the retention period (0 through 999 days) for the created basic data exchange diskette file. If the retention period is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file. For more information on diskette file retention, see *File Statement (for Diskette Files), RETAIN parameter,* in Chapter 1.

S1, S2, or S3: Identifies the diskette containing the first diskette to be processed. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, POST uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, POST uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

EOD: Specifies that the special end-of-data condition (see the note that follows) is to be detected when a special E format diskette file is being copied or added to a disk file.

NOEOD: Specifies that the special end-of-data condition (see the note that follows) is not to be detected when a special E format diskette file is being copied or added to a disk file. If neither EOD nor NOEOD is specified, the NOEOD condition is assumed.

Note: The 5260 Retail System indicates the end-of-data condition with a record of blanks. If EOD is specified when a basic exchange diskette file is being copied, data may be lost.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, POST uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskette. Processing resumes with the diskette in slot S1.

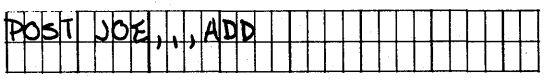
- If M1.nn or M2.nn is also specified, POST uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazine. Processing resumes at location M1.01 and continues through M2.10.

If a parameter is not specified, AUTO is assumed.

Examples

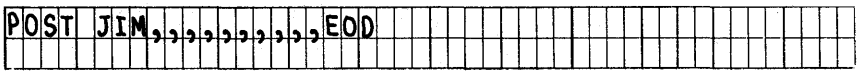
Example A

Add a special E format diskette file labeled JOE to an existing disk file labeled JOE:



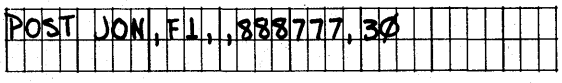
Example B

Create a disk sequential file labeled JIM from a special E format diskette file labeled JIM, and check for special end-of-data.



Example C

Create a basic data exchange diskette file labeled JON on a diskette with a volume label of 888777 from a disk file labeled JON. The file is to be saved for 30 days.



PRESTOR Procedure

Function

The PRESTOR procedure copies the system security files from a diskette onto the disk. (The PRSAVE procedure can be used to copy the system security files from disk to diskette.) PRESTOR can be run only by the system master security officer from the system console in display station mode. No other programs can be running when the PRESTOR procedure is run. The system security files must have been allocated via the PROF and PRSRC procedures and must be large enough to contain the system security files being copied from diskette. If the security system files already on disk are too small, you can use the PROF and PRSRC procedures to increase the size of the files.

If both the password security file and the resource security file are defined on disk and in the diskette file, both files are restored. If only the password security file is defined on disk, only the password security file will be restored even though the diskette file may also contain a copy of a resource security file. If both files are defined on disk but only the password security file is in the diskette file, a warning message is displayed. The operator can then choose to restore just the password security file or to cancel $PRST without restoring either file.

For information about the system security files and the operator classes that can be assigned, see the description of security in the *Installation and Modification Reference Manual.*

The PRESTOR procedure runs the $PRST utility program.

Command Statement Format

PRESTOR vol-id,label,[date]

Parameters

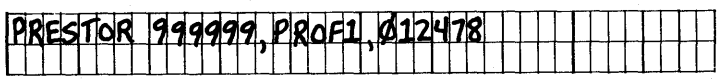
vol-id: The volume ID of the diskette; 1 through 6 alphameric characters.

label: The label of the diskette file containing the security files.

date: The creation date of the diskette file. If date is not specified and two or more files on the diskette have the specified label, the first file with the specified label is copied to the disk.

Example

Restore the system security files from a diskette file labeled PROF1. The PRSAVE procedure was used to save that file on diskette 999999. The file creation date is January 24, 1978:



PRLIST Procedure

Function

The PRLIST procedure provides a printout of the security file in four formats. First is a listing of system security (user IDs, passwords, and other sign-on information). The remaining formats are listings of the resource security by user ID, owner ID, or resource name.

The PRLIST procedure runs the $PRLT utility program.

Note: If an attempt is made to print output with ideographic headings at a nonideographic printer, blanks are printed in place of the ideographic characters.

Command Statement Formats

The SSP procedure command format to list system security is:

```
PRLIST SYSTEM,[NOPW]  ,[NOSORT]
              [PW  ]  ,[SORT  ]
```

The SSP procedure command format to list resource security is:

```
PRLIST RESOURCE,[RNAME  ]  ,[ALL ]  ,[NOSORT]
                [OWNERID]  ,[USER]  ,[SORT  ]
                [USERID ]
```

Parameters

NOPW: Do not display passwords.

PW: Display passwords to the level of the requestor.

RNAME: List secured resources by resource name.

OWNERID: List secured resources by owner IDs.

USERID: List secured resources by user ID.

ALL: List all entries. User must be a master security officer or a security officer.

USER: List only entries for current user.

NOSORT: Do not sort entries before displaying listing.

SORT: Sort entries in alphabetical order before displaying listing.

Example

To list all of the resource security entries in alphabetical order by user ID, the master security or security officer enters:

```
PRLIST RESOURCE,USERID,ALL,SORT
```

PRMENU Procedure

Function

The PRMENU procedure is used to assign a menu to a user and, optionally, to restrict that user to requesting jobs from only that menu. PRMENU can be run only by the master security officer or another designated security officer and only from the system console when it is being used as a display station.

The *Installation and Modification Reference Manual* describes the menu security display and the operator actions required.

The PRMENU procedure runs the $PRMN utility program.

Command Statement Format

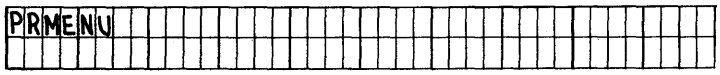
PRMENU

Parameters

None

Example

To run the PRMENU procedure, the security officer enters:



PROF Procedure

Function

The PROF procedure is used by the master security officer to activate or de-activate password or badge security and to define, modify, delete, or increase the size of the password security file. Other security officers designated by the master security officer can use the PROF procedure to modify the password security file. The PROF procedure can be run only from the system console when it is being used as a display station.

CAUTION

When the size of the password security file is changed, the file is reallocated. The contents of the file are lost. Therefore, you should save the contents of the file on a diskette before reallocating the file. You can use the PRSAVE procedure to save the contents of the password security file. You can use the PRESTOR procedure to place the contents in the newly allocated password security file.

The *Installation and Modification Reference Manual* describes the password security file and the operator classes that can be defined. That appendix also describes the displays issued by the PROF procedure and the operator actions required.

The PROF procedure runs the $PROF utility program.

Command Statement Format

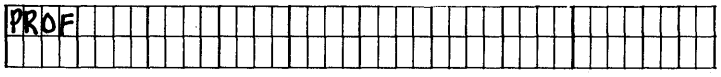
PROF

Parameters

None

Example

To run the PROF procedure, the security officer enters:



PRSAVE Procedure

Function

The PRSAVE procedure saves the system security files on a diskette. PRSAVE can be run only by the system master security officer and only from the system console when it is being used as a display station. The saved system security files can be restored on disk by the PRESTOR procedure.

If a resource security file is defined on disk, both the password security file and the resource security file are copied to the diskette file. If a resource security file is not defined on disk, only the password security file is copied to the diskette file.

For information about the system security files and the operator classes that can be assigned, see the description of security in the *Installation and Modification Reference Manual.*

The PRSAVE procedure runs the $PRSV utility program.

Command Statement Format

PRSAVE vol-id,label

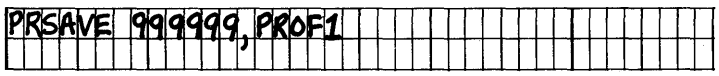
Parameters

vol-id: The volume ID of the diskette; 1 through 6 alphameric characters.

label: The label of the diskette file that will contain the security files.

Example

Save the system security file on diskette 999999. The diskette file will have the label PROF1:



PRSRC Procedure

Function

The PRSRC procedure is used by the master security officer to activate or de-activate resource (file and library) security and to define, modify, or delete the resource security file. Other security officers designated by the master security officer can use the PRSRC procedure to modify the resource security file. The PRSRC procedure can be run only form the system console when it is being used as a display station.

The *Installation and Modification Reference Manual* describes the resource security function and the user access codes that can be assigned for files and libraries that are to be protected. That appendix also describes the displays issued by the PRSRC procedure and the operator actions required.

The PRSRC procedure runs the $PRES utility program.

Command Statement Format

PRSRC

Parameters

None

Example

To run the PRSRC procedure, the security officer enters:

```
PRSRC
```

PRSRCID Procedure

Function

The PRSRCID procedure is used by the owner of a file or library to display the resource security file records for files or libraries for which the operator is classified as an owner. The operator can change the resource security file records for any file or library which he owns. The PRSRCID procedure can be run only if password security is active and if the resource security file exists.

When PRSRCID is run, the resource owner utility menu appears:

```
RESOURCE OWNER UTILITY MENU

  1. DISPLAY NAMES OF FILES/LIBRARIES

  2. CHANGE FILE/LIBRARY RECORD

  3. END OF JOB

ENTER NUMBER OF DESIRED OPTION ........
```

The operator should then enter the number of the desired option and press the Enter/Rec Adv key to continue. If the operator selects the 3 option, the PRSRCID procedure will terminate.

If the operator chooses option 1 from the resource owner utility menu, the following display appears:

```
DISPLAY NAMES OF FILES/LIBRARIES

  FILE/LIBRARY NAME ..... ALL

  PAYROLL


     CONTROL
(A - ADVANCE TO ADDITIONAL NAMES; R - RETURN TO BEGINNING OF FILE;
 C - RETURN TO MENU)
```

Not shown on the 960-character display screen.

This screen displays the labels of all of the files and libraries for which the operator is listed as an owner. Up to 36 labels are initially displayed. To display additional labels, the operator can enter the control character A and press the Enter/Rec Adv key. To return to the first list of labels, the operator can enter the control character R and press the Enter/Rec Adv key. To return to the resource owner security menu, the operator can enter the control character C and press the Enter/Rec Adv key. If the operator is not listed as an owner of any file or library, no labels are displayed.

If the operator chooses option 2 from the resource owner utility menu, the following display appears:

```
CHANGE FILE/LIBRARY RECORD

 FILE/LIBRARY NAME ..... PAYROLL
 AUDIT (Y,N) ........... Y
 ACCESS CODES AND USER IDS:                PUBLIC ACCESS (E,R,G,O).........
  (O-OWNER, G-CHANGE, R-READ ONLY, E-EXECUTE LIBRARY MEMBER)
 O,SAM        G,TOM       G,MARY            G,BILL       R,SUE       R,JANE
 R,DICK
      CONTROL      U
(A - ADVANCE TO ADDITIONAL USERS; U - UPDATE RECORD; F - GO TO NEXT RECORD;
 R - RETURN TO FIRST RECORD; I,FILE/LIBRARY NAME - ADVANCE TO THAT RECORD;
 C - RETURN TO MENU)
```

Not shown on the 960-character display screen.

This initial screen displays the first record in the resource security file for which the operator is classified as an owner. The operator can then delete, modify, or add to information on the display. Any information except the name of the file or library can be changed. If the audit option or public access is to be changed for this record, the operator should make that change on this display. If additional users remain to be displayed for this file or library, the operator can enter control character A. Control character A causes the next group of users to be displayed. The operator can then make any desired changes to the information displayed. If no more users remain to be displayed, the portion of the display that lists the users is blank. After making all of the desired changes to the record, the operator can enter control character U to update the record. The following list describes the control characters that can be used by the operator when he is using the CHANGE FILE/LIBRARY RECORD display:

Control Character	Function
A	Advance to additional users.
U	Update the record with the changes keyed by the operator. If the U control character is entered from an additional users display, the first display for the record will appear after the record had been updated.
F	Go to the next record. Displays the next resource security file record for which the operator is listed as an owner. When the end of the resource security file is reached, a message is displayed.
R	Return to the first record. The first resource security file record for which the operator is listed as an owner is displayed.
I	Advance to the identified resource security file record. A comma immediately followed by the file or library name must follow the I; for example, to display the record for the file labeled MASTER, the operator would enter I,MASTER in the control field.

For information about resources security function and the user access codes that can be assigned, see the *Installation and Modification Reference Manual.*

The PRSRCID procedure runs the $PRON utility program.

Command Statement Format

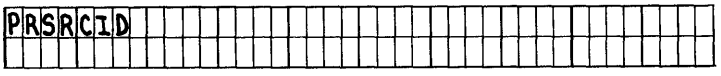
PRSRCID

Parameters

None

Example

To run the PRSRCID procedure, the owner of a file or library enters:



REBLD Procedure

The REBLD procedure is not documented in this manual. For information about this procedure, see the *WSU Reference Manual*.

RELOAD Procedure

Function

The RELOAD procedure initiates an IPL from a diskette file produced as output from the BACKUP procedure.

Space allocations for the system library (#LIBRARY) and the library directory within the library file are displayed on the display screen during execution of RELOAD. The allocations can be altered at the time they are displayed. Unaltered allocations remain as they were when the diskette file was created by the BACKUP procedure. In addition to changing the space allocations for the system library, the operator can also:

- Change the size of the task work area.
- Specify that the disk VTOC is to be rebuilt if security is not active.
- Specify that the system configuration record be rebuilt.
- Change the size of the history file.
- Change the number of possible disk VTOC entries.
- For the ideographic version of the SSP, specify the number of blocks in the extended character file. The file can be from 80 through 175 blocks long.

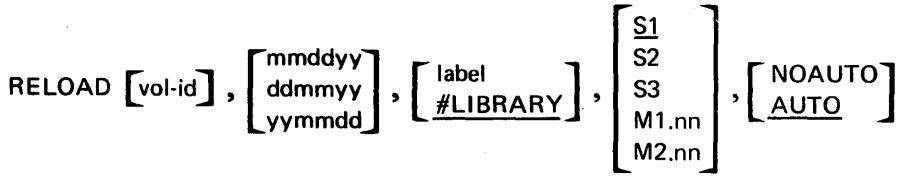
For a detailed description of the above parameters, see the *Installation and Modifications Reference Manual*. For detailed operator instructions on running the RELOAD procedure, see the *Operator's Guide*.

The RELOAD procedure runs the $LOADI utility program. The $LOADI utility program is the only means of increasing the size of the system library directory. When using $LOADI, however, be aware that library members that exist on the disk but do not exist in the backed-up library are lost when the backed-up library is returned to the disk. If library members on the disk should be saved, use the FROMLIBR procedure to copy them before executing RELOAD or $LOADI; then use TOLIBR to replace the saved members after the library is reloaded. The size of the reloaded library might have to be increased for the additional members.

Notes:

1. If you increase the size of the system files (such as the history file, the system library, or the size of the disk VTOC), be sure enough disk space exists between the system files and the first user file.
2. The ideographic version of the SSP will require more disk space for the extended character file, the task work area, and the system library than the nonideographic version of the SSP. Therefore, if the RELOAD procedure is replacing one version with the other, some files might have to be saved or moved before the RELOAD procedure is run.

Command Statement Format



Note: Parameters 4 and 5 are ignored for systems without a diskette magazine drive.

Parameters

vol-id: The volume ID of the first (or only) diskette; 1 through 6 alphameric characters.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the diskette file. The date specified must be in the same format as the date specified when the diskette file was created.

label: The label of the diskette file containing the system library. If a label is not specified, the system library (#LIBRARY) is assumed.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a fourth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, RELOAD uses only the specified slot. When the end of a diskette is reached, the SSP displays a message; the operator must insert the next diskette in the specified slot.
- If M1.nn or M2.nn is also specified, RELOAD uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine in the same magazine slot. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

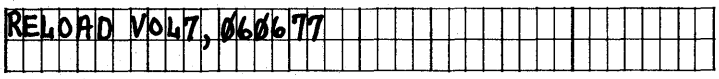
- If S1, S2, or S3 is also specified, the RELOAD procedure uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, RELOAD uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a fifth parameter is not specified, AUTO is assumed.

Example

Reload the system library backed up on June 6, 1977. The vol-id of the diskette containing the library is VOL7 and the label of the file is #LIBRARY:



REMOVE Procedure

Function

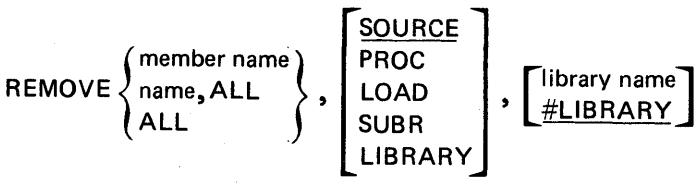
The REMOVE procedure deletes the specified library member(s), unless they are SSP members.

If single-program mode was not selected during system configuration, the space that was occupied by the deleted members can be reused only if one of the following is true:

- The program attempting to reuse the space is the only program using the library, and no active members physically follow the deleted members in the library.
- The CONDENSE procedure was used to compress the library. (The CONDENSE procedure can be used to compress a user library or the system library.)
- The library is the system library, and the BACKUP and RELOAD procedure sequence was run. (BACKUP and RELOAD compress the system library.)

If single-program mode was selected during system configuration, the space that was occupied by the deleted members can be used for new members, provided there are no active members physically located after the deleted members in the library. If active members are located after deleted members, the CONDENSE procedure can be used to make the space available for new members; or for the system library, the BACKUP and RELOAD procedure sequence can also be used.

Command Statement Format



Parameters

member name: The name of the non-SSP library member to be deleted.

name,ALL: The beginning characters of the names of non-SSP members to be deleted. Up to 7 characters can be used.

ALL: Remove all non-SSP members of the specified type or all types. ALL can be specified only if the REMOVE procedure is run from the system console and if no other jobs are running.

SOURCE: Remove non-SSP source member(s). If the second parameter is not specified, SOURCE is assumed.

PROC: Remove non-SSP procedure member(s).

LOAD: Remove non-SSP load member(s).

SUBR: Remove non-SSP subroutine member(s).

LIBRARY: Remove non-SSP members of all member types (SOURCE, PROC, LOAD, and SUBR).

library name: The name of the library containing the members to be deleted. If a library name is not specified, the system library (#LIBRARY) is assumed.

Examples

Example A

Delete the non-SSP procedure member named JOE from the library called ULIB1:

```
REMOVE JOE,PROC,ULIB1
```

Example B

Delete all non-SSP system library members that are named SAM:

```
REMOVE SAM,LIBRARY
```

Example C

Delete all non-SSP system library members beginning with the characters PAY:

```
REMOVE PAY,ALL,LIBRARY
```

RENAME Procedure

Function

The RENAME procedure changes the label of an existing disk file or library. A library can be renamed only if it was created by release 03 or a later release of System/34 and if the library is not currently active. The system library cannot be renamed.

The RENAME procedure runs the $RENAM utility program.

Command Statement Format

RENAME label1,label2, [mmddyy / ddmmyy / yymmdd]

Parameters

label1: The current label of the file or library.

label2: The new label of the file or library.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file. The date must be in the same format as the session date. If the creation date is not specified and more than one file exists with the specified label 1, the last file created is relabeled.

Example

Change the label of an existing file from OLDPAY to NEWPAY:

```
RENAME OLDPAY,NEWPAY
```

REQUESTX Procedure

Function

The REQUESTX procedure interactively registers or cancels an available user facility in an X.21 public data network. Depending upon the specific public data network, registration for a variety of facilities or services can be done by service order at subscription time, or interactively over the network itself with the REQUESTX procedure.

Two conditions must exist before REQUESTX can issue a facility registration or cancellation request:

1. At least one MLCA line must be configured as X.21 switched.

2. The X.21 switched line associated with the facility being registered or canceled must be available. The REQUESTX procedure must be run before initiating the communication component that requires the facility.

To initiate the registration or cancellation of a user facility, enter the REQUESTX procedure. The REQUESTX procedure is interactive and requires no additional parameters. The REQUESTX procedure may be entered from any display station and does not require a dedicated system.

After entering the REQUESTX procedure, a screen is displayed that prompts for the line number and the sequence associated with the facility request or cancellation:

```
                    X.21  REQUESTX  UTILITY
                                FOR
          DYNAMIC FACILITY REGISTRATION/CANCELLATION


     ENTER LINE NUMBER (1,2,3,4)
     ENTER FACILITY REGR/CANC REQUEST

     STATUS OF REQUEST -


ENTER-SEND REQUEST TO NETWORK      CMD7-EOJ      HELP-MORE INFORMATION
```

The system assumes that the operator is familiar with the specifications of the X.21 public data network to which System/34 is attached. The selection sequence must be entered in the proper format, with the proper delimiters, and the correct numeric codes for the facility being registered or canceled. Because registration and cancellation specifications differ from network to network, System/34 does not check the syntax of the selection sequence. However, System/34 will verify that the specified characters are the numeric digits 0 through 9, hyphens (-), periods (.), commas (,), or slashes (/). After the line number and the selection sequence have been specified, press the Enter/Rec Adv key to transmit the request to the network.

Command Statement Format

REQUESTX

Parameters

None

Examples

Example A

The Japanese X.21 Public Data Network, DDX, requires the customer to interactively register for and cancel the Closed User Group facility. In Japan, to register the Closed User Group facility, the following sequence is entered:

```
133/2/yy/xxxxxxx-
```

The above values and characters have the following meanings unique to the Japanese X.21 Public Data Network:

133	The Closed User Group facility request code
2	The facility registration code
yy	Two user defined numeric characters that represent the Closed User Group facility index of the remote system
xxxxxxx	The 7-digit network address of the remote system
/	Used as a separator (or delimiter) for the above values
-	Indicates the end of the selection sequence

The process can be repeated to initiate another facility request for the same X.21 line, or a different communication line can be selected. The operator can continue with additional registration or cancellation requests. The status of each request for registration, whether successful or unsuccessful, is displayed and logged to the history file. To cancel the REQUESTX procedure, use Cmd key 7.

Example B

In Japan, to cancel the Closed User Group facility, the following sequence is entered:

133/9/yy-

The above values and characters have the following meanings unique to the Japanese X.21 Public Data Network:

133	The Closed User Group facility request code
9	The facility cancellation code
yy	Two user-defined numeric characters that represent the Closed User Group facility index of the remote system
/	Used as a separator (or delimiter) for the above values
-	Indicates the end of the selection sequence

The status of each request for cancellation, whether successful or unsuccessful, is displayed and logged to the history file. To cancel the REQUESTX procedure, use Cmd key 7.

RESPONSE Procedure

Function

The RESPONSE procedure allows users to specify response options to be taken automatically when error messages are displayed at the system console. Without auto response, processing stops until the system console operator responds to the message with an option value. Auto response is particularly convenient when the system must run without an operator in attendance.

The RESPONSE procedure allows users to select messages to which the system will respond with an option and then continue processing. Only messages sent to the system console can be answered with auto response. The selected option value must be a value that is allowed for the message. Messages issued by the SSP, communications components, program products, and user-created programs can be responded to automatically.

Note: For *all situations* where a message might be encountered and its associated auto response value applied, you should carefully evaluate the effect of taking that response value in your environment. The *only* messages and auto response values that are warranted for use are listed in Examples A and B of the $ARSP utility in Chapter 4.

The RESPONSE procedure specifies a source member that contains the user-defined option values to be applied to the messages, and the name of the library that contains the source member. The RESPONSE procedure runs the $ARSP utility program, which places the auto response values from the source member in a load member, and applies them to the appropriate messages.

You must first create the source member before you run the RESPONSE procedure. See the description of the $ARSP utility program in Chapter 4 for information about creating the source member.

Once the $ARSP utility program has been run by the RESPONSE procedure, the system will respond automatically to all messages for which auto response values exist. If you do not want the system to respond automatically to a message, you must run the utility again to remove the default values.

The auto response facility cannot be run on systems with ideographic support.

Command Statement Format

RESPONSE source name, [library name / #LIBRARY]

Parameters

source name: The name of the source member that contains the auto response control statements (alpha codes) and auto response specification statements (MIC numbers).

library name: The name of the library that contains the source member. If a library name is not specified, the system library (#LIBRARY) is assumed.

Example

To apply auto response values defined in source member VALUES contained in the system library (#LIBRARY), enter:

RESTORE Procedure

Function

The RESTORE procedure restores to the disk a diskette file that was copied from the disk by one of the following:

- The ORGANIZE procedure
- The SAVE procedure
- The $COPY utility program

The RESTORE procedure can also be used to restore on the disk one or all of the entire group of files previously saved by a SAVE ALL request.

When only one file is to be restored, you can change the space allocation of the disk file by specifying the RECORDS (or BLOCKS) parameter. You can also specify the block number where the file is to begin or a preferred disk placement by specifying the LOCATION parameter and value3.

A RESTORE request reconstructs on the disk a file with the same attributes, except LOCATION, that the file had before it was copied to the diskette.

For information about the LOCATION parameter, see the descriptions of the FILE OCL statements in Chapter 1.

Messages to insert diskettes for multivolume files are automatically displayed as required, with appropriate label and volume-sequence-number checking.

The RESTORE procedure runs the $COPY utility program.

$COPY will not process a disk file that is #LIBRARY, a user library, a spool file, an offline multivolume file, or a diskette file that was not created by $COPY.

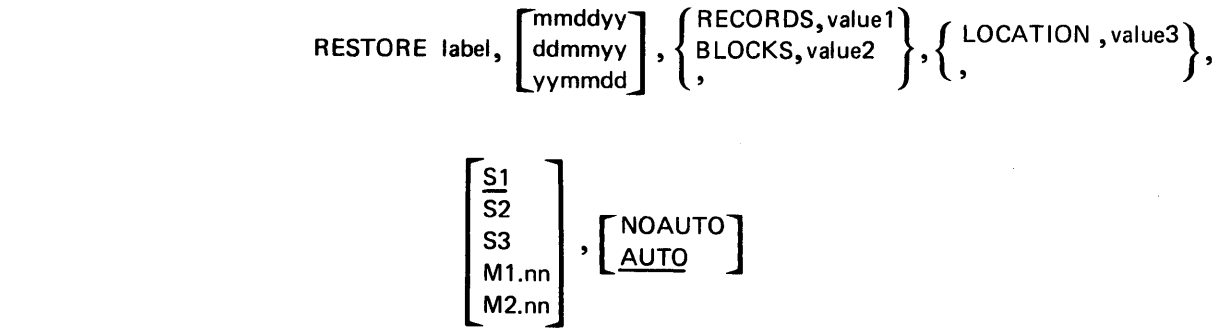
When you are copying all disk files, all members of a specified file group, or all files that are not members of a file group, the receiving diskette(s) must not contain unexpired files. When you are copying all files from diskette back to disk, the process must start with the first file on the first diskette.

Command Statement Formats

For restoring all previously saved files:

```
RESTORE [ALL] , [name   ] , [mmddyy] , [S1  ] , [NOAUTO]
                [#SAVE  ]   [ddmmyy]   [S2  ]   [AUTO  ]
                            [yymmdd]   [S3  ]
                                       [M1.nn]
                                       [M2.nn]
```

For restoring a single previously saved file:



Parameters

ALL: All files previously saved are to be restored to the disk. If the first parameter is not specified, ALL is assumed.

name: The name associated with the entire set of files previously saved on the diskette by the SAVE (SAVE ALL) procedure. If the second parameter is not specified on a RESTORE ALL statement, #SAVE is assumed.

label: The label of the single diskette file that is to be restored to the disk. If more than one file exists with the specified label and if the creation date is not specified, the system restores the first file it finds with the specified label.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the diskette file. The date specified must be in the same format as the date that was specified when the diskette file was created.

RECORDS: The disk file is to be made large enough to contain the number of records specified by value1.

value1: The number of records the disk file can contain.

BLOCKS: The disk file is to be made large enough to contain the number of blocks specified by value2.

value2: The length of the disk file, in blocks.

LOCATION: Indicates that the location (beginning block number) or a preferred disk placement will be specified by value3.

value3: value3 can be:

- The beginning block number of the new disk file. The following table gives the beginning block number location for disks 2 through 4:

Disk	Beginning Block Number Location
Disk 2	25203
Disk 3	50406
Disk 4	75609

- A1, which indicates that placement on disk 1 is preferred. If you specify disk A1, the file is allocated in the first segment (lowest address) on disk 1 that is large enough to contain the file. If not enough space is available on disk A1, the SSP attempts to allocate the file on disk A2.

- A2, which indicates that placement on logical disk A2 is preferred. Although disks 2 through 4 are physically separate, the system treats these disks as one logical disk: A2 is the logical name. If you specify a preferred placement on disk A2 (LOCATION-A2) for a multiple-disk configuration, the SSP allocates the file in the last segment (highest address) of available storage on logical disk A2 that is large enough to contain the file. For example:
  - On a three-drive system, the file is placed in the last available segment of disk 3.
  - On a four-drive system, the file is placed in the last segment of disk 4.

  If not enough space is available on the last physical disk of logical disk A2, the SSP attempts to find space on the other disks that make up A2. If not enough space is available on logical disk A2, the SSP attempts to allocate the file on disk A1.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, RESTORE uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the specified slot.

- If M1.nn or M2.nn is also specified, RESTORE uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, RESTORE uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

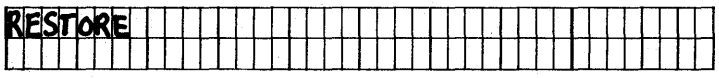
- If M1.nn or M2.nn is also specified, RESTORE uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

If a parameter is not specified, AUTO is assumed.

Examples

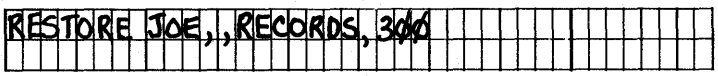
Example A

Restore all files previously saved by a SAVE procedure (a name for the set of files was not specified when SAVE was run):



Example B

Restore a file named JOE that was saved or organized on a diskette:



In this example, RECORDS specifies that the restored file must be large enough to contain 300 records.

SAVE Procedure

Function

The SAVE procedure performs one of the following functions:

- Copies a single disk file to diskette(s).
- Copies all disk files to diskette(s).
- Copies all disk files that are not members of a file group to diskette(s).
- Copies all members of a specified file group to diskette. File labels of files that belong to a file group contain a period. The characters preceding the period identify the file group, and the characters following the period identify the file within the group. As for all file labels, the maximum number of characters is 8. Examples of labels of files within a file group are:

A.B.GO A.B.INV A.INV A.ACCTS A.PROLL	Files in file group A
A1.INV A1.ACCTS A1.PROLL	Files in file group A1
A.B.GO A.B.INV	Files in file group A.B

Files with labels that do not contain a period are not part of a file group.

Sequential, indexed, and direct disk files can be copied to diskette(s) by SAVE, and can reside on diskette(s) as single-volume or multivolume files. Sequential, indexed, and direct disk files can also be added to files saved previously and can reside as single-volume or multivolume files. A message to insert a diskette is displayed to the operator whenever a SAVE request causes a multivolume diskette file to be created or extended (added to).

The SAVE procedure cannot be used to copy a file that is being used by any other job on the system. If the SAVE procedure is used to copy all disk files (ALL is specified as both the first and the fifth parameters), the procedure can be run only from the system console, and no other jobs can be running.

The SAVE procedure cannot be used to copy a library. You can use the FROMLIBR procedure to copy members from a library.

The SAVE procedure runs the $COPY utility program.

$COPY will not process a disk file that is #LIBRARY, a user library, or a spool file. Also, it will not process an offline multivolume file or a diskette file that was not created by $COPY.

When you are copying all disk files, all members of a specified file group, or all files that are not members of a file group, the receiving diskette(s) must not contain unexpired files. When you are copying all files from diskette back to disk, the process must start with the first file on the first diskette.

Command Statement Formats

For saving one disk file on diskette or for adding a disk file to a file saved previously:

```
SAVE label, [retention days] , [mmddyy] , vol-id, [S1]  , [NOAUTO]
            [1           ]   [ddmmyy]          [S2  ]   [AUTO  ]
            [ADD         ]   [yymmdd]          [S3  ]
                                               [M1.nn]
                                               [M2.nn]
```

For saving all disk files on diskette:

```
SAVE [ALL] , [retention days] , [name] , vol-id, ALL, [S1]  , [NOAUTO]
             [      1       ]   [#SAVE]                [S2  ]   [AUTO  ]
                                                       [S3  ]
                                                       [M1.nn]
                                                       [M2.nn]
```

For saving all disk files that are not members of a file group:

```
SAVE [ALL] , [retention days] , [name] , vol-id, , [S1]  , [NOAUTO]
             [      1       ]   [#SAVE]                [S2  ]   [AUTO  ]
                                                       [S3  ]
                                                       [M1.nn]
                                                       [M2.nn]
```

For saving all disk files belonging to a specified file group:

```
SAVE [ALL] , [retention days] , [name] , vol-id, filegroup, [S1]  ,
             [      1       ]   [#SAVE]                     [S2  ]
                                                            [S3  ]
                                                            [M1.nn]
                                                            [M2.nn]

     [NOAUTO]
     [AUTO  ]
```

Note: The last two parameters are ignored for systems without a diskette magazine drive.

Parameters

ALL: If ALL is specified as the first parameter, and a fifth parameter is not specified, all files that are not members of file groups are copied. (If no file groups exist, all files are saved.)

If ALL is specified as both the first and the fifth parameters, all disk files are copied, whether or not file groups exist.

If ALL is specified as the first parameter, and a file group is specified as the fifth parameter, all members of that file group are copied.

Notes:

1. If a first parameter is not specified, ALL is assumed.
2. If the first parameter is ALL or is not specified, the diskette(s) to which files are being copied cannot contain any active files.
3. Offline multivolume files on disk are ignored when ALL is the first parameter.

label: The label of one disk file to be saved. The diskette file will have the same label.

Note: An offline, multivolume file on disk cannot be saved.

retention days: The length of the retention period (0 through 999 days) for the diskette file. If the retention period is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file. For more information on diskette file retention, see *File Statement (for Diskette Files)*, RETAIN parameter, in Chapter 1.

ADD: A single disk file is to be added to a file previously saved on diskette.

name: The name associated with the entire set of saved files. If name is not specified, #SAVE is assumed.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the disk file. The date must be in the same format as the system date. If the creation date is not specified and more than one file exists with the specified label, the last file created is saved.

vol-id: The volume label of the diskette; 1 through 6 alphameric characters.

filegroup: Identifies the file group to be copied.

Note: Files on diskette cannot be processed as members of a file group.

S1, S2, or S3: Identifies the diskette to be used for output. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be be used for output. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, SAVE uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes are required; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, SAVE uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes are required, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, SAVE uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes are required, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, SAVE uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes are required, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

Each time the SAVE procedure is run, the first diskette to receive output must be in the specified location. If the diskette in that location is not capable of receiving the output, the system will not search the following slots for a valid diskette. Instead, an error message will be displayed and the operator will have to insert a diskette that is capable of receiving the output into the specified slot.

If a parameter is not specified, AUTO is assumed.

Examples

Example A

Save all files for a period of seven days on a diskette with a volume ID of 345678:

```
SAVE ALL,7,#SAVE,345678,ALL
        or
SAVE ,7,,345678,ALL
```

Example B

Save a file labeled JOE (created on November 12, 1977), and add this file to an existing diskette file labeled JOE (with a volume ID of 654321):

```
SAVE JOE,ADD,771112,654321
```

Example C

Save all files belonging to file group A1. The name associated with the set of saved files is SAVE2. The volume ID of the diskette is 345678, and the file will be saved for 39 days:

```
SAVE ALL,39,SAVE2,345678,A1
```

SAVELIBR Procedure

Function

The SAVELIBR procedure copies all members (except SSP members) from a user library to a diskette file.

The SAVELIBR procedure runs the $MAINT utility program.

Command Statement Format

```
SAVELIBR library name, [retention days] ,vol-id, [S1 ] ,
                       [  1  ]                   [S2 ]
                                                 [S3 ]
                                                 [M1.nn]
                                                 [M2.nn]

                 [NOAUTO]
                 [AUTO  ]
```

Parameters

library name: Name of the user library to copy to the diskette file. The diskette file is given the same name as the library. The library name cannot be #LIBRARY. If the library name is not specified, the user is prompted to supply the library name.

retention days: Length of the retention period for the diskette file. The retention days can be from 0 through 999 days. If the retention days are not specified, 1 day is assumed.

vol-id: Pack identification of the diskette. If the volume ID is not specified, the user is prompted to enter a volume ID.

location: Identifies either the diskette slot location (S1, S2, or S3) or the magazine location (M1.nn or M2.nn) that contains the first diskette to be used for output. If the location is not specified, slot S1 is assumed.

NOAUTO: Processing starts at a particular slot or magazine location specified on the location parameter and uses only the specified slot or continues until the end of the magazine.

AUTO: Processing starts at a particular slot or magazine location specified on the location parameter and uses all three slots or both magazines. If this parameter is not specified, AUTO is assumed.

Example

Save the contents of a library called DANLIB on a diskette with a vol-id of VOL001 for a retention period of 30 days. S1 contains the diskette to be used and only 1 diskette is required.

```
SAVELIBR DANLIB,3Ø,VOLØØ1,,NOAUTO
```

SET Procedure

Function

The SET procedure establishes the following display station environment items:

- Number of lines printed per page
- Print belt image
- Session date format
- Session date
- Default region size
- Library assigned to display station
- Printer for display station output
- Forms number

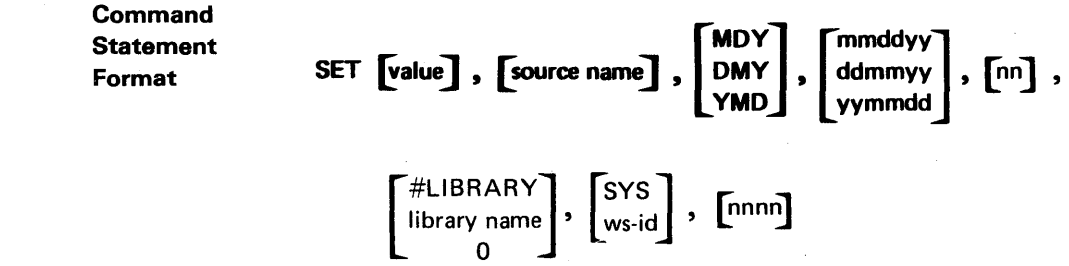
The item(s) specified is placed in the *display station configuration record*, which defines display station characteristics, and remains unchanged until a subsequent SET procedure is run from the display station, the system library is reloaded, or the system is configured again.

Notes:

1. If the SET procedure is run during an inquiry request (the operator pressed the Attn key), changes are in effect only during the inquiry request. Information displayed via the STATUS command will not reflect changes made during the inquiry request.
2. If the restart function is used to restart a checkpointed job, these values might be changed.

The SET procedure runs the $SETCF utility program.

Command Statement Format



Parameters

value: The number of lines to be printed per page. The maximum number of lines that can be specified is 112; the minimum value is 1.

For information about how the value specified determines the actual number of lines printed per page, see the description of the FORMS OCL statement in Chapter 1.

source name: The name of the source member that contains the new print belt characters for the line printer. The first record in the source member must be an IMAGE OCL statement. The new characters begin in the first position of the second record in the member.

The SSP includes the following print belt members:

Print Belt Member Name	Associated Printer
BELT48	5211 or 3262
BELT64	5211
BELT96	5211 or 3262
BELT188	5211
BELT48HN (FORTRAN)	5211 or 3262
BELT64B	3262
BELT64C	3262
BELT188B	3262

Note: The print belt member name is also referred to as the universal character set buffer (UCSB) in the *IBM 5211 Printer Models 1 and 2 Component Description and Operator's Guide,* GA24-3658, and the *IBM 3262 Printer Models A1 and B1 Component Description and Operator's Guide,* GA33-1530.

MDY: Specifies that the session date format is month-day-year.

DMY: Specifies that the session date format is day-month-year.

YMD: Specifies that the session date format is year-month-day.

mmddyy or *ddmmyy* or *yymmdd:* The session date. The date must be specified in session date format.

nn: The default region size in K (K = 1024) bytes.

library name: The name of the user library assigned to the display station. If 0 or #LIBRARY is specified, no user library is active.

Note: The specified library does not become active until the next time the operator signs on at the display station.

SYS: The system printer is used for display station output. The assignment can be overridden by a SYSLIST or PRINTER OCL statement.

Note: System list output for the display station will be printed on the previously assigned printer until the next session is initiated at the display station. You can use the SYSLIST procedure to immediately change the printer used for system list output.

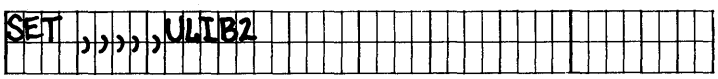
ws-id: The ID of the printer assigned for display station output. The assignment can be overridden by a PRINTER OCL statement.

Note: System list output for the display station will be printed on the previously assigned printer until the next session is initiated at the display station.

nnnn: The forms number used for display station output. The forms number can be overridden by a FORMS or PRINTER OCL statement.

Example

Change the user library for the display station to the library named ULIB2:



SETFILE Procedure

Function

The SETFILE procedure sets a disk file characteristic that enables immediate access to added records when processing sequentially by key.

This operation does not involve copying the file to a new disk location.

The SETFILE procedure runs the $SLFL utility program.

Command Statement Format

SETFILE label, [mmddyy / ddmmyy / yymmdd] , {IFILE / NIFILE}

Parameters

label: The label of an indexed file to be set.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file to be set.

IFILE: Specifies that the file is assigned the IFILE characteristic. A file with the IFILE characteristic enables index sequential users immediate access to added records. Extended disk data management and extended index data management must be configured to process a file with the IFILE characteristic.

NIFILE: Specifies that the IFILE characteristic is to be removed from the specified file.

Example

To specify the IFILE characteristic for a file named ADDREC:

```
SETFILE ADDREC,,IFILE
```

SETRETRY Procedure

Function

The SETRETRY procedure displays the configured Primary SDLC error retry count values for each communication line and allows the values to be changed. The value is the number of multiples-of-7 retries that Primary SDLC uses to attempt to contact a secondary station. If the secondary station does not respond during the specified number of retries, a permanent timeout error is reported.

Some intelligent modems such as the IBM 3865 require more time to equalize than the error retry count allows. Thus, a permanent timeout error may be reported when the modem is equalizing (not a permanent error situation). Increasing the error retry count value from the default (a value of 1, or 7 retries) prevents a permanent timeout error from occurring.

It is recommended that the default value of 1 be used unless a problem has been identified. This value only affects the operation of Primary SDLC and its associated SNA task (Remote Work Station Support, SNA Peer Support–Primary only, System/34 Finance Support, or Link Station Test).

Note: The communication line must be varied off or disabled for the change to become effective.

Command Statement Format

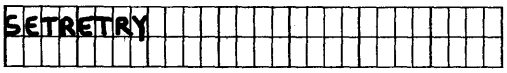
SETRETRY

Parameters

None

Example

To display and modify the Primary SDLC error retry count values, enter:



The following screen is displayed that allows you to modify the error retry count values. Remember that the value entered for a line number specifies the number of multiples of 7. For example, a value of 2 specifies 14 retries.

```
                    PRIMARY SDLC ERROR RETRY COUNTS

                        LINE        VALUE
                         1           1
                         2           1
                         3           1
                         4           1

The value is the number of multiples of 7 retries used by Primary SDLC
to contact a secondary station.  If the secondary station does not
respond within the specified number of retries, a permanent timeout
error is reported.  Valid values are 1-5.
```

SPECIFY Procedure

Function

The SPECIFY procedure is used to specify parameters relative to the SDLC communications line.

Item	Parameter
Secondary station address	ADDR
Line type	LINE
Switch type	SWTYP
Line number	LNUM
Identification data	ID

Additional communications adapter items for SDLC that can be altered are included in the ALTERSDL procedure. To identify the current values of these parameters, use the STATUS control command.

Running SPECIFY alters the values only in the communications configuration record for the requesting display station and affects only the SDLC programs that are loaded from the display station from which SPECIFY is run. SPECIFY does not affect SDLC information for remote work stations or for SSP-ICF. If the line type specified in the LINE parameter is different than the line type specified during hardware configuration, the SPECIFY procedure must be run on the system console before an SSP-ICF SNA subsystem can be enabled.

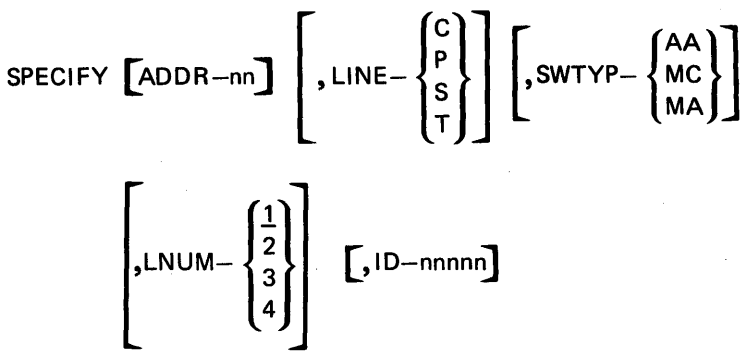
Changes made by SPECIFY remain in effect until:

- The items are changed by the $SETCF utility program or the SPECIFY procedure.
- The system library is reloaded or the system is configured again. When the system library is reloaded or the system is configured, the SDLC line items are set to the values specified during hardware configuration.

  Note: The SPECIFY procedure is intended only for data communications programming that uses SDLC. For information about SDLC, see the *IBM Synchronous Data Link Control General Information Manual.*

  The SPECIFY procedure runs the $SETCF utility program.

Command Statement Format



Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If LINE-S or LINE-C is specified, SWTYP must be specified. If SWTYP is specified, LINE-S or LINE-C must be specified. If LINE-P or LINE-T is specified, SWTYP defaults to 0.

Parameters

ADDR: The hexadecimal SDLC address of the secondary station.

LINE: Specifies the line facility:

- LINE-P specifies point-to-point nonswitched line.
- LINE-S or LINE-C specifies point-to-point switched line.
- LINE-T specifies secondary station on a multipoint line.

Note: For information on making a switched line connection for an inoperable point-to-point line to a remote work station, see the description of data communications problem determination in the *Operator's Guide*.

SWTYP: Specifies the switch type:

- SWTYP-AA specifies that if a switched line (LINE-S) is specified and the modem is in automatic-answering mode, System/34 automatically answers the call.
- SWTYP-MC specifies that if a switched line (LINE-S) is specified, the System/34 operator initiates the call manually.
- SWTYP-MA specifies that if a switched line (LINE-S) is specified, the System/34 operator answers the call manually.

If SWTYP is not specified, the value from the configuration record is used.

LNUM: Specifies the line (1, 2, 3, or 4) to which the parameters in this procedure apply. If not specified, LNUM defaults to line 1.

ID: Specifies the 5-character hexadecimal number used in an exchange of identification between the host system and the System/34 secondary SDLC station.

STARTM Procedure

Function

The STARTM procedure is used to start automatic monitoring a BSC multipoint line that has the MLCA (multiline communications adapter). The MLCA responds to all host polls or selects on a BSC multipoint line with a negative acknowledgment. A nonswappable BSC interrupt handler is not required in main storage for MLCA. Therefore, System/34 can be placed in the host's polling list without any tasks in System/34 being started until necessary.

Auto monitoring continues until stopped by the STOPM procedure, or until either a batch BSC job is started or an SSP-ICF BSC subsystem is enabled on that line.

Notes:

1. The STARTM procedure must be run from the system console and cannot be run if the line is already being monitored or is allocated to a communications component.
2. When a batch BSC job goes to EOJ or when the SSP-ICF BSC CICS or BSCEL subsystem is disabled, auto monitoring automatically goes into effect.

The STARTM procedure runs the $MMST utility program.

Command Statement Format

STARTM line number, code, station address

Parameters

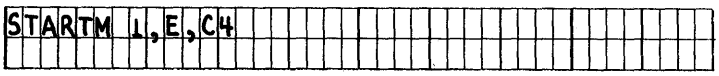
line number: The line number that is placed in auto monitor mode. Line numbers 1 through 4 are valid entries. This line must be a multipoint line.

code: Specifies transmission code. Valid entries are E (EBCDIC) and A (ASCII).

station address: The 2-character hexadecimal multipoint station address (either the poll or select sequence can be specified).

Example

This example shows how to specify line 1 to be auto monitored for an EBCDIC station address of C4:



STOPM Procedure

Function

This procedure is used to stop the auto monitoring function.

Note: The line being monitored could have been previously placed in auto monitor mode in one of three ways:

- By the STARTM procedure
- If the SSP-ICF BSC CICS or BSCEL subsystem is disabled
- If a batch BSC job terminated

The STOPM procedure runs the $MMSP utility program.

Command Statement Format

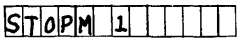
STOPM line number

Parameters

line number: Specifies that auto monitor mode is discontinued for this line. Line numbers 1 through 4 are valid entries.

Example

This example shows how to stop auto monitoring line 1:



SYSLIST Procedure

Function

The SYSLIST (system list) procedure changes the method of listing system list output. (System list output is the output generated by all SSP utility programs except for the data communications utility programs and service aids. Assembler programs can optionally generate system list output.) The SYSLIST procedure causes the system list output to be:

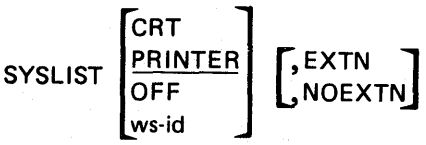
- Listed on the printer that was assigned to the display station during system configuration
- Listed on one of the other printers
- Displayed at the display station
- Not listed at all

The SYSLIST assignment remains in effect until the system processes a SYSLIST statement from the display station, until the SYSLIST procedure is run from the display station, or until the display station session ends.

Note: If the SYSLIST procedure is run during an inquiry request (the operator pressed the Attn key), the change made by the SYSLIST procedure is in effect only during the inquiry request. Information displayed via the STATUS command will not reflect a change made during the inquiry request.

The function of the SYSLIST procedure is the same as the function of the SYSLIST OCL statement.

Command Statement Format



Parameters

CRT: Display output on the display screen at the requesting display station. Up to 18 lines are displayed at a time on a 1920-character display screen. Up to 9 lines are displayed at a time on a 960-character display screen. After each system list display, the operator can request that the next set of lines be displayed or that the display be terminated. If the operator enters a blank, the number of lines shown on the next display will be the same as the number of lines shown on the previous display. If the operator enters 0, the display will be terminated. If the operator enters any characters other than a blank or 0 through 18 for a 1920-character display screen, the next 18 lines are displayed; if less than 18 lines remain, fewer lines are displayed. If the operator enters any character other than a blank or 0 through 9 for a 960-character display screen, the next 9 lines are displayed; if less than 9 lines remain, fewer lines are displayed.

Note: System output from MRT (multiple requestor terminal) programs, programs that release the requesting display station, and programs run from the input job queue is listed on the system printer.

PRINTER: Print output on the printer that was assigned to the display station during system configuration or assigned by the SET procedure or $SETCF utility program. If a parameter is not specified, PRINTER is assumed.

OFF: Do not list output.

ws-id: Print output on the printer with the specified work station ID (name). Work station IDs are assigned during system configuration and can be modified by the ASSIGN control command. To print output on the system printer, specify the work station ID of the system printer.

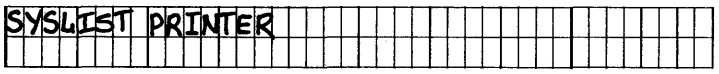
EXTN: Print or display extended characters in the system list output.

NOEXTN: Do not print or display extended characters in the system list output. The system-defined default ideographic character will be listed for any extended character.

Note: The EXTN and NOEXTN parameters are for the ideographic version of the SSP and are ignored for nonideographic systems. When a session begins, extended character processing for system list output is assumed. During the session, the EXTN and NOEXTN parameters can be used to turn on and turn off extended character processing for system list output.

Example

Print output on the printer assigned to the display station during system configuration:



TOLIBR Procedure

Function

The TOLIBR procedure copies into a library either a disk or diskette file containing one or more library members. Any number of library members can be contained in a file to be copied into a library by TOLIBR.

All sector-mode files to be copied by TOLIBR must have been created either by the $MAINT utility or the FROMLIBR procedure.

Note: If the sector-mode file was created by IBM System/32, System/32 SCP members are not copied.

Each library member in a record-mode file that is to be copied by TOLIBR must begin with a COPY statement and end with a CEND statement. For the formats of the COPY and CEND statement, see *Storing Library Members in Disk or Diskette Files* in Chapter 6.

COPY and CEND statements are automatically inserted in members created by $MAINT. *You* must insert them at the beginning and end of members that were not created by the $MAINT utility program or by the FROMLIBR procedure.

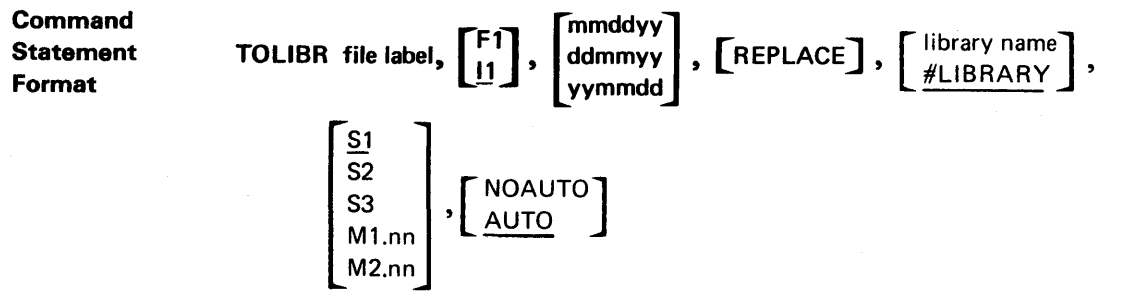
If the record-mode file is organized as a direct file, you must insert an END statement following the CEND statement that terminates the last member in the file. The format of the END statement is:

```
// END
```

where only one blank must separate the // and the END.

The TOLIBR procedure runs the $MAINT utility program.

Command Statement Format



Note: Parameters 6 and 7 are ignored for systems without a diskette magazine drive.

Parameters

file label: The label of the file containing the member(s) to be copied into the library.

F1: The file is on disk.

I1: The file is on diskette. If the second parameter is not specified, I1 is assumed.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file containing the member(s) to be copied. The date must be specified in the session date format. If more than one file exists with the specified file label, and if the date is not specified, the following applies:

- If F1 was specified, the file with the most recent creation date is copied.
- If I1 was specified or defaulted to, the first file in the diskette VTOC is copied.

REPLACE: Replace the library member specified, if one exists.

If REPLACE is not specified, members are placed in the library until a duplicate is found, at which time the system displays a message telling the operator that a duplicate exists. In response to the message, the operator can either cancel the job or continue processing. If the job is continued, the new member replaces the existing member in the library. If other duplicates are found during the job, then existing members are automatically replaced and no messages are displayed regarding the duplicate members.

If REPLACE is specified, new members replace existing duplicate members in the library, and no messages regarding the new members are displayed.

library name: The name of the library that will contain the copied member(s). If the library name is not specified, the system library (#LIBRARY) is assumed.

S1, S2, or S3: Identifies the diskette slot containing the first diskette from which members are to be copied. If a sixth parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette from which members are to be copied. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, TOLIBR uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, TOLIBR uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, TOLIBR uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, TOLIBR uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a seventh parameter is not specified, AUTO is assumed.

Example

Copy the members from a diskette file called PAY into a user library called ULIB1 and replace any duplicate members:

TRANSFER Procedure

Function

The TRANSFER procedure copies disk files to basic data exchange or I exchange diskette files or copies basic data exchange or I exchange diskette files to disk files. TRANSFER can:

- Add a diskette file that is in the basic data exchange or an I exchange format to an existing sequential disk file
- Convert a basic data exchange or an I exchange diskette file to a disk sequential or indexed file
- Convert a disk file to a basic data exchange or an I exchange diskette file (basic data exchange files are sequential files)
- Add records from a disk file to an existing basic data exchange or an I exchange diskette file

For information about the basic data exchange and the I exchange format, see Appendix C.

Notes:

1. When basic data exchange files are moved, TRANSFER cannot be used to either move files between the disk and diskette 1 diskettes (one-sided diskettes) initialized in the 512-byte format or to move files between the disk and diskette 2D diskettes (two-sided, double-density diskettes) initialized in the 1024-byte format. If the diskette format is not known, you can use the CATALOG procedure to list the diskette VTOC. This listing shows whether the diskette was initialized in the 128-, 256-, 512-, or 1024-byte format. When I exchange files are moved, TRANSFER can be used either to move files between the disk and diskette 1 diskettes initialized in the 128- or 512-byte format or to move files between the disk and diskette 2D diskettes initialized in the 256- or 1024-byte format.
2. An offline, multivolume file extent on disk cannot be converted by TRANSFER to a basic data exchange or an I exchange file.
3. Deleted records in a delete-capable file are not transferred to diskette. Also, if a delete-capable file is transferred to diskette, when it is transferred back to disk, it will no longer be delete capable.

When a basic data exchange diskette file is added to an existing sequential disk file, the record length of the disk file is used for all records added to the file. When an I exchange diskette file is added to an existing sequential disk file, the record length of the disk file must equal the record length of the diskette file. When a basic data exchange or an I exchange diskette file is converted to a sequential or an indexed disk file, records are placed in the disk file sequentially, using the record length of the diskette file.

A disk file to be converted or added by TRANSFER to a basic data exchange or an I exchange diskette (always sequential) can be a sequential, indexed, or direct file. When a basic exchange diskette file is created from a disk file and the record length of the disk file is less than or equal to the diskette sector size (128 bytes for the diskette 1 diskettes, 256 for the diskette 2D diskettes), the record length of the diskette file is set to the record length of the disk file. If the record length of the disk file is greater than the diskette sector size, all records are truncated to the sector size, which then becomes the record length.

When an I exchange diskette file is created from a disk file, the record length of the diskette file is set to the record length of the disk file, regardless of record length of the disk file.

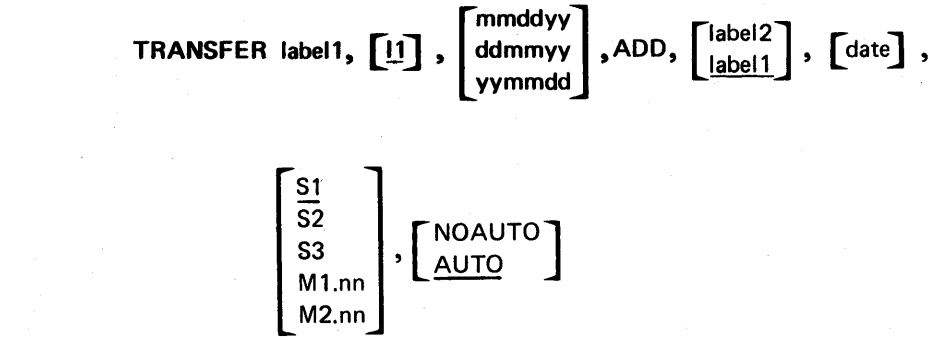
When a disk file is added to an existing basic exchange or an I exchange diskette file, the record length of the diskette file must be the same as the record length of the disk file. If the record length of the diskette file equals the diskette sector size and the disk file record length is greater than the diskette sector size, the disk file records are truncated to the sector size.

For an example of converting a source member or a procedure member to a diskette file in 128-byte basic data exchange format, see the description of the $MAINT utility program in Chapter 4.

The TRANSFER procedure runs the $BICR utility program.

Command Statement Formats

For transferring a file from diskette to an existing disk file:



For transferring a file from diskette to a new disk file:

$$\textbf{TRANSFER label1}, \left[\underline{\text{I1}}\right], \begin{bmatrix}\text{mmddyy}\\ \text{ddmmyy}\\ \text{yymmdd}\end{bmatrix}, \left[\underline{\text{NOADD}}\right], \begin{Bmatrix}\text{key length, key location}\\ ,\end{Bmatrix},$$

$$\begin{Bmatrix}\text{RECORDS, value1}\\ \text{BLOCKS, value2}\\ ,\end{Bmatrix}, \begin{bmatrix}\underline{\text{S1}}\\ \text{S2}\\ \text{S3}\\ \text{M1.nn}\\ \text{M2.nn}\end{bmatrix}, \begin{bmatrix}\text{NOAUTO}\\ \underline{\text{AUTO}}\end{bmatrix}$$

For transferring a disk file to a new diskette file or for adding a disk file to an existing diskette file:

$$\textbf{TRANSFER label1, F1}, \begin{bmatrix}\text{mmddyy}\\ \text{ddmmyy}\\ \text{yymmdd}\end{bmatrix}, \text{vol-id}, \begin{bmatrix}\text{retention days}\\ \underline{1}\end{bmatrix}, \begin{Bmatrix}\text{ADD}, \begin{bmatrix}\text{label2}\\ \text{label1}\end{bmatrix}\\ ,\end{Bmatrix},$$

$$\begin{bmatrix}\underline{\text{S1}}\\ \text{S2}\\ \text{S3}\\ \text{M1.nn}\\ \text{M2.nn}\end{bmatrix}, \begin{bmatrix}\text{NOAUTO}\\ \underline{\text{AUTO}}\end{bmatrix}, \begin{bmatrix}\text{IFORMAT}\\ \underline{\text{EXCHANGE}}\end{bmatrix}$$

Note: The eighth and ninth parameters are ignored for systems without a diskette magazine drive.

Parameters

label1: The label of the file being transferred. If a new file is being created, it is given the label specified by label1.

I1: A basic data exchange or an I exchange diskette file is being transferred to a disk sequential or indexed file. If the second parameter is not specified, I1 is assumed.

F1: A disk file is being transferred to a basic data exchange or an I exchange diskette file.

mmddyy or *ddmmyy* or *yymmdd:* The creation date of the file being transferred. If more than one file exists with the specified label 1, and if the date is not specified, the following applies:

- If the file being transferred resides on disk, the file with the most recent date is transferred to diskette.
- If the file being transferred resides on diskette, the file that is first in the diskette VTOC is transferred.

ADD: If the second parameter is I1 and the fourth parameter is ADD, the records in a diskette file are added to the records in an existing disk sequential file. The first record from the diskette file is placed after the last record in the disk file.

If the second parameter is F1 and the sixth parameter is ADD, the records in a disk file are added to an existing basic data exchange or I exchange file on diskette. The first record from the disk file is placed after the last record in the diskette file.

label2: The label of the existing disk file to which a basic data exchange or an I exchange diskette file is to be added, or the label of a basic data exchange or an I exchange diskette file to which records from a disk file are to be added. label2 is valid only if ADD is specified. If label2 is omitted, label1 is assumed.

date: The creation date of the existing disk file. date is valid only if ADD is specified. The date must be specified in the same format as the program date.

NOADD: The basic data exchange or I exchange diskette file being transferred will become a new disk file with label1 as the label. NOADD is assumed whenever a file is transferred from diskette to disk.

key length: The key length for the disk indexed file that is being created. key length can be any decimal number from 1 through 29. It must be specified with key location, and the sum of key length and key location must not exceed the record length plus 1.

key location: The relative displacement of the start position of the record keys for an indexed disk file that is being created. For basic data exchange files, key location can be any decimal number from 1 through 128 for diskette 1 or from 1 through 256 for diskette 2D. For I exchange files, key location can be any decimal number from 1 through 4096. Key location must be specified with key length, and the sum of key length and key location must not exceed the record length plus 1.

RECORDS,value1: Specifies that the disk file being created must be large enough to contain the number of records specified by value1.

Note: Either RECORDS,value1 or BLOCKS,value2 (see following) is required if (1) a multivolume file is being transferred, or (2) the created disk file is to be larger than the file being transferred.

BLOCKS,value2: Specifies that the disk file being created must be large enough to contain the number of blocks specified by value2.

Note: Either BLOCKS,value2 or RECORDS,value1 (see preceding) is required if (1) a multivolume file is being transferred, or (2) the created disk file is to be larger than the file being transferred.

vol-id: The volume ID for the diskette; 1 through 6 alphameric characters.

retention days: The length of the retention period (0 through 999 days) for the created basic data exchange or I exchange diskette file. If the retention period is not specified, 1 day is assumed. If a retention period of 999 days is specified, the diskette file is a permanent file. For more information on diskette file retention, see *File Statement (for Diskette Files)*, RETAIN parameter, in Chapter 1.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a parameter is not specified S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, TRANSFER uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the specified slot.

- If M1.nn or M2.nn is also specified, TRANSFER uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, TRANSFER uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, TRANSFER uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

Each time the TRANSFER procedure is run to copy a file to diskette, the first diskette to receive output must be in the specified location. If the diskette in that location is not capable of receiving the output, the system will not search the following slots for a valid diskette. Instead, an error message will be displayed and the operator will have to insert a diskette that is capable of receiving the output into the specified slot.

If a parameter is not specified, AUTO is assumed.

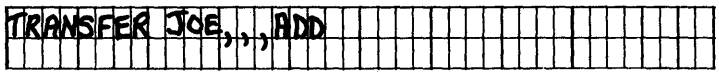
IFORMAT: A disk file is being transferred to an I exchange diskette file.

EXCHANGE: A disk file is being transferred to a basic data exchange diskette file. If a parameter is not specified, EXCHANGE is assumed.

Examples

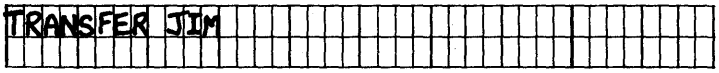
Example A

Add a basic data exchange or an I exchange diskette file labeled JOE to an existing disk file labeled JOE:



Example B

Create a disk sequential file labeled JIM from a diskette basic data exchange or I exchange file labeled JIM:



Example C

Create an I exchange diskette file labeled JON on a diskette with a volume label of 888777 from a disk file labeled JON. The file is to be saved for 30 days:

WSUTXCR Procedure

The WSUTXCR procedure is not documented in this manual. For information about this procedure, see the *WSU Reference Manual*.

WSUTXEX Procedure

The WSUTXEX procedure is not documented in this manual. For information about this procedure, see the *WSU Reference Manual*.

WSUTXRV Procedure

The WSUTXRV procedure is not documented in this manual. For information about this procedure, see the *WSU Reference Manual*.

XREST Procedure (For the Ideographic Version of the SSP)

Function

The XREST procedure restores from diskette the extended character file. The restored file is placed in a disk file labeled #EXTN. The diskette contains the IBM-supplied extended characters and any user-generated extended characters.

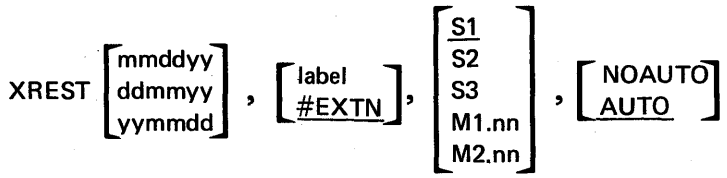
The XREST procedure can be run only on files saved by the XSAVE procedure.

The XREST procedure can be run only from the system console and cannot be run while any other jobs are being run.

You can use the Character Generator Utility to enter ideographic characters in the extended character file. For information about how to use the Character Generator Utility, see the *IBM System/34 Character Generator Utility User's Guide and Reference Manual.*

The XREST procedure runs the $XREST utility program.

Command Statement Format



Parameters

mmddyy or ddmmyy or yymmdd: The creation date of the diskette file. The date specified must be in the same format as the date that was specified when the diskette file was created.

label: The label of the diskette file that contains the extended character file. If a label is not specified, #EXTN is assumed.

S1, S2, or S3: Identifies the diskette slot containing the first diskette to be processed. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be processed. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, XREST uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes remain to be processed; the operator must insert the next diskette into the specified slot.

- If M1.nn or M2.nn is also specified, XREST uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes remain to be processed, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, XREST uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, XREST uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes remain to be processed, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

  If a parameter is not specified, AUTO is assumed.

Example

Restore the extended character file from diskette. The label of the diskette file is XCHAR2:

```
XREST ,XCHAR2
```

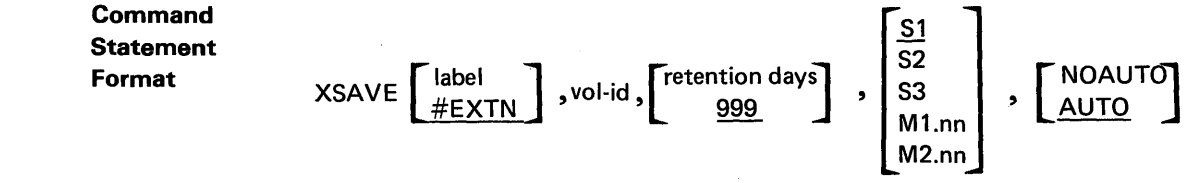
XSAVE Procedure (For the Ideographic Version of the SSP)

Function

The XSAVE procedure saves the contents of the extended character file on diskette. For information about how the extended character file is generated, see the RELOAD procedure.

The XSAVE procedure runs the $XSAVE utility program.

Command Statement Format



Parameters

label: The label to be assigned to the diskette file. If a label is not specified, #EXTN is assumed.

vol-id: The volume label of the diskette; 1 through 6 alphameric characters.

retention days: The length of the retention period (0 through 999 days) for the diskette file. If a retention period is not specified, 999 is assumed and the file is treated as a permanent file. For more information on diskette file retention, see the description of the RETAIN parameter under *File Statement (for Diskette Files)* in Chapter 1.

S1, S2, or S3: Identifies the diskette slot containing the diskette to be used for output. If a parameter is not specified, S1 is assumed.

M1.nn or M2.nn: Identifies the magazine location containing the first diskette to be used for output. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 through 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

NOAUTO: Specifies the following:

- If S1, S2, or S3 is also specified, XSAVE uses only the specified slot. When the end of a diskette is reached, the SSP displays a message if more diskettes are required; the operator must insert the next diskette into the slot being used.

- If M1.nn or M2.nn is also specified, XSAVE uses only the magazine slot that contains the specified location. Processing begins with the diskette at the specified location and continues through the last diskette in the magazine. If more diskettes are required, the SSP displays a message; the operator must insert the next magazine into the magazine slot being used. Processing resumes at the first location in the magazine.

AUTO: Specifies the following:

- If S1, S2, or S3 is also specified, XSAVE uses all three individual slots. Processing begins with the diskette in the specified slot and continues through the diskette in slot S3. If more diskettes are required, the SSP displays a message; the operator must then insert the next diskettes. Processing resumes with the diskette in slot S1.

- If M1.nn or M2.nn is also specified, XSAVE uses both magazine slots. Processing begins with the diskette at the specified location and continues through the diskette in location M2.10. If more diskettes are required, the SSP displays a message; the operator must then insert the next magazines. Processing resumes at location M1.01 and continues through M2.10.

Each time the XSAVE procedure is run, the first diskette to receive output must be in the specified location. If the diskette in that location is not capable of receiving the output, the system does not search the following slots for a valid diskette. Instead, an error message is displayed and the operator has to insert a diskette that is capable of receiving the output into the specified slot.

If a parameter is not specified, AUTO is assumed.

Example

Save the extended character file on a diskette labeled BACK02. The label of the diskette file should be XCHAR2, and the file should be a permanent file:

```
XSAVE XCHAR2,BACK02
```

Chapter 3. Control Commands

This chapter describes the System/34 control commands, which are supplied as part of the SSP. The control commands are divided into three groups:

- Display station operator control commands that can be entered from any display station, including the system console when it is being used as a display station. Display station operator control commands can be entered from the keyboard or selected from a programmer-defined menu. They cannot be coded as part of a procedure. The display station operator control commands are:

CANCEL	JOBQ	PRTY
CHANGE	MENU	RELEASE
CONSOLE	MODE	STATUS
HOLD	MSGTIME	
IDELETE	OFF	

- Subconsole operator commands that can be entered from any display station that has been designated as a subconsole when it is in subconsole mode (display stations are designated as subconsoles at system configuration time). The subconsole commands are:

CANCEL	RELEASE	STATUS
CHANGE	REPLY	STOP
HOLD	RESTART	TIME
MSG	START	

- System operator control commands that can be entered only from the system console by the system operator. They can be entered only from the keyboard. The system operator control commands are:

ASSIGN	MSG	START
CANCEL	PRTY	STATUS
CHANGE	RELEASE	STOP
HOLD	REPLY	TIME
IDELETE	RESTART	VARY

Note that the same command (for example, CANCEL) can often be used as a display station operator control command, as a subconsole operator command, and as a system operator control command. In most cases, the function and the format of a command depends upon which of the three uses it has.

Note: The SETDUMP service command, which controls the address compare dump function of System/34, is described in *Appendix D. SSP Service Procedures and Commands.*

The description of each control command contains:

- The function, or functions, of the command
- The format of the command (for a description of the conventions used to describe formats, see *Conventions Used for Describing Statement Formats* at the front of this manual)
- Descriptions of the command parameters
- An example, or examples, of how the command is used

For complete System/34 operating information, see the *Operator's Guide*.

Display Station Operator Control Commands

CANCEL Command

Function

When used as a display station operator control command, the CANCEL command cancels a job on the input job queue or an entry on the spool file.

Format

```
CANCEL  { JOBQ,jobname }
(C)     { (J)         }
        { PRT,spool-id }
        { (P)         }
```

Parameters

JOBQ: Cancels a job on the input job queue.

jobname: The 8-character, system-assigned job name of the job to be canceled. The display station operator can use the STATUS JOBQ control command to display the job names of the jobs on the input job queue.

The job to be canceled must have been placed on the input job queue by the operator using the JOBQ command or the OCL statement, and the operator must enter the CANCEL command from the same display station that was used when the job was placed on the queue.

PRT: Cancels an entry on the spool file.

spool-id: The 6-character, system-assigned spool file entry ID of the entry to be canceled. The display station operator can use the STATUS PRT control command to display the names of the entries on the spool file. The display station operator may cancel only the spool file entries that he placed on the spool file.

Examples

Example A

To cancel job W1001111 on the input job queue, the display station operator enters:

```
CANCEL JOBQ,W1001111
```

or

```
C J,W1001111
```

Example B

To cancel entry SP0141 on the spool file, the display station operator enters:

```
CANCEL PRT,SP0141
```

or

```
C P,SP0141
```

CHANGE Command

Function

The CHANGE command changes the following items:

- The number of copies to be printed for an entry on the spool file
- The forms number to be used for an entry on the spool file
- The defer status of a spool file entry that is currently being intercepted

Note: The display station operator may change only entries that he places on the spool file.

Format

$$\text{CHANGE (G)} \left\{ \begin{array}{l} \text{COPIES, nn, spool-id} \\ \text{FORMS, xxxx, spool-id} \\ \text{DEFER, } \left[\begin{array}{l} \text{YES} \\ \underline{\text{NO}} \end{array} \right] \text{, spool-id} \end{array} \right\}$$

Parameters

COPIES: Changes the number of copies of printed output for an entry on the spool file.

FORMS: Changes the forms number to be used for an entry on the spool file.

DEFER: Changes the defer status of a spool file entry that is currently being intercepted.

nn: The number of copies to be printed for the entry. nn can be any decimal number from 1 through 99.

xxxx: The forms number of the forms to be used for the specified entry; 1 through 4 characters.

YES: The system does not begin printing spooled output from the specified spool file entry until the entry is completely intercepted.

If the spool file entry is being intercepted and printed when the CHANGE command is entered, printing of the entry stops and another spool file entry, if one exists, begins printing. Once the spool file entry has been completely intercepted, the entry can then be printed.

NO: The spooled output from the specified spool file entry can begin printing before the job is completely intercepted. If a parameter is not specified, NO is assumed.

spool-id: The 6-character spool file entry ID of an entry on the spool file. The STATUS PRT control command can be used to determine the spool file entry ID.

Example

To change entry SP0011 on the spool file so that five copies of its output are printed, the display station operator enters:

```
CHANGE COPIES,5,SP0011
```

or

```
G COPIES,5,SP0011
```

CONSOLE Command

Function

The CONSOLE command causes the display station from which the command was entered to become the system console. The command is allowed only if all the following conditions are true:

- The command is entered from a display station that was assigned as an alternative system console during system configuration.
- The CONSOLE command is entered while the display station is not in inquiry mode.
- Initial program load (IPL) has been performed or the current system console is not functional.

If password security is active, then the command must be entered by an operator who is assigned the security status of system operator or higher.

For information about assigning security status, see the description of the $PROF utility program in Chapter 4.

Notes:

1. If the CONSOLE command is entered after IPL has completed, the alternative system console from which the command is entered must have the same or larger screen (1920-character) as the current system console. If the CONSOLE command is entered before IPL has completed, the alternative system console can be assigned to a smaller (960-character) screen.
2. If an alternative console is made the system console via the CONSOLE command and then made an alternative console again, it will have the NOSUB attribute. For more information on the NOSUB attribute, see the ASSIGN command later in this chapter. For example, display station W1 is the system console and W2 is an alternative console functioning as a subconsole. If W1 fails, W2 may be made the system console via the CONSOLE command. When W1 becomes the system console again, W2 becomes an alternate console with the NOSUB attribute. If the system operator wants to reactivate the subconsole function at W2, he should issue the ASSIGN SUB command.

Format

CONSOLE

Parameters

None

Example

An I/O error occurs at the system console. To assign display station W3 (an alternative system console) as the system console, the system operator signs on at display station W3 and enters:

```
CONSOLE
```

HOLD Command

Function

The HOLD command prevents a specified spool file entry from being printed. The HOLD command has the same effect as specifying PRIORITY-0 on the PRINTER OCL statement.

If a HOLD command specifies an entry that is being printed, the printing of the entry is interrupted and the printing of the next entry on the spool file begins. The held entry is eligible for printing after a RELEASE control command releases that entry.

Format

HOLD PRT,spool-id
(H) (P)

Parameters

PRT: Indicates that an entry on the spool file should be held.

spool-id: The 6-character spool file entry to be held. The STATUS PRT control command can be used to determine the spool file entry ID.

The display station operator may hold only entries that he placed on the spool file.

Example

To hold entry SP0036 on the spool file, the display station operator enters:

HOLD PRT,SP0036

or

H P,SP0036

IDELETE Command

Function

The IDELETE command specifies whether informational messages should be suppressed.

Format

IDELETE [ON / OFF]

Parameters

ON: The SSP displays informational messages only if an INPUT/OUTPUT display is active. Otherwise, informational messages are suppressed. If a parameter is not specified, ON is assumed.

OFF: The SSP displays informational messages.

Example

To suppress displaying of informational messages (for example, messages displayed via a // * OCL statement), the display station operator enters:

```
IDELETE
```

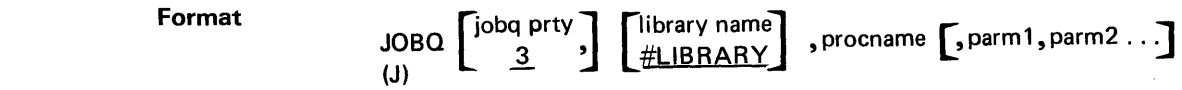
JOBQ Command

Function

The JOBQ command places a job on the input job queue.

Note: For the ideographic version of the SSP, jobs run from the input job queue print alphameric system headings.

Format



Parameters

jobq prty: Specifies job queue priority; that is, the job's order of execution from the input job queue. The job queue priority can be any decimal number from 1 through 5. The system runs jobs in order of decreasing job queue priority. For example, all jobs with a job queue priority of 5 are run before any other jobs in the job queue. Jobs with the same job queue priority are run in the order they were placed in the job queue. Jobs with a job queue priority of 1 are the last jobs run by the system. If the jobq prty parameter is not specified, a job queue priority of 3 is assigned to the job, and the comma shown in this parameter should not be specified.

library name: Specifies the active user library for the job. The system searches the active user library and then the system library for the procedures and load members executed in this job and for the message members and display screen formats used in this job. If a library name is not specified, the system library (#LIBRARY) is assumed.

procname: The name of the procedure (a procedure member) that defines the job to be placed on the input job queue.

parm1,parm2,. . . : Parameters required by the procedure. From 1 through 11 parameters can be specified.

Example

To place the PAYROLL procedure, which is located in library PAYLIB, on the input job queue with a job queue priority of 5, the display station operator enters:

```
JOBQ 5,PAYLIB,PAYROLL
```

or

```
J 5,PAYLIB,PAYROLL
```

MENU Command

Function

The MENU command activates the menu function and displays a specified menu on the display screen at the display station.

For a sample menu display and for information about creating menus, see the description of the BLDMENU procedure in Chapter 2.

Notes:

1. The display station operator can also activate a menu by specifying the menu name during the sign-on procedure.
2. The operator terminates the menu by selecting option 0.
3. For the ideographic version of the SSP, an error message is displayed if an attempt is made to display a menu that contains ideographic characters at a display station that cannot display ideographic characters.

Format

MENU menuname [,library]

Parameter

menuname: The name of the menu to be displayed; 1 through 6 alphameric characters.

library: The name of the library containing the MENU. If a library name is not specified, the system searches the active user library, if one exists, and then searches the system library (#LIBRARY) for the menu.

Note: Locating and displaying a menu *does not* activate that library. Use the LIBRARY OCL statement to activate the library before attempting to execute jobs from the displayed menu.

Example

To display menu M12, which is in the active user library, the display station operator enters:

```
MENU M12
```

MODE Command

Function

The MODE command puts the display station into a mode of operation (standby mode) in which it waits to be acquired and used by a currently executing program. If the MODE command is entered when the display station is already waiting to be acquired by a program, the MODE command returns the display station to its normal operating mode. While a display station is waiting to be acquired by a program, only the MODE, MSG, and OFF control commands can be entered.

Format

MODE

Parameters

None

Example

A display station is in normal operating mode. To change the mode of display station operation so that the display station can be acquired by a program, the display station operator enters:

```
MODE
```

MSG Command

Function

When used as a display station operator control command, the MSG command:

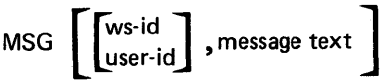
- Sends a message to the system console or to a selected display station or display station operator
- Displays messages that are queued to the display station when the command is entered with no parameters

  Note: Messages received by a display station are placed on the message queue for that display station. If a session is active at the display station, the audible alarm sounds and the message light comes on each time a message is placed on the queue. Up to 15 messages can be on a display station message queue. If messages are received when the queue is full, the messages are not placed on the queue and the sender is notified that the message queue is full. Messages on the queue are displayed when (1) an operator signs on the display station, (2) the operator enters the MSG control command with no parameters, or (3) a job being run from the display station terminates. When a message is displayed, it is removed from the queue.

- Displays a dual-routed system console message

  Note: The MSG command displays only the last rerouted message.

Format



Notes:

1. If no parameters are specified, the messages queued to the display station are displayed.
2. If the first parameter is not entered but the second one is, the message is sent to the system console.

Parameters

ws-id: The 2-character work station ID of the display station to which the message is sent. The STATUS WORKSTN control command can be used to determine the work station IDs. If the work station ID of the system console is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console. If the first parameter is not entered and a message text is entered, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console. The message is placed in a message queue for the system console instead of in the queue for the display station assigned as the system console.

user-id: The 1- to 8-character, user ID that identifies the operator to whom the message is sent. Each display station operator enters a user ID on the SIGN ON display when the session is started. The STATUS WORKSTN control command can be used to determine user IDs. If the user ID of the system operator is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console. If the first parameter is not entered and a message text is entered, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console.

message text: Up to 60 alphameric or special characters.

Note: For the ideographic version of the SSP, the message text can contain ideographic characters; however, the SSP will not send the message to a nonideographic display station. If an attempt is made to send an ideographic message to a nonideographic display station, the SSP issues an error message.

Example

A display station operator is preparing to submit a job that requires diskette BFILE. To inform the system operator that diskette BFILE is required, the display station operator enters:

```
MSG ,NEXT JOB REQUIRES DISKETTE BFILE
```

OFF Command

Function

The OFF command terminates a display station session. After the session ends, the SIGN ON display appears on the display screen at the display station. The OFF command is not processed unless all jobs using the display screen have ended. Messages on the message queue for the display station are removed from the queue at this time.

Format

OFF [DROP / HOLD]

Parameters

DROP: Drops the communication line connection for a remote work station on a switched line. The SIGN ON display will not appear. A message indicating that the sign off was successful will be displayed. If a parameter is not entered, DROP is assumed.

HOLD: Holds the communication line connection and displays the SIGN ON display for a remote work station on a switched line.

Example

To terminate a display station session, the display station operator enters:

```
OFF
```

PRTY Command

Function

When used as a display station operator control command, the PRTY command assigns execution priority to the next job submitted from the display station. If the next job submitted is placed on the input job queue, the job will execute with the specified priority once it reaches the top of the job queue, but the job's position on the queue is not affected. To specify the position of a job on the job queue, refer to the JOBQ statement in Chapter 1 or to the JOBQ command earlier in this chapter. If the next job submitted is not placed on the input job queue, the job receives the specified execution priority.

Format

```
PRTY [ ON     ]
     [ HIGH   ]
     [ MEDIUM ]
     [ OFF    ]
     [ NORMAL ]
     [ LOW    ]
```

Parameters

ON or *HIGH:* System resources are assigned to the job before they are assigned to a lower priority (MEDIUM, NORMAL, OFF, or LOW) job. If a parameter is not specified, ON is assumed.

MEDIUM: System resources are assigned to the job before they are assigned to a lower priority (NORMAL, OFF, or LOW) job.

OFF or *NORMAL:* System resources are assigned to the job before they are assigned to a LOW priority job.

LOW: System resources are assigned to the job after all other higher priority jobs have been assigned system resources.

Example

To run the next job submitted as a medium priority job, the display station operator enters:

```
PRTY ,MEDIUM
```

RELEASE Command

Function

The RELEASE command releases for printing a specified job on the spool file. In order to be printed, jobs that were held on the spool file either by a HOLD command or by a PRIORITY-0 specification on the PRINTER OCL statement must be released.

Format

```
RELEASE PRT,spool-id
(L)     (P)
```

Parameters

PRT: Indicates that a specified job on the spool file should be released.

spool-id: The 6-character spool file job ID of the job to be released. The STATUS PRT control command can be used to determine the spool file job ID. The display station operator may release only spool file entries that he placed on the spool file.

Example

To release job SP0013 on the spool file, the display station operator enters:

```
RELEASE PRT,SP0013
```

or

```
L P,SP0013
```

STATUS Command

Function

When used as a display station operator control command, the STATUS command displays any of the following:

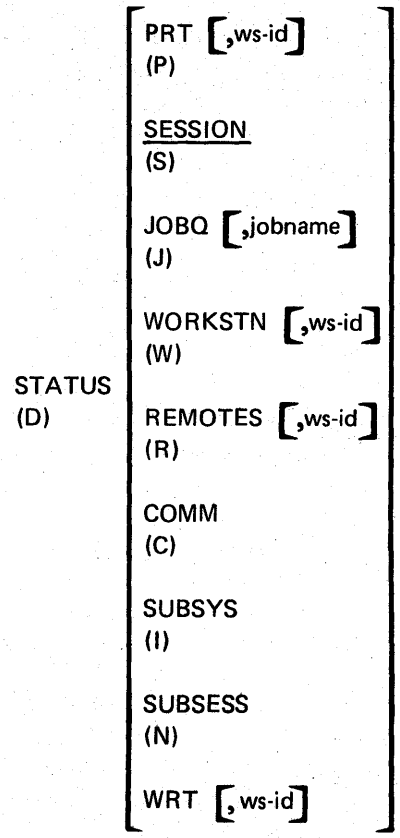
- Spool file entries that the display station operator has placed on the spool file (entries can be displayed for all printers or for a specified printer)
- Status information about the display station session
- Input job queue entries for jobs submitted from the requesting display station
- Status information about the local display stations and printers, the remote display stations and printers that are not offline, and the diskette drive
- Status information about the communication lines
- Status information about enabled SSP-ICF subsystems
- Status information about active SSP-ICF sessions
- Status information about the spool writers

Each time status information is displayed on the display screen, the operator can:

- End the status display
- Display the next page of status information (if the last page is being displayed, the status display resets to the first page of status information)
- Enter a control command
- Reset the status display to the first page of status information
- Request an updated status display

For a detailed description of the information appearing on the status displays, see the *Operator's Guide*.

Format


STATUS
(D)
PRT [,ws-id]
(P)
SESSION
(S)
JOBQ [,jobname]
(J)
WORKSTN [,ws-id]
(W)
REMOTES [,ws-id]
(R)
COMM
(C)
SUBSYS
(I)
SUBSESS
(N)
WRT [,ws-id]

Parameters

PRT: Displays information about the entries that the display station operator has placed on the spool file or about the entries that the display station operator has placed on the spool file for a specified printer.

SESSION: Displays information about the display station session. If a parameter is not entered, SESSION is assumed.

JOBQ: Displays information about jobs on the input job queue. Only information about jobs from the requesting display station is displayed.

WORKSTN: Displays status information about:

- The local display stations and printers
- The remote display stations and printers that are not offline
- The diskette drive

If a ws-id parameter is not specified, the status of all of the devices that satisfy the preceding criteria is displayed. If a ws-id parameter is specified, the status of only the specified device is displayed.

REMOTES: Displays status information about the remote display stations and the remote printers. If a ws-id parameter is not specified, the status of all remote display stations and remote printers is displayed. If a ws-id parameter is specified, the status of only the specified device is displayed.

COMM: Displays status information about the communication lines.

SUBSYS: Displays status information about enabled SSP-Interactive Communications Feature subsystems.

SUBSESS: Displays status information about active SSP-ICF sessions.

WRT: Displays status information about the spool writers or about the spool writer for a specified printer.

jobname: The 8-character, system-assigned job name. If jobname is specified, only status information for that job is displayed.

ws-id: The work station ID of the device for which status information is to be displayed.

Example

To display status information about a display station session, the display station operator enters:

```
STATUS
```

TIME Command

Function

The TIME command displays the time of day and the system date. The time is based upon the time specified by the system operator during IPL and is displayed in the following format:

hours:minutes:seconds

The system date is displayed in the system date format:

mmddyy, ddmmyy, or yymmdd

Format

TIME

Parameters

None

Example

To display the time of day and the system date, the display station operator enters:

TIME

Subconsole Operator Control Commands

CANCEL Command

Function

For printers controlled by this subconsole, the CANCEL command cancels the following entries from the spool file:

- A specified entry
- All entries
- All entries for a specified printer

The CANCEL command also cancels a specified job on the input job queue.

Format

```
                 ⎧ JOBQ,jobname      ⎫
                 ⎪ (J)               ⎪
CANCEL           ⎨                   ⎬
(C)              ⎪ PRT ⎧,spool-id⎫   ⎪
                 ⎩ (P) ⎨,ALL     ⎬   ⎭
                 ⎩     ⎩,ws-id   ⎭   ⎭
```

Parameters

JOBQ: Cancels a job on the input job queue.

jobname: The 8-character, system-assigned job name of the job to be canceled. The subconsole operator can use the STATUS JOBQ control command to display the names of jobs on the input job queue.

PRT: Cancels an entry, all entries on the spool file for the specified printer, or all entries on the spool file for all printers controlled by the subconsole.

spool-id: The 6-character ID of the spool file entry to be canceled. The STATUS PRT control command can be used to determine the spool file entry IDs.

ALL: Cancels all entries on the spool file for all printers controlled by the subconsole.

ws-id: Cancels all entries on the spool file destined for the printer with the specified ws-id.

Example

To cancel spool file entry SP0001, the subconsole operator enters:

```
CANCEL PRT,SP0001
```

or

```
C P,SP0001
```

CHANGE Command

Function

The CHANGE command changes the following items:

- The printer used by an entry or entries on the spool file
- The number of copies to be printed for an entry on the spool file
- The forms number to be used for an entry on the spool file
- The position of an entry on the spool file
- The defer status of a spool file entry that is currently being intercepted
- The priority of the spool writer for a specified printer
- The resident/swappable attribute of the spool writer for a specified printer
- The number of separator pages printed by the spool writer before printing each spool file entry

Format

```
                 ⎧ ID, printer-id-1 {, spool-id   }                  ⎫
                 ⎪                  {, printer-id-2}                  ⎪
                 ⎪ COPIES, nn, spool-id                               ⎪
                 ⎪ FORMS, xxxx, spool-id                              ⎪
                 ⎪ PRT, spool-id [, spool-id-1]                       ⎪
                 ⎪ (P)                                                ⎪
                 ⎪ DEFER, [YES] , spool-id                            ⎪
CHANGE           ⎪                [NO]                                ⎪
(G)              ⎪ PRTY, [HIGH  ] [, printer-id]                      ⎪
                 ⎪       [NORMAL]                                     ⎪
                 ⎪ RES, [YES] [, printer-id]                          ⎪
                 ⎪        [NO]                                        ⎪
                 ⎪ SEP, [0]                                           ⎪
                 ⎪      [1]                                           ⎪
                 ⎪      [2]  [, printer-id]                           ⎪
                 ⎪      [3]                                           ⎪
```

Parameters

ID: Changes the printer used for an entry or entries on the spool file.

COPIES: Changes the number of copies of printed output for an entry on the spool file.

FORMS: Changes the forms number to be used for an entry on the spool file.

PRT: Changes the position of an entry on the spool file.

DEFER: Changes the defer status of a spool file entry that is currently being intercepted.

PRTY: Changes the priority of the spool writer.

RES: Changes the resident/swappable attribute for the spool writer.

SEP: Changes the number of separator pages the spool writer prints before printing each spool file entry.

printer-id: The printer ID of the printer. If a printer ID is not specified, the system printer is assumed.

- If PRTY is specified in the first parameter position, the priority of the spool writer for the printer identified by printer-id is changed. The new spool writer priority remains in effect until changed by another CHANGE command or until IPL is performed.

- If RES is specified in the first parameter position, the resident/swappable attribute of the spool writer for the printer identified by printer-id is changed. You must first use the STOP command to cause the spool writer to stop printing. The new resident/swappable attribute remains in effect until changed by another CHANGE command or until IPL is performed.

- If SEP is specified in the first parameter position, the number of separator pages printed by the spool writer for the printer identified by printer-id is changed. The new number of separator pages remains in effect until changed by another CHANGE command or until IPL is performed.

printer-id-1: The printer ID of the new printer to be used. If a spool file entry ID is specified in the third parameter position, that entry will use the new printer (printer-id-1). If a printer ID is specified in the third parameter position (printer-id-2), all spool file entries currently using the printer specified by printer-id-2 will use the new printer (printer-id-1).

printer-id-2: The printer ID used by spool file entries for which the printer is to be changed. The subconsole operator must control this printer.

nn: The number of copies to be printed for the entry. nn can be any decimal number from 1 through 99.

xxxx: The forms number of the forms to be used for the specified entry; 1 through 4 characters.

spool-id: The 6-character spool file entry ID of an entry on the spool file. The STATUS PRT control command can be used to determine the spool file entry ID. The subconsole operator must control the printer on which the entry is destined to be printed.

spool-id-1: The 6-character spool file entry ID of an entry on the spool file. The entry being changed (spool-id) is placed on the spool file following the entry with the ID specified by spool-id-1. If spool-id-1 is not specified, the entry being changed is moved to the front of the spool file with a priority value of 5. For more information about spool file entry priorities, see the *Concepts and Design Guide*.

YES: If DEFER is specified in the first parameter position, the system does not begin printing spooled output from the specified spool file entry until the entry is completely intercepted. If the spool file entry is being intercepted and printed when the CHANGE command is entered, printing of the entry stops and another spool file entry, if one exists, begins printing. Once the spool file entry has been completely intercepted, the entry can then be printed.

If RES is specified in the first parameter position, the spool writer for the printer identified by printer-id is to be resident. You must first use the STOP command to cause the spool writer to stop printing.

NO: If DEFER is specified in the first parameter position, the spooled output from the specified spool file entry can begin printing before the entry is completely intercepted. If a parameter is not specified, NO is assumed.

If RES is specified in the first parameter position, the spool writer for the printer identified by printer-id is not to be resident. You must first use the STOP command to cause the spool writer to stop printing. If a parameter is not specified, NO is assumed.

HIGH: The spool writer for the printer identified by printer-id is to have a high priority.

NORMAL: The spool writer for the printer identified by printer-id is to have a normal priority. If a parameter is not specified, NORMAL is assumed.

0,1,2,3: The number of separator pages the spool writer for the printer identified by printer-id prints before each spool file entry. If a parameter is not specified, 0 (zero) separator pages is assumed.

Examples

Example A

To change entry SP0011 on the spool file so that five copies of its output are printed, the subconsole operator enters:

```
CHANGE COPIES,5,SP0011
```

or

```
G COPIES,5,SP0011
```

Example B

To change the printer used by an entry or entries on the spool file from P3 to P2, the subconsole operator enters:

```
CHANGE ID,P2,P3
```

or

```
G ID,P2,P3
```

Example C

To change entry SP0015 on the spool file so that it follows entry SP0008, the subconsole operator enters:

```
CHANGE PRT,SP0015,SP0008
```

or

```
G P,SP0015,SP0008
```

HOLD Command

Function

The HOLD command prevents a specified entry on the spool file, or all entries for a specified printer on the spool file, from being printed. The HOLD command has the same effect as specifying PRIORITY-0 on the PRINTER OCL statement.

If a HOLD command specifies an entry that is being printed, the printing of that entry is interrupted and the printing of the next entry on the spool file begins. The held entry is eligible to be printed when a RELEASE control command releases that entry.

The HOLD command holds only those entries currently on the spool file. The HOLD command does not hold those entries added to the spool file after the command is entered.

Format

```
HOLD PRT  { ,spool-id }
(H)  (P)  { ,ws-id    }
```

Parameters

PRT: The PRT parameter, used alone, can be used by only the system operator. The subconsole operator must enter PRT with a second parameter specified. PRT used with spool-id indicates that the specified entry on the spool file should be held. PRT used with ws-id indicates that all entries for a specified printer should be held.

spool-id: The 6-character spool file entry ID of the entry to be held. The STATUS PRT control command can be used to determine the spool file entry ID. The subconsole operator must control the printer on which the entry is destined to be printed.

ws-id: The ID of the printer for which spool file entries will be held from printing. The subconsole operator must control the specified printer.

Example

To hold entry SP0036 on the spool file, the subconsole operator enters:

```
HOLD PRT,SP0036
```

or

```
H P,SP0036
```

MSG Command

Function

When used as a subconsole control command, the MSG command sends a message to the system console or to a selected display station or display station operator.

Format

MSG [ws-id / user-id] ,message text

Note: If the first parameter is not entered but the second one is, the message is sent to the system console.

Parameters

ws-id: The 2-character work station ID of the display station to which the message is sent. The STATUS WORKSTN control command can be used to determine the work station IDs. If the work station ID of the system console is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console. If the first parameter is not entered and a message text is entered, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console. The message is placed in a message queue for the system console instead of in the queue for the display station assigned as the system console.

user-id: The 1- to 8-character user ID that identifies the operator to whom the message is sent. Each display station operator enters a user ID on the Sign On display when the session is started. The STATUS WORKSTN control command can be used to determine user IDs. If the user ID of the system operator is specified, the message is sent to the display station interface (the COMMAND display) at the display station assigned as the system console. If the first parameter is not entered and a message text is entered, the message is sent to the system console interface (the CONSOLE display) at the display station assigned as the system console.

message text: Up to 60 alphameric or special characters.

Note: For the ideographic version of the SSP, the message text can contain ideographic characters; however, the SSP does not send the message to a nonideographic display station. If an attempt is made to send an ideographic message to a nonideographic display station, the SSP issues an error message.

Example

A display station operator is preparing to submit a job that requires diskette BFILE. To inform the system operator that diskette BFILE is required, the display station operator enters:

```
MSG ,NEXT JOB REQUIRES DISKETTE BFILE
```

RELEASE Command

Function

The RELEASE command releases for printing a specified entry on the spool file, all individually held entries, or all individually held entries for the specified printer. It releases entries that were held on the spool file either by a HOLD command or by a PRIORITY-0 specification on the PRINTER OCL statement.

Format

```
RELEASE PRT  { ,spool-id }
(L)      (P) { ,ALLH     }
             { ,ws-id    }
```

Parameters

PRT: Indicates that a specified entry on the spool file, all individually held entries, or all individually held entries for the specified printer should be released.

Note: The PRT parameter, used alone, can be used by only the system operator. The subconsole operator must enter PRT with a second parameter specified, and the subconsole operator must control the printer on which the entries are to be printed.

spool-id: The 6-character spool file entry ID of the entry to be released. The STATUS PRT control command can be used to determine the spool file entry ID. The subconsole operator must control the printer on which the entry is destined to be printed.

ALLH: Indicates that all entries are to be released that were individually held by a HOLD control command or by a PRIORITY-0 specification on a PRINTER OCL statement. The ALLH parameter does not release the entire spool file if the entire spool file was held by the system operator. The ALLH parameter applies only to those spool file entries that are scheduled to be printed on a printer controlled by the subconsole.

ws-id: All entries for the specified printer are released. The subconsole operator must control the specified printer.

Example

To release entry SP0013 on the spool file, the system operator enters:

```
RELEASE PRT,SP0013
```

or

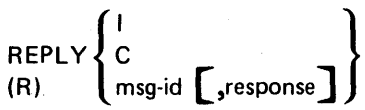
```
L P,SP0013
```

REPLY Command

Function

The REPLY command is used by the subconsole operator to respond to messages that were directed to the subconsole.

Format



Note: The command name (REPLY or R) is not required when msg-id is used.

Parameters

I: All informational messages on the display screen at the subconsole are responded to.

C: Compresses the display so that only messages still needing a response are displayed.

msg-id: The 2-character message ID that identifies the message being responded to. (The message ID is displayed along with the message on the display screen at the subconsole.)

response: The subconsole operator's response to the message. For example, if an SSP message is being responded to, the response might be 0, 1, 2, 3, or D.

Example

The following message appears on the display screen at the subconsole:

```
02 SYS-1405 OPTIONS(012 )
        DO YOU WANT SPOOL SEPARATOR PAGES ON PRINTER P1. . .
```

To select option 0, the subconsole operator enters:

```
REPLY 02,0

or

R 2,0

or

2,0
```

RESTART Command

Function

Restarts printing output for an entry being printed from the spool file to a printer controlled by the subconsole operator. The RESTART command restarts the printed output from the top of the first page or from the top of a specified page.

A RESTART control command can be used either to restart printing after a STOP PRT control command is entered or to restart printing an entry while it is being printed from the spool file.

Format

```
RESTART PRT,[page number] {,ws-id}
(T)        (P)
```

Parameters

PRT: Indicates that the printing of an entry from the spool file is to be restarted.

page number: The number of the page where printing is to restart. If the page number is not specified, printing restarts at the beginning of the printed output. The maximum page number is 65535.

ws-id: The work station ID of the printer being restarted. If a work station ID is not specified, the system printer is assumed. The subconsole operator must control the printer that is being restarted.

Note: The last parameter does not need to be specified if this subconsole controls the system printer (and if the system printer is the printer for which the spool writer is to be restarted).

Example

To restart printing of an entry on the spool file at the top of page 6 on printer P2, the subconsole operator enters:

```
RESTART PRT,6,P2
```

or

```
T P,6,P2
```

START Command

Function

Starts the spool writer for a specific printer or all printers that are controlled by the subconsole operator so that the printing of entries to the spool file can begin.

Format

START PRT, [forms number] {,ALL / ,ws-id}
(S) (P)

Parameters

PRT: Starts the spool writer to begin the printing of entries on the spool file. The PRT parameter can be used to start the spool writer after IPL if the autostart function was not selected during IPL. The PRT parameter can also be used to start the spool writer after the STOP PRT control command has been entered to stop the spool writer; printing begins with the first available entry on the spool file.

If a second parameter is specified, only entries using that specified forms number are printed.

If a third parameter is not specified, the spool writer for the system printer is started, provided this subconsole controls the system printer.

Note: If the subconsole operator enters START PRT while system activity is stopped as the result of a STOP SYSTEM command, the spool writer will not start until the system operator enters a START SYSTEM control command.

forms number: If this parameter is specified, only entries using the specified forms number are printed. If a forms number is not specified, printing begins with the first available entry on the spool file. The forms number may be 1 through 4 characters long.

ws-id: The work station ID of the printer for which the spool writer is to be started. The printer must be controlled by this subconsole operator.

ALL: The spool writers for all printers that are controlled by this subconsole are started.

Notes:
1. If ws-id or ALL is not specified, the system printer is assumed.
2. The last parameter does not need to be specified if this subconsole controls the system printer (and if the system printer is the printer for which the spool writer is to be started).

Example

To start printing all spool file entries with forms number 3333 on all printers controlled by the subconsole operator, the subconsole operator enters:

```
START PRT,3333,ALL
```

or

```
S P,3333,ALL
```

STATUS Command

Function

When used as a subconsole operator control command the STATUS command displays any of the following items:

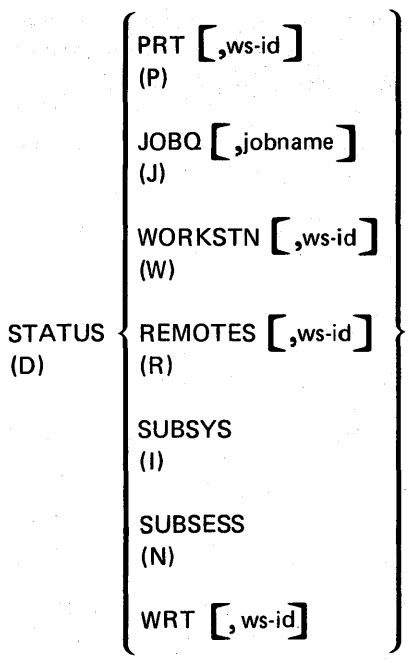
- The spool file entries for all printers or for a specified printer that the subconsole operator controls
- Any jobs on the input job queue that were entered from that display station
- The status of:
  - The local display stations and printers
  - The remote display stations and printers that are not offline
  - The diskette drive
- The status of the remote display stations and printers
- The status of enabled SSP-ICF subsystems
- The status of active SSP-ICF sessions
- The status of the spool writers

Each time status information is displayed on the display screen, the operator can:

- End the status display
- Display the next page of status information (if the last page is being displayed, the status display resets to the first page of status information)
- Enter a control command (except REPLY)
- Reset the status display to the first page of status information
- Request an updated status display

For a detailed description of the information appearing on status displays, see the *Operator's Guide.*

Format


STATUS
(D)
PRT [,ws-id]
(P)
JOBQ [,jobname]
(J)
WORKSTN [,ws-id]
(W)
REMOTES [,ws-id]
(R)
SUBSYS
(I)
SUBSESS
(N)
WRT [,ws-id]

Parameters

PRT: Displays information about the entries on the spool file that the subconsole operator controls or about the entries for a specified printer controlled by the subconsole operator.

JOBQ: Displays entries on the input job queue that were placed on the input job queue from that display station.

jobname: The 8-character, system-assigned job name of the job for which status information is displayed.

WORKSTN: Displays status information about:

- The local display stations and printers
- The remote display stations and printers that are not offline
- The diskette drive
- A selected device that satisfies one of the preceding criteria

REMOTES: Displays status information about the remote display stations and printers or about a specified remote display station or printer.

SUBSYS: Displays information about enabled SSP-ICF subsystems.

SUBSESS: Displays information about active SSP-ICF sessions.

WRT: Displays status information about the spool writers or about the spool writer for a specified printer.

ws-id: The work station ID of the display station or printer for which status information is to be displayed.

Example

To display the status of all jobs on the spool file controlled by the subconsole operator, the subconsole operator enters:

```
STATUS PRT
```

or

```
D P
```

STOP Command

Function

The STOP command stops the spool writer for a specified printer or for all printers controlled by the subconsole operator in order to prevent the printing of entries from the spool file.

Format

$$\underset{(P)}{\text{STOP}}\ \underset{(P)}{\text{PRT}},\ \begin{bmatrix}\text{PAGE}\\ \text{JOB}\end{bmatrix}\ \begin{Bmatrix}\text{,ALL}\\ \text{,ws-id}\end{Bmatrix}$$

Parameters

PRT: Stops the spool writer to prevent the printing of entries from the spool file. If the spool writer is printing an entry from the spool file when the command is entered and a second parameter is not specified, printing stops as soon as the command is processed. The subconsole operator can start the spool writer to resume the printing by entering the START PRT control command or the RESTART control command.

PAGE: Stops the spool writer at the end of the current page.

JOB: Stops the spool writer at the end of the current spool file entry.

ALL: The spool writers are stopped for all printers that are controlled by the subconsole.

ws-id: The work station ID of the printer for which the spool writer is stopped. If a work station ID is not specified, the system printer is assumed. The subconsole operator must control the printer for which the spool writer is stopped.

Note: The last parameter does not need to be specified if this subconsole controls the system printer (and if the system printer is the printer for which the spool writer is to be stopped).

TIME Command

Function

The TIME command displays the time of day and the system date. The time is based upon the time specified by the system operator during IPL and is displayed in the following format:

hours:minutes:seconds

The system date is displayed in the system date format:

mmddyy, ddmmyy, or yymmdd

Format

TIME

Parameters

None

Example

To display the time of day and the system date, the display station operator enters:

TIME

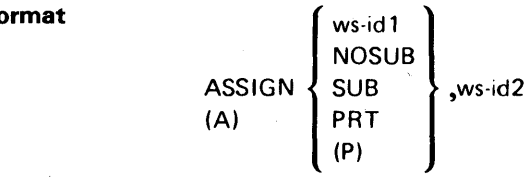
ASSIGN Command

Function

The ASSIGN command temporarily exchanges the IDs of two display stations or two printers, or temporarily assigns a printer as the system printer, or activates or de-activates subconsole support. The ASSIGN command can be used to assign an alternative display station or printer for an inoperative display station or printer. The original IDs are restored when IPL is performed.

Note: If the IDs of two devices are being exchanged, both devices must be offline. If subconsole support is being activated, the display station must have been configured as a subconsole, and must be offline before the command will be accepted. (You can use the VARY control command to place a device offline.)

Format



Parameters

ws-id1: The ID of a display station or printer. After the SSP processes the ASSIGN control command, the display station or printer will have the ID specified by ws-id2. (IDs were assigned during system configuration.)

NOSUB: De-activates subconsole support from the ID specified by ws-id2.

SUB: Activates subconsole support for the device specified by ws-id2.

PRT: Assigns the printer specified by ws-id2 as the system printer.

ws-id2: The ID of a display station that will temporarily exchange IDs with the device identified by ws-id1, or the ID of a display station that will have subconsole support either activated or de-activated. If PRT is entered in the first parameter position, ws-id2 is the work station ID of a printer that is to be assigned as the system printer.

Examples

Example A

Display station W3 is inoperative; to exchange IDs with the display station identified as W1, the system operator enters:

```
ASSIGN W1,W3
```

or

```
A W1,W3
```

Example B

To assign printer P2 as the system printer, the system operator enters:

```
ASSIGN PRT,P2
```

or

```
A P,P2
```

Example C

To de-activate subconsole support for display station W3, the system operator enters:

```
ASSIGN NOSUB,W3
```

or

```
A NOSUB,W3
```

Example D

To activate subconsole support for display station W3, the system operator enters:

```
ASSIGN SUB,W3
```

or

```
A SUB,W3
```

Note: ASSIGN SUB is needed in only two cases:

- If ASSIGN NOSUB has been issued for that work station
- If the subconsole is an alternative console and then was returned to alternative status because another display station was made the system console

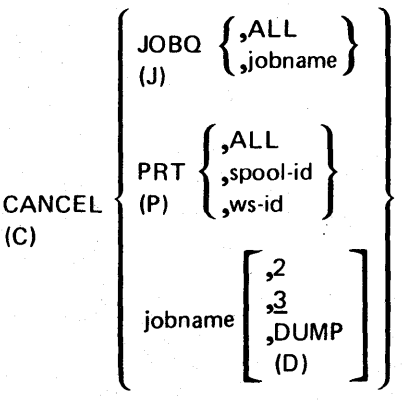
CANCEL Command

Function

When used as a system operator control command, the CANCEL command cancels any of the following:

- A specified job on the input job queue
- All jobs on the input job queue
- A specified entry on the spool file
- All entries on the spool file
- All entries on the spool file destined for a specified printer
- A currently executing job

Format



Parameters

JOBQ: Cancels the specified job (jobname) on the input job queue; if ALL is specified, cancels all jobs on the input job queue.

PRT: Cancels the specified entry (spool-id) on the spool file; if ALL is specified, cancels all entries on the spool file; if ws-id is specified, cancels all entries on the spool file that are destined for that printer.

ALL: Deletes all jobs from the input job queue or all entries from the spool file.

jobname: When specified in the first parameter position, jobname is the name of a currently executing job to be canceled. The second parameter specifies the type of termination. The STATUS USERS control command can be used to display the names of jobs currently executing.

When specified in the second parameter position, jobname is the name of a job on the input job queue to be canceled. The STATUS JOBQ control command can be used to display the names of jobs on the input job queue.

spool-id: The 6-character spool file entry ID of the spool file entry to be canceled. The STATUS PRT control command can be used to determine the spool file entry IDs.

ws-id: Deletes all entries from the spool file for the printer with the specified ws-id.

2: A controlled cancel is taken. All files being used by the job are closed. Remaining job steps are not executed.

3: An immediate cancel is taken. Files being used by the job are not closed. Remaining job steps are not executed.

Note: If 3 or DUMP (D) is specified when a job is canceled:

- New files created by the program that was executing are lost.
- If the program that was executing added records to a nonshared file, the added records are lost.

DUMP: An immediate cancel with a task dump is taken. Files being used by the job are not closed. The main storage assigned to the job is written into the system dump area on disk. Remaining job steps are not executed.

Note: If 3 or DUMP (D) is specified when a job is canceled:

- New files created by the program that was executing are lost.
- Records added to a nonshared file by the program that was executing are lost.

Examples

Example A

To cancel all jobs on the input job queue, the system operator enters:

```
CANCEL JOBQ,ALL
```

or

```
C J,ALL
```

Example B

To immediately cancel job W2001820, which is currently executing, the system operator enters:

```
CANCEL W2001820
```

or

```
C W2001820
```

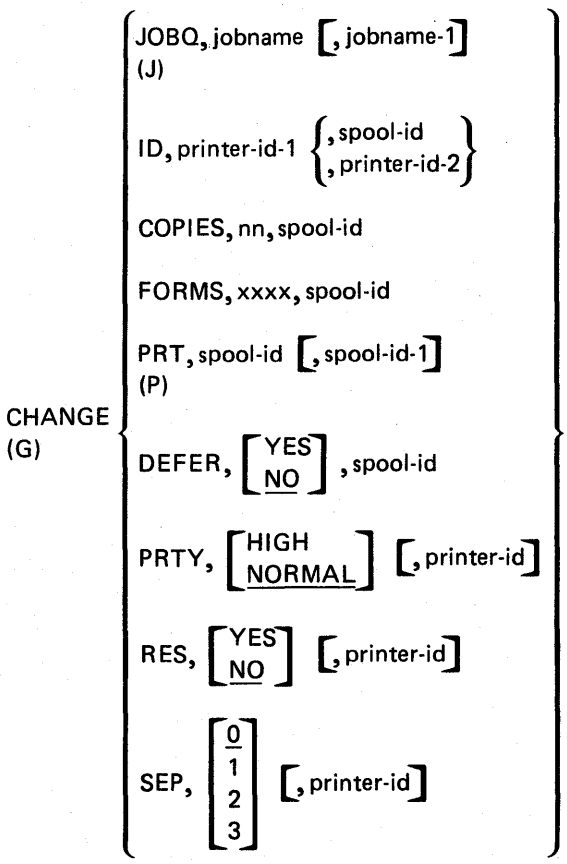
CHANGE Command

Function

The CHANGE command changes the following items:

- The position of a job on the input job queue or an entry on the spool file
- The printer used by an entry, or entries on the spool file
- The number of copies to be printed for an entry on the spool file
- The forms number to be used for an entry on the spool file
- The defer status of a spool file entry that is currently being intercepted
- The priority of the spool writer
- The resident/swappable attribute of the spool writer
- The number of separator pages printed by the spool writer before printing each spool file entry

Format



Parameters

JOBQ: Changes the position of a job on the input job queue.

PRT: Changes the position of an entry on the spool file.

ID: Changes the printer used for an entry or a group of entries on the spool file.

COPIES: Changes the number of copies of printed output for an entry on the spool file.

FORMS: Changes the forms number to be used for an entry on the spool file.

DEFER: Changes the defer status of a spool file entry currently being intercepted.

PRTY: Changes the priority of the spool writer.

RES: Changes the resident/swappable attribute for the spool writer.

SEP: Changes the number of separator pages the spool writer prints before printing each spool file entry.

jobname: The 8-character, system-assigned job name of the job to be changed. The STATUS JOBQ control command can be used to determine the job name.

jobname-1: The 8-character, system-assigned job name of a job on the input job queue. The job being changed (jobname) is placed on the input job queue following the job with the name specified by jobname-1. The job being changed assumes the same job queue priority as the job with the name specified in jobname-1. If jobname-1 is not specified, the job being changed is moved to the front of the queue and has a job queue priority of 5. Refer to the JOBQ command earlier in this chapter for more information on job queue priority.

spool-id: The 6-character spool file entry ID of an entry on the spool file. The STATUS PRT control command can be used to determine the spool file entry ID.

spool-id-1: The 6-character spool file entry ID of an entry on the spool file. The entry being changed (spool-id) is placed on the spool file following the entry with the ID specified by spool-id-1. If spool-id-1 is not specified, the entry being changed is moved to the front of the spool file with a priority value of 5. For information about spool file entry priorities, see the *Concepts and Design Guide*.

printer-id: The printer ID of the printer. If a printer ID is not specified, the system printer is assumed.

If PRTY is specified in the first parameter position, the priority of the spool writer for the printer identified by printer-id is changed. The new spool writer priority remains in effect until changed by another CHANGE command or until IPL is performed.

If RES is specified in the first parameter position, the resident/swappable attribute of the spool writer for the printer identified by printer-id is changed. You must first use the STOP command to cause the spool writer to stop printing. The new resident/swappable attribute remains in effect until changed by another CHANGE command or until IPL is performed.

If SEP is specified in the first parameter position, the number of separator pages printed by the spool writer for the printer identified by printer-id is changed. The new number of separator pages remains in effect until changed by another CHANGE command or until IPL is performed.

printer-id-1: The printer ID of the new printer to be used. If a spool file entry ID is specified in the third parameter position, that entry will use the new printer (printer-id-1). If a printer ID is specified in the third parameter position (printer-id-2), all spool file entries currently using the printer specified by printer-id-2 will use the new printer (printer-id-1).

printer-id-2: The printer ID of the existing spool file entries to be changed to the printer ID specified by printer-id-1.

nn: The number of copies to be printed for the entry. nn can be any decimal number from 1 through 99.

xxxx: The forms number of the forms to be used for the specified entry; 1 through 4 characters.

YES: If DEFER is specified in the first parameter position, the system will not begin printing spooled output from the specified spool file entry until the entry is completely intercepted. If the spool file entry is being intercepted and printed when the CHANGE command is entered, printing of the entry is stopped, and another spool file entry, if one exists, begins printing. Once the spool file entry has been completely intercepted, the entry can then be printed.

If RES is specified in the first parameter position, the spool writer for the printer identified by printer-id is to be resident. If YES is specified, you must first use the STOP command to cause the spool writer to stop printing.

NO: If DEFER is specified in the first parameter position, the spooled output from the specified spool file job can begin printing before the job is completely intercepted. If a parameter is not specified, NO is assumed.

If RES is specified in the first parameter position, the spool writer for the printer identified by printer-id is not to be resident. You must first use the STOP command to cause the spool writer to stop printing. If a parameter is not specified, NO is assumed.

HIGH: The spool writer for the printer identified by printer-id is to have a high priority.

NORMAL: The spool writer for the printer identified by printer-id is to have a normal priority. If a parameter is not specified, NORMAL is assumed.

0,1,2,3: The number of separator pages the spool writer for the printer identified by printer-id prints before each spool file entry. If a parameter is not specified, 0 (zero) separator pages is assumed.

Examples

Example A

To move job W3083001 on the input job queue so that it follows job W1051050, the system operator enters:

```
CHANGE JOBQ,W3083001,W1051050
```

or

```
G J,W3083001,W1051050
```

Example B

To change entry SP0011 on the spool file so that five copies of its output are printed, the system operator enters:

```
CHANGE COPIES,5,SP0011
```

or

```
G COPIES,5,SP0011
```

HOLD Command

Function

The HOLD command prevents a specified entry, all entries on the spool file for the specified printer, or the entire spool file from being printed. The HOLD command has the same effect as specifying PRIORITY-0 on the PRINTER OCL statement.

If a HOLD command specifies an entry that is being printed, the printing of that entry is interrupted and the printing of the next entry on the spool file begins. The held entry is eligible to be printed when a RELEASE control command releases that entry.

If a HOLD command specifying that the entire spool file should be held is entered while an entry is being printed, the remainder of that entry is printed.

Format

```
HOLD PRT [,spool-id]
(H)  (P) [,ws-id  ]
```

Parameters

PRT: Indicates that an entry, all entries on the spool file for the specified printer, or the entire spool file should be held. If a second parameter is not specified, the entire spool file is held.

spool-id: The 6-character spool file entry ID of the entry to be held. The STATUS PRT control command can be used to determine the spool file entry ID.

ws-id: The ID of the printer for which spool file entries will be held from printing.

Example

To hold entry SP0036 on the spool file, the system operator enters:

```
HOLD PRT,SP0036
```

or

```
H P,SP0036
```

To hold the entire spool file, the system operator enters:

```
HOLD PRT
```

or

```
H P
```

IDELETE Command

Function

When used as a system operator control command, the IDELETE command specifies whether informational messages directed to the system console should be suppressed.

Format

IDELETE [ON / OFF]

Parameters

ON: The SSP does not display informational messages. (The system operator does not have to respond to informational messages.) The audible alarm does not sound and the message light does not come on when an informational message is received. If a parameter is not specified, ON is assumed.

OFF: The SSP displays informational messages.

Example

To suppress displaying of informational messages directed to the system console (for example, messages directed to the system console via the // * OCL statement), the system operator enters:

```
IDELETE
```

Note: Messages displayed with the // ** OCL statement are not suppressed.

MSG Command

Function

When used as a system operator control command, the MSG command sends a message to all display stations or to a selected display station or display station operator.

Format

MSG { ALL / ws-id / user-id } , message text

Parameters

ALL: Sends the message to all active display stations, except the system console itself.

Note: If the message text contains ideographic characters, the SSP sends the message to only the display stations that can display ideographic characters. The SSP displays an informational message at the system console if the message was not sent to one or more display stations.

ws-id: The 2-character work station ID of the display station to which the message is sent. The STATUS WORKSTN control command can be used to determine the work station ID.

user-id: The user ID that identifies the operator to whom the message is sent. Each display station operator enters a user ID on the SIGN ON display when the session is started. The STATUS WORKSTN control command can be used to determine the user ID.

Note: If more than one operator is signed on with a given user-id, only one of the operators will receive the message.

message text: Up to 60 alphameric or special characters.

Note: For the ideographic version of the SSP, the message text can contain ideographic characters; however, the SSP does not send the message to a nonideographic display station. If an attempt is made to send an ideographic message to a nonideographic display station, the SSP issues an error message.

Example

To notify the operator of display station W1 that no more jobs can be entered, the system operator enters:

```
MSG W1,PLEASE DO NOT SUBMIT ANY MORE JOBS
```

PRTY Command

Function

When used as a system operator control command, the PRTY command changes the execution priority of a currently executing job or a job on the input job queue.

Format

```
PRTY jobname [ ,ON     ]
             [ ,HIGH   ]
             [ ,MEDIUM ]
             [ ,OFF    ]
             [ ,NORMAL ]
             [ ,LOW    ]
```

Parameters

jobname: The 8-character, system-assigned job name of the job whose priority is to be changed. The STATUS USERS or STATUS JOBQ control command can be used to determine the job name.

ON or *HIGH:* System resources are assigned to the job before they are assigned to a lower priority (MEDIUM, NORMAL, OFF, or LOW) job. If a second parameter is not specified, ON is assumed.

MEDIUM: System resources are assigned to the job before they are assigned to a lower priority (NORMAL, OFF, or LOW) job.

OFF or *NORMAL:* System resources are assigned to the job before they are assigned to a LOW priority job.

LOW: System resources are assigned to the job after all other higher priority jobs have been assigned system resources.

Example

To assign high priority to job W2010112, the system operator enters:

```
PRTY W2010112,HIGH
```

RELEASE Command

Function

The RELEASE command releases for printing a specified entry on the spool file, all individually held entries, all individually held entries for the specified printer, or the entire spool file. It releases entries that were held on the spool file either by a HOLD command or by a PRIORITY-0 specification on the PRINTER OCL statement.

Format

```
RELEASE PRT  [ ,spool-id ]
(L)     (P)  [ ,ALLH     ]
             [ ,ws-id    ]
```

Parameters

PRT: Indicates that a specified entry on the spool file, all individually held entries, all individually held entries for the specified printer, or the entire spool file should be released. If a second parameter is not specified, the entire spool file is released; entries that were individually held by a HOLD control command or by a PRIORITY-0 specification on a PRINTER OCL statement are still held.

spool-id: The 6-character spool file entry ID of the entry to be released. The STATUS PRT control command can be used to determine the spool file entry ID.

ALLH: Indicates that all entries that were individually held by a HOLD control command or by a PRIORITY-0 specification on a PRINTER OCL statement are to be released. The ALLH parameter will not release the entire spool file if the spool file was held by a HOLD PRT command.

ws-id: The ID of the printer to which entries will be released for printing. All entries for the specified printer are released.

Examples

Example A

To release entry SP0013 on the spool file, the system operator enters:

```
RELEASE PRT,SP0013
```

or

```
L P,SP0013
```

Example B

To release the entire spool file, except those entries that were individually held, the system operator enters:

```
RELEASE PRT
```

or

```
L P
```

Example C

To release only those jobs that were individually held, the system operator enters:

```
RELEASE PRT,ALLH
```

or

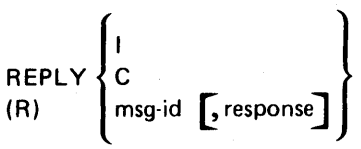
```
L P,ALLH
```

REPLY Command

Function

The REPLY command is used by the system operator to respond to messages that were directed to the system console, or to respond to subconsole messages displayed on the system console through the status message function.

Format



Note: The command name (REPLY or R) is not required when msg-id is used.

Parameters

I: All informational messages on the display screen at the system console are responded to.

C: Compresses the display so that only messages still needing a response are displayed.

msg-id: The 2-character message ID that identifies the message being responded to, or the 2-character message ID that identifies the message being responded to for the subconsole currently being displayed on the status message display. (The message ID is displayed along with the message on the display screen at the system console.)

Note: If a message with the specified message ID (msg-id) exists for the subconsole, it will be responded to even if it has not yet been displayed on the status message display.

response: The system operator's response to the message. For example, if an SSP message is being responded to, the response might be 0, 1, 2, 3, or D.

Example

The following message appears on the display screen at the system console:

```
02 SYS-1405 OPTIONS (012 )
        DO YOU WANT SPOOL SEPARATOR PAGES ON PRINTER P1 . . .
```

To select option 0, the system operator enters:

```
REPLY 02,0
```

or

```
R 2,0
```

or

```
2,0
```

RESTART Command

Function

Restarts printing output for an entry being printed by the spool writer.

The RESTART command restarts the printed output from the top of the first page or from the top of a specified page.

A RESTART control command can be used to restart printing:

- After a STOP PRT control command is entered.
- For an entry being printed from the spool file.

Format

```
RESTART PRT, [page number] [,ws-id]
(T)         (P)
```

Parameters

PRT: Indicates that the printing of an entry from the spool file is to be restarted.

page number: The number of the page where printing is to restart. If the page number is not specified, printing restarts at the beginning of the printed output. The maximum page number is 65535.

ws-id: The work station ID of the printer. (Work station IDs are assigned during system configuration.) If a work station ID is not specified, the system printer is assumed.

Example

To restart printing of the current spool file entry at the top of page 6 on the system printer, the system operator enters:

```
RESTART PRT,6
```

or

```
T P,6
```

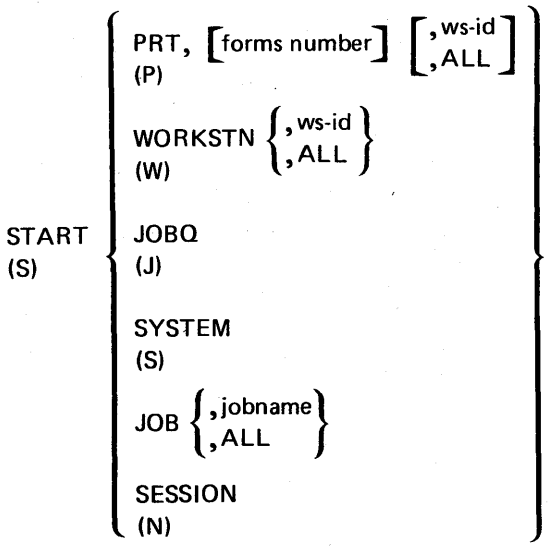
START Command

Function

The START command performs any of the following functions:

- Starts the spool writers for all printers or for a specified printer to begin the printing of entries on the spool file
- Allows the initiation of jobs from all display stations or from a specified display station for which a STOP WORKSTN control command was entered
- Starts the running of jobs on the input job queue
- Resumes the system activity that was stopped by a STOP SYSTEM control command
- Resumes the execution of a job, or all jobs, that were stopped by a STOP JOB control command
- Resumes the SSP-ICF activity that was stopped by a STOP SESSION control command

Format



Parameters

PRT: Starts the spool writer(s) to begin the printing of entries on the spool file. The PRT parameter can be used to start the spool writer after IPL if the autostart function was not selected during IPL. The PRT parameter can also be used to start the spool writer after the STOP PRT control command has been entered to stop the spool writer; printing begins with the first available entry on the spool file.

If a second parameter is specified, only entries using that specified forms number are printed.

If a third parameter is not specified, the spool writer for the system printer is started.

Note: If the system operator enters START PRT while system activity is stopped as the result of a STOP SYSTEM command, the spool writer will not start until the system operator enters a START SYSTEM control command.

WORKSTN: Allows initiation of jobs and entry of control commands from all display stations (if ALL is entered in the second parameter position), or from a specified display station for which a STOP WORKSTN control command was entered.

JOBQ: Starts running jobs on the input job queue. When the input job queue becomes empty, running of jobs begins automatically when one or more jobs are placed on the queue.

SYSTEM: Resumes the system activity that was stopped by a STOP SYSTEM control command.

JOB: Resumes execution of a job, or all jobs, that were stopped by a STOP JOB control command.

forms number: The forms number used by the spool file entries to be printed; 1 through 4 characters. Only entries using the specified forms number are printed.

ws-id: The work station ID of the display station or printer to be started. Work station IDs are defined during system configuration.

SESSION: Resumes the SSP-ICF activity that was stopped by a STOP SESSION control command.

ALL: If PRT is specified as the first parameter, the spool writers for all printers are started.

If WORKSTN is specified as the first parameter, ALL allows the initiation of jobs and the entry of control commands stopped by a STOP WORKSTN,ALL control command.

If JOB is specified as the first parameter, the execution of all jobs stopped by a STOP JOB,ALL control command resumes.

jobname: The 8-character, system-assigned job name of the job to be started. The STATUS USERS control command can be used to determine the job name.

Examples

Example A

The initiation of jobs from the input job queue was stopped by a STOP JOBQ control command. To start running jobs from the input job queue, the system operator enters:

```
START JOBQ
```

Example B

Initiation of all jobs, including jobs from the input job queue and entries from spool file, was stopped by a STOP SYSTEM,ALL control command. To allow initiation of jobs on the system, the system operator enters:

```
START SYSTEM
```

or

```
S S
```

STATUS Command

Function

When used as a system operator control command, the STATUS command displays any of the following items:

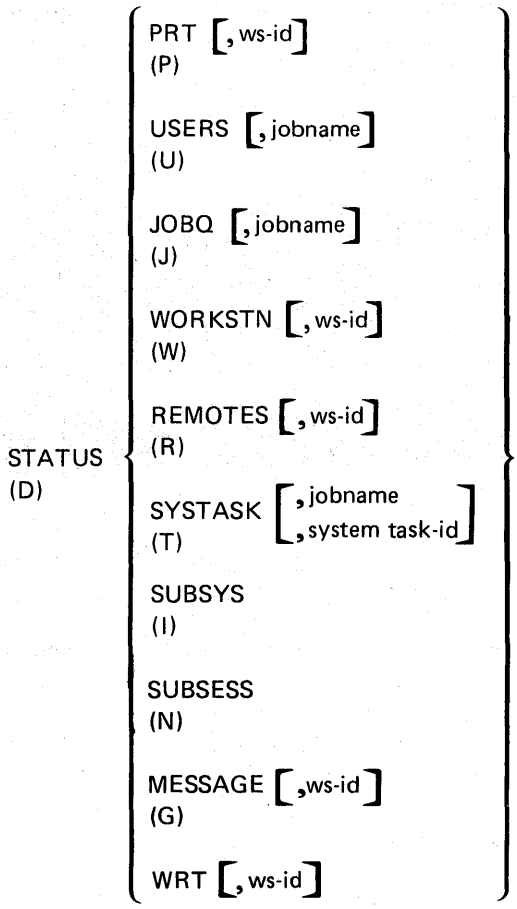
- The spool file entries for all printers or for a specified printer
- The status of jobs running on the system
- Any entries on the input job queue
- The status of the local display stations and printers, the remote display stations and printers that are not offline, and the diskette drive
- The status of the remote display stations and printers
- The status of the tasks running on the system
- The status of enabled SSP-ICF subsystems
- The status of active SSP-ICF sessions
- Any subconsole messages still needing a response
- The status of the spool writers

Each time status information is displayed on the display screen, the operator can:

- End the status display
- Display the next page of status information (if the last page is being displayed, the status display resets to the first page of status information)
- Enter a control command
- Reset the status display to the first page of status information
- Request an updated status display

For a detailed description of the information appearing on status displays, see the *Operator's Guide*.

Format



Parameters

PRT: Displays information about all entries on the spool file or about the entries for a specified printer.

USERS: Displays the status of a selected job, or all jobs, running on the system.

JOBQ: Displays entries on the input job queue.

WORKSTN: Displays status information about:

- The local display stations and printers
- The remote display stations and printers that are not offline
- The diskette drive

or

- A selected device that satisfies one of the preceding criteria

REMOTES: Displays status information about the remote display stations and printers or about a specified remote display station or printer.

SYSTASK: Displays the active tasks in the system and their TCB (task control block) addresses. This information is used if you run the SETDUMP command described in Appendix D.

SUBSYS: Displays information about enabled SSP-ICF subsystems.

SUBSESS: Displays information about active SSP-ICF sessions.

MESSAGE: Displays the messages that are not replied to for each of the subconsoles or for a specific subconsole only.

WRT: Displays information about the spool writers or about the spool writer for a specified printer.

ws-id: The work station ID of the display station or printer for which status information is to be displayed.

jobname: The 8-character, system-assigned job name of the job for which status information is displayed.

system task id: The 2-character ID that identifies the TCB (task control block) as a system task.

Example

To display the status of all jobs on the spool file, the system operator enters:

```
STATUS PRT
```

or

```
D P
```

STOP Command

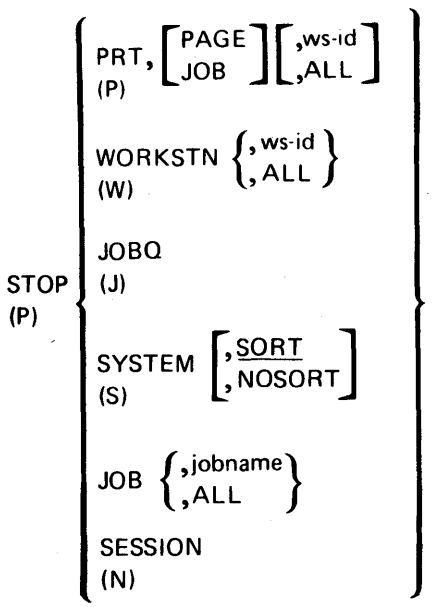
Function

The STOP command performs any of the following functions:

- Stops the spool writers for all printers or a specified printer to prevent the printing of entries from the spool file
- Stops the initiation of jobs from all display stations other than the system console or stops the initiation of jobs from a specified display station
- Stops the initiation of jobs from the input job queue
- Begins an orderly shutdown of the system
- Stops the execution of all jobs or a specified job
- Stops the initiation of jobs from incoming SSP-ICF sessions

Note: STOP SYSTEM is not allowed if any SSP-ICF subsystems are enabled.

Format



Parameters

PRT: Stops the spool writer(s) to prevent the printing of entries on the spool file. If a work station ID is not specified, the system printer is assumed. If the spool writer is printing an entry from the spool file when the command is entered and a second parameter is not specified, printing stops as soon as the command is processed. The system operator can start the spool writer(s) to resume the printing by entering the START PRT control command or the RESTART control command.

PAGE: Stops the spool writer at the end of the current page.

JOB: If PRT is specified in the first parameter position, stops the spool writer at the end of the current spool file entry. If JOB is specified as the first parameter, all executing jobs are suspended.

ws-id: If PRT is the first parameter, ws-id is the work station ID of the printer for which the spool writer is stopped. If a work station ID is not specified, the system printer is assumed.

If WORKSTN is the first parameter, ws-id is the work station ID of the display station from which no new jobs can be initiated.

ALL: If PRT is specified as the first parameter, the spool writers for all printers are stopped. If WORKSTN is specified as the first parameter, no new jobs can be initiated and certain control commands cannot be entered from a display station.

WORKSTN: Stops the initiation of jobs and the entry of certain control commands from all display stations except the system console. All display station jobs currently running continue to run. If a work station ID is entered in the second parameter position, only initiation from the specified display station is stopped.

JOBQ: Stops the initiation of jobs from the input job queue. The job that is already running continues until completed.

SYSTEM: The system operator uses the STOP SYSTEM control command to begin an orderly shutdown of system activities. STOP SYSTEM causes the following to occur:

- For each display station input operation to a program, a condition code that indicates that a STOP SYSTEM control command was entered is returned to the program along with the input data. Programs should check for this condition, close all files, and go to end of job as soon as possible.

- Initiation of jobs and certain control commands on the system is stopped. No new jobs can be initiated from the input job queue, the spool file, or any display station including the system console.

SORT: Index keys are sorted as part of the system shutdown. If SYSTEM is specified as the first parameter, and a second parameter is not specified, SORT is assumed.

NOSORT: Index keys are not sorted.

JOB: If JOB is specified in the first parameter position, stops the execution of all jobs and the initiation of jobs from all work stations, except from the system console, or stops the execution of a specified job (jobname). The STOP JOB control command can be used to: (1) suspend a job that appears to be holding control of the system or (2) suspend all jobs in order to run an important job or a job that can be run only when no other jobs are running.

To start the initiation of jobs from all work stations or from a specified work station, use the system operator START WORKSTN command.

jobname: The 8-character, system-assigned job name of the job to be stopped. The STATUS USERS control command can be used to determine the job name.

SESSION: Stops the initiation of jobs from incoming SSP-ICF sessions.

Example

The system operator is preparing for system shutdown. To stop the initiation of jobs from the display stations and the input job queue, and the processing of entries from the spool file, the system operator enters:

```
STOP SYSTEM
```

TIME Command

Function

The TIME command displays the time of day and the system date. The time is based upon the time specified by the system operator during IPL and is displayed in the following format:

hours:minutes:seconds

The system date is displayed in the system date format:

mmddyy, ddmmyy, or yymmdd

The system date advances when the time reaches 24:00:00.

Format

TIME

Parameters

None

Example

To display the time of day and the system date, the system operator enters:

TIME

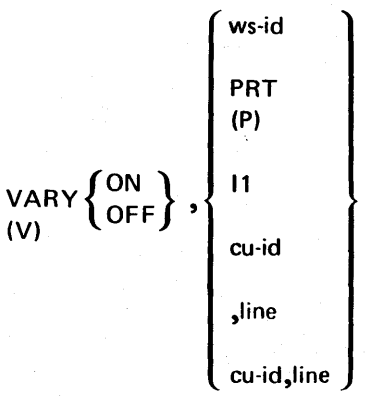
VARY Command

Function

The VARY command changes the status of a display station, a printer, the system printer, or the diskette drive from online to offline or from offline to online. Offline devices cannot be allocated for use by programs on the system. The VARY command cannot be used to take offline a device that is allocated to a program unless an I/O error has occurred on the device.

The operator can use a STATUS WORKSTN or STATUS REMOTES control command to determine the current status of devices on the system.

Format



Parameters

ON: The specified device is placed online.

OFF: The specified device is placed offline.

Note: If the device(s) varied offline is a subconsole, any messages that were queued to the subconsole are sent to the system console.

ws-id: The work station ID of a display station or printer. The status of the specified device is changed. Work station IDs are assigned during system configuration.

PRT: The status of the system printer is changed.

I1: The status of the diskette drive is changed.

cu-id: The 3-character controller ID. All work stations associated with the specified controller are placed online or offline.

line: The communications line number. When this parameter is entered without the cu-id parameter, all controllers and work stations associated with this line are placed online or offline. When this parameter is entered with the cu-id parameter on the VARY ON command, the specified device is placed online on the specified line.

Note: The parameter *cu-id,line* is valid only for switched communications lines.

Example

The system printer has been taken offline. To place the system printer back online, the system operator enters:

```
VARY ON,PRT
```

or

```
V ON,PRT
```

Chapter 4. SSP Utility Programs

This chapter describes the following utility programs, which are supplied as part of the SSP:

$ARSP	$DUPRD	$INIT	$PNLM	$RSTRT
$BACK	$FBLD	$LABEL	$POST	$SETCF
$BICR	$FREE	$LOADI	$PRES	$SFGR
$BMENU	$HELP	$MAINT	$PRMN	$SLFL
$BUILD	$HIST	$MGBLD	$PROF	$UASC
$COPY	$HSML	$$MMSP	$PRON	$UASF
$DDST	$IDSET	$MMST	$PRST	$XNLM
$DELET	$IEDS	$PACK	$PRSV	XREST
	$IENBL	$PDSR	$RENAM	$XSAVE

Note that the SSP utility programs used for data communications ($MRJE, $SRJE, $DCFUP, and $DCSUP) are not described in this manual. Descriptions of these programs can be found in the *Data Communications Reference Manual*. Also, the $CNFIG utility program used during system configuration is not described in this manual. $CNFIG is described in the *Installation and Modification Reference Manual*.

For each utility program described in this chapter, the following information is given:

- The function of the utility program
- The format of the utility control statements (see *Conventions Used for Describing Statement Formats* at the front of this manual)
- Descriptions of the parameters for the utility control statements
- The control statement sequence required to run the utility program
- An example, or examples, of how the utility program is used

MAIN STORAGE REQUIREMENTS

Each SSP utility program requires 14 K bytes of main storage.

RULES FOR CODING UTILITY CONTROL STATEMENTS

The rules for coding utility control statements are:

1. *Statement identifier.* Two slashes (//) in positions 1 and 2, followed by a blank, must precede the statement identifier. Do not use blanks within the identifier.

2. *Blanks.* Use one or more blanks between the identifier and the first parameter.

3. *Statement parameters.* Keyword parameters can be in any order, but positional parameters must always appear in the same position relative to other positional parameters in the statement. Use a comma to separate one parameter from another. Use a hyphen (-) within each keyword parameter to separate the keyword from the information you supply. Do not use blanks between parameters; do not use blanks within a parameter unless the parameter contains a value enclosed by single quotation marks (for example, 'CONSTANT VALUE').

   The following is an example of a utility control statement:

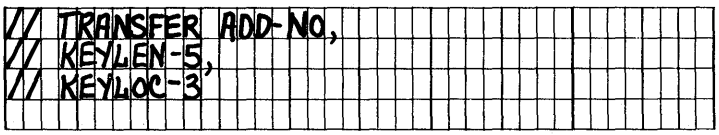
   The statement identifier is COPYI1. The keyword parameters are NAME and PACK. The information supplied by the parameters is JOE and 123456.

4. *END control statement.* This utility control statement indicates the end of utility control statements for a utility. An END control statement must be the last control statement entered for a utility if utility control statements are used.

5. *Continuation.* Some utility control statements can be entered as two or more records. A record can consist of a maximum of 120 characters, including blanks and commas. A utility control statement can be continued if statement parameters are entered.

   Rules for using continuation are:
   a. Place a comma after the last parameter in every record except the last record. The comma, followed by a blank, tells the system that the statement is continued in the next record.
   b. Begin each new record with // in positions 1 and 2.
   c. Leave one or more blanks between the // and the first parameter in the record.

   The following is an example of a continued utility control statement:

6. *Comments.* Comments can be included in utility control statements in the following places:
   a. After the last parameter in a utility control statement (a comment cannot be entered if no parameters are coded on a statement that has parameters defined) or in a record that is continued. (See the description of continuation in item 5.) Leave one or more blanks between the last parameter and the comment.

      In the following example, the comment is after the last parameter.

```
// REMOVE LABEL-ALL,UNIT-I1,PACK-XYZ  COMMENT A
```

      In the following example, a comment is included on a continued record.

```
// REMOVE LABEL-FILE1,UNIT-F1,          COMMENT B
//        DATA-YES                      COMMENT C
```

   b. After the statement identifier in a utility control statement that has no parameters defined. Leave one or more blanks between the statement identifier and the comment. For example:

```
// END END OF JOB STEP 3
```

   c. As a separate record. An asterisk (*) must be entered in position 1 followed by one or more blanks and the comment. For example:

```
* COMMENT D
```

      Note: If sequence numbers are used on utility control statements, the sequence numbers are considered comments and the rules for coding comments apply.

$ARSP—Auto Response Utility Program

Function

The $ARSP utility program places an auto response value in a message load member. A message load member is a special type of library load member in which a message text associated with a message identification code (MIC) is stored. The $ARSP utility program is executed when the RESPONSE procedure is run.

Note: For *all situations* where a message might be encountered and its associated auto response value applied, you should carefully evaluate the effect of taking that response value in your environment. The *only* messages and auto response values that are warranted for use are listed in Examples A and B.

Auto Response Source Member

An auto response source member is input to $ARSP. The auto response source member is created like any other library source member. For example, it can be copied from a display station to a library or from one library to another by the $MAINT utility program. The auto response source member can also be entered by a program such as the source entry utility (SEU) described in the *IBM System/34 Source Entry Utility Reference Manual*, SC21-7657.

Auto response source members can contain three types of statements: auto response control statements (alpha codes) and auto response specification statements (MIC numbers) are mandatory; comment statements are optional.

To enter statements in an auto response source member, enter an auto response control statement followed by an auto response specification statement for each MIC number.

If a source member contains more than one alpha code, you must enter an auto response control statement (alpha code) followed by the MIC numbers associated with that alpha code before you enter the next alpha code. For example, if both SYS and RPG messages exist in a source member, you might enter all the MIC numbers associated with the alpha code SYS before you enter RPG and the MIC numbers associated with it. You can enter the alpha codes in any sequence you choose, but their associated MIC numbers must be in ascending order.

Auto Response Control Statement

The auto response control statement specifies the alpha code for the auto response specification statements (MIC numbers) that follow. The alpha codes are the characters displayed immediately before the MIC numbers when an error message is displayed. For example, SYS is the alpha code for SYS-1395, and 1395 is the MIC number.

If USER is the alpha code specified in the auto response control statement, you must specify the member name and the library name containing the message load member when you enter the auto response control statement.

The format of the auto response control statement is:

alpha code [,member name] [,library name] [comment]

alpha code: The alpha code of the messages being updated with auto response values. The alpha code must be one of the following:

SYS	DFU	ASM	EMU
KBD	SEU	FORT	ESU
RPG	SDA	CBL	NRD
SORT	WSU	BAS	USER

member name: The name of the message load member to be updated. If the alpha code is not USER, $ARSP determines the member name, and the parameter is ignored. If the alpha code is USER, the parameter is required.

library name: The name of the library which contains the message load member. If the alpha code is not USER, $ARSP determines the library name, and the parameter is ignored. If the alpha code is USER, you can either specify the library name or let the system default to #LIBRARY.

comment: A comment preceded by one or more blanks may be added for information.

Auto Response Specification Statement

The auto response specification statement specifies the message identification code (MIC) of the message to be updated and the auto response value for that message. The utility updates only the messages specified in the auto response specification statements. Auto response values for other messages remain unchanged.

If an auto response specification statement is specified for a message that is not in a message load member, $ARSP will issue an error message.

The format of the auto response specification statement is:

```
mic  auto response [comment]
```

mic: The message identification code. The MIC number must be specified as a 4-character decimal number from 0000 through 9999, and must be entered in columns 1 through 4 of the auto response specification statement. You must enter MIC numbers in ascending order, according to alpha code. For example, for SYS messages, enter 1395 before 2557.

auto response: The auto response value. The auto response value must be 0, 1, 2, 3, D, or N. Enter the value in column 6 of the auto response specification statement. If you specify N, any existing auto response value for the message identified by the alpha code and MIC number will be removed. If you specify 0, 1, 2, 3, or D, this value is placed in the message load member, replacing any auto response value that may already exist for the message.

comment: A comment preceded by one or more blanks may be added for information.

Comment Statement (optional)

The format of the comment statement is:

```
* comment
```

Comment statements must have an asterisk (*) as the first character (column 1), followed by a blank. Comment statements can be interspersed within the auto response control statements and the auto response specification statements. These statements provide additional information about the auto response values, but do not affect the function of the utility.

WARNING

In the following cases, the System/34 auto response facility user should not specify auto response values:

1. The message indicates permanent input/output errors.
2. The message indicates that a dump has been taken.
3. The message requires operator intervention before an option can be taken.
4. The message requires retry operations.
5. The messages are auto response facility error messages.

When you choose auto response values, make sure that they do not result in lost data or fill the history file with repetitive messages.

PROGRAMMING CONSIDERATIONS

Consider the following when using the System/34 auto response facility to supply option values:

- The auto response facility will supply values only if all the following are true:
  - The message is being sent to the system console.
  - The message requires an option response.
  - The response is a value that the message allows.
- The option value may be changed or deleted after it is placed in the message load member.
- Both the message and the value supplied by auto response are logged to the history file.
- You may want to maintain two source members. One of these would contain auto response values; the other would contain the auto response control statements and auto response specification statements showing the value N. By maintaining two source members, you can use one to turn the auto response facility on for unattended operation and one to turn the auto response facility off for attended operation.

Example A

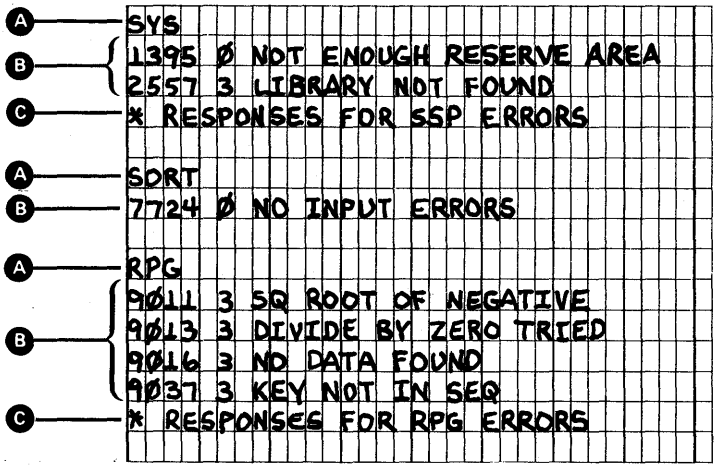
The following example shows the correct format for entering the auto response control statement and the auto response specification statement for error message RPG-1012:

(A) Auto response control statement

(B) Auto response specification statement, including default value and comment statement

Example B

The following example shows the correct sequence for entering auto response control statements, auto response spcification statements, and comment statements. The messages and auto response values shown below are warranted for use in the Finance Subsystem (SFS) environment.



- **A** Auto response control statements
- **B** Auto response specification statements, including default values and comments
- **C** Comment statements

Utility Control Statement Format

// RESPONSE SOURCE-source name [,LIBRARY- {library name / #LIBRARY}]

Parameters

SOURCE: Specifies the name of the auto response source member that contains the auto response control statement and the auto response specification statements required to update the message load member(s).

LIBRARY: Specifies the name of the library containing the auto response source member. #LIBRARY is the default.

Control Statement Sequence

```
// LOAD $ARSP
// RUN
// RESPONSE SOURCE-source name, . . .
// END
```

$BACK–Backup Library Utility Program

Function

The $BACK utility program compresses the entire system library and copies it to a diskette file.

When the system library is copied to the diskette(s), library members are shifted to remove gaps between them (unused space between members is collected at the end of the library). The output diskette(s) must not contain active files.

More than one diskette may be required to contain the system library. When this situation arises, the operator is automatically instructed to insert another diskette, after which processing resumes.

For information about determining the number of backup diskettes required to contain the library, see the *Installation and Modification Reference Manual.*

To reconstruct on the disk a library that was backed up on (copied to) diskettes, use the RELOAD procedure or perform an IPL from the diskettes created by $BACK. The vol-id of the first (or only) diskette containing the library becomes the vol-id of the disk after the reload operation.

For a step-by-step description of how to reload the system library, see the *Operator's Guide.*

$BACK can be run only from the system console and cannot be run while any other jobs are being run.

$BACK is executed when the BACKUP procedure is run.

Utility Control Statement Format

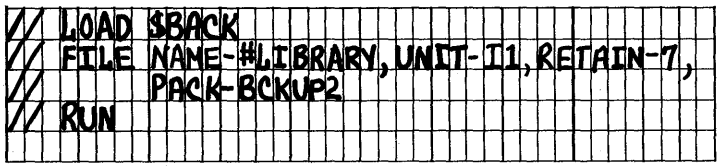
Utility control statements are not used.

Control Statement Sequence

```
// LOAD $BACK
// FILE NAME-#LIBRARY,UNIT-I1, . . .
// RUN
```

Example

Back up the system library onto a diskette labeled BCKUP2. The diskette file will be called #LIBRARY and should be saved for seven days:



$BICR–Basic Data Exchange Utility Program

Function

The $BICR utility program converts a disk file to a basic data exchange or an I exchange file on diskette, converts a basic data exchange or an I exchange diskette file to a sequential or an indexed disk file, adds a basic data exchange or an I exchange diskette file to a sequential disk file, or displays the contents of a basic data exchange or an I exchange diskette file. All basic data exchange diskette files that are input for $BICR must be in the 128-byte format for diskette 1 diskettes (one-sided diskettes) or in the 256-byte format for diskette 2D diskettes (two-sided, double-density diskettes). All basic data exchange diskette files created by $BICR are in the 128-byte (for diskette 1) or the 256-byte (for diskette 2D) basic data exchange format.

For information about diskette formats, see Appendix C.

Notes:

1. An offline, multivolume file extent on disk cannot be converted by $BICR to a basic data exchange or an I exchange file.
2. Deleted records in a delete-capable file are not transferred to diskette. Also, if a delete capable file is transferred to diskette, when it is transferred back to disk, it will no longer be delete capable.
3. Files will not be extended when you transfer data from diskette to disk.

When a basic data exchange diskette file is added to an existing disk file, the records in the diskette file are either truncated or padded with hexadecimal zeros (hex 00) to conform to the record length of the disk file. When an I exchange diskette file is added to an existing disk file, the record length of the diskette file must be the same as the record length of the diskette file.

When a new disk file is created from a basic data exchange or an I exchange diskette file, the record length of the disk file is set to that of the diskette file. When a new basic data exchange diskette file is created from a disk file, the record length of the diskette file is set to that of the disk file or to 128 (for diskette 1) or 256 (for diskette 2D), whichever is smaller. When an I exchange diskette file is created from a disk file, the record length of the diskette file is set to that of the disk file. When a disk file is being added to an existing basic exchange or I exchange diskette file, the record length of the diskette file must be the same as the record length of the disk file. If the record length of a basic exchange diskette file is equal to the diskette sector size (128 bytes for diskette 1 diskettes, 256 bytes for diskette 2D diskettes), but the record length of the disk file to be added to the diskette file is greater than the diskette sector size, the disk records are truncated to the diskette sector size.

$BICR processes records consecutively during file conversion. If input for $BICR is an indexed disk file, records are read sequentially by key.

$BICR is executed when the TRANSFER procedure is run.

Utility Control Statement Formats

To create a basic data exchange diskette file from a disk file, or to convert a basic data exchange or an I exchange diskette file to a disk sequential file:

```
[// TRANSFER]
// END
```

To create an I exchange diskette file from a disk file:

```
// TRANSFER [ADD-NO] , FORMAT-IFORMAT
// END
```

To add the data in a basic data exchange or an I exchange diskette file to a disk sequential file, or to add data in a disk file to a basic data exchange diskette file:

```
// TRANSFER ADD-YES
// END
```

To add data in a disk file to an I exchange diskette file:

```
// TRANSFER ADD-YES,FORMAT-IFORMAT
// END
```

To create an indexed file on the disk from a basic data exchange or an I exchange diskette file:

```
// TRANSFER [ADD-NO],KEYLEN-value,KEYLOC-value
// END
```

To display the contents of a basic data exchange or an I exchange diskette file:

```
// DISPLAY [FROM-{value 1 / 1}][,TO-value 2]
// END
```

Parameters

For the TRANSFER statement:

ADD: ADD-YES specifies that records in the input file are added to the output file. The first record from the input file is placed after the last record in the existing output file. If the input file (specified on the COPYIN FILE statement) is on a diskette(s), records are added to the disk file until the end of either the diskette file or the disk file is reached. However, the add operation is not started unless the space available on the disk file is large enough to contain the records in the file segment on the diskette currently inserted. If the input file is on disk, then records are added to the diskette file until the end of the disk file is reached. The output diskette file must be physically the last file on the diskette.

KEYLEN: The length of the record keys when an indexed file is to be created on the disk. The value can be any decimal number from 1 through 29.

Note: KEYLEN must be specified with KEYLOC, and the sum of their values must not exceed the record length plus 1.

KEYLOC: The relative displacement of the start position of the record key in the records. For basic data exchange files, the value can be any decimal number from 1 through 128 for diskette 1, or 1 through 256 for diskette 2D. For I exchange files, the value can be any decimal number from 1 through 4096.

Note: KEYLOC must be specified with KEYLEN, and the sum of their values must not exceed record length plus 1.

For the DISPLAY statement:

FROM: Value 1 specifies the first record to be displayed. If FROM is not specified, the first record in the file is assumed.

TO: Value 2 specifies the last record to be displayed. If TO is not specified, the last record in the file is assumed. The maximum value that can be specified is 8 388 607.

Control Statement Sequence

```
// LOAD $BICR
// FILE NAME-COPYIN,UNIT-{I1 / F1},LABEL-from file label . . .
[// FILE NAME-COPYO,UNIT-{I1 / F1},LABEL-to file label . . .]
// RUN
[// TRANSFER]
[// DISPLAY ]
// END
```

Notes:

1. If a new disk file is to be created from a multivolume diskette file, the COPYO FILE statement must be given, and the required RECORDS (or BLOCKS) parameter must be large enough to contain the entire diskette file.
2. If a new disk file is to be created from a single-volume diskette file (with the same space requirement and same label as the diskette file), do not specify the COPYO FILE statement.
3. If a new disk file larger than the diskette file is to be created, the COPYO FILE statement must be specified with the required RECORDS (or BLOCKS) parameter.
4. If a file is being created on diskette, the COPYO FILE statement with a PACK parameter is required.

Example

Create a basic data exchange diskette file (JOEB1) from a disk file (JOE):

```
// LOAD $BICR
// FILE NAME-COPYIN,UNIT-F1,LABEL-JOE
// FILE NAME-COPYO,UNIT-I1,LABEL-JOEB1,PACK-9
// RUN
// TRANSFER
// END
```

$BMENU–Build Menu Utility Program

Function

The $BMENU utility program creates the screen format load member that along with the command load member, is required to display a fixed-format menu or a free-format menu. (For information about fixed-format menus and free-format menus, see the description of the BLDMENU procedure in Chapter 2.)

Input to $BMENU consists of one or two message load members located in the library that is to contain the screen format load member for the menu:

- The *command load member*, which is required, is a second-level message load member that contains the statements to be used as system input when the operator selects items from the menu.

- The *display text load member* is a first-level message load member that is optional for fixed-format menus and is required for free-format menus. The display text load member for a fixed-format menu defines the descriptive text to be displayed along with each item number on the menu. The display text load member for a free-format menu defines the contents of lines 3 through 20 on the menu display. If, however, a free-formed format menu is to be displayed on a 960-character display, lines 15 through 20 will not be displayed. Lines 3 through 8 are displayed in the first half of the menu display, and lines 9 through 14 are displayed in the second half of the menu display.

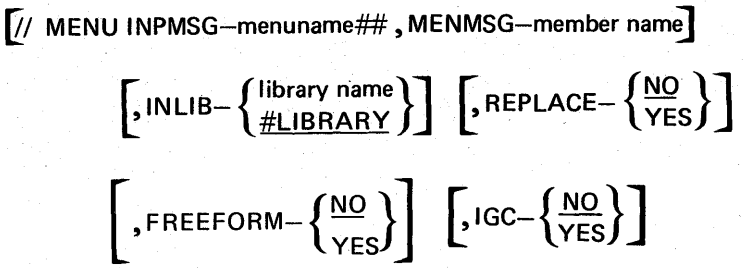
For information about the contents of the source members from which the command load member and the display text load member are created, see the description of the BLDMENU procedure in Chapter 2. For information about how to create message load members from message source members, see the description of the CREATE procedure in Chapter 2 or the description of the $MGBLD utility program later in this chapter.

Output from $BMENU consists of:

- The *display screen format load member* for the menu, which contains the menu display. $BMENU places the display screen format load member in the specified library. The display screen format load member is one of two library members that must exist to display a menu. The other required member is the command load member.

- A *listing of the menu* and *warning messages*.

Utility Control Statement Format



Parameters

INPMSG: Specifies the name of the command load member. The last 2 characters of the member name must be ##. The characters preceding the ## are the 1- through 6-character name of the menu to be generated.

MENMSG: Specifies the 1- through 8-character name of the display text load member, if one exists. The name must be a valid member name and cannot be the same as the menu name. If MENMSG is not specified, $BMENU uses the information in the command load member to generate the descriptive text for the items in the menu.

INLIB: Specifies the name of the library containing the command load member and the display text load member, if one is used. $BMENU also places the screen format load member for the menu in the specified library. If INLIB is not specified, the system library (#LIBRARY) is assumed.

REPLACE: REPLACE-YES specifies that if a screen format load member already exists with the same name as the menu being created, $BMENU automatically deletes the existing member.

CAUTION

If a menu is rebuilt while it is being displayed at a display station, the display station operator should request the rebuilt menu by entering a MENU control command. If this is not done, the old version of the menu is displayed, and it may not correspond to the command input defined for the new menu.

If REPLACE-NO is specified and a duplicate load member exists with the same name as the menu being created, $BMENU issues a message. The operator must then decide whether to delete the existing member or to cancel the job. If the REPLACE parameter is not specified, REPLACE-NO is assumed.

FREEFORM: FREEFORM-NO specifies that a fixed-format menu should be built. If the FREEFORM parameter is not specified, FREEFORM-NO is assumed.

FREEFORM-YES specifies that a free-format menu should be built.

IGC: The IGC parameter is used for the ideographic version of the SSP and is ignored for nonideographic systems. IGC-NO specifies that the system-generated text (for example, the ENTER NUMBER, COMMAND, or OCL prompt) should not be displayed as ideographic characters. If the IGC parameter is not specified, IGC-NO is assumed.

IGC-YES specifies that the system-generated text should be displayed as ideographic characters and ideographic data can be entered into the input field of the menu screen.

CAUTION
If a menu is built using the IGC parameter, the menu cannot be displayed from a nonideographic terminal.

Control Statement Sequence

```
// LOAD $BMENU
// RUN
// MENU INPMSG-menuname## . . .
// END
```

Example

Use the $BMENU utility program to build a fixed-format menu called MENU01. Figure 4-1 shows MENU01 as it should appear on the display screen.

```
COMMAND                                                          W1
                              MENU: MENU01
 1. CUSTOMER BILLING KEYING          13. DISPLAY MENU2
 2. LUMBER IN STOCK                  14.
 3. LUMBER ON ORDER                  15.
 4. ACCOUNTS RECEIVABLE              16.
 5. ACCOUNTS PAYABLE                 17.
 6. PAYROLL                          18.
 7. INVENTORY CHECKLIST              19.
 8. NEW ACCOUNTS                     20.
 9.                                  21.
10. BUILDER CONTRACTS                22.
11. PRINT LUMBER IN STOCK            23.
12. PROCESS STATEMENTS               24.




ENTER NUMBER, COMMAND, OR OCL.

                                                          <- READY
```

Figure 4-1. Display of MENU01

To generate MENU01, perform the following steps:

Step 1. Create the required command load member. To create the command load member, you must first create the source member from which the load member is to be generated. For this example, the source member will be called MENU01## and the load member will be called MENU01. Both will be located in library ULIB2.

You can use the $MAINT utility program to create the source member by entering the following statements from the keyboard:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-S,NAME-MENU01##,
//         TO-ULIB2,RECL-65
MENU01##,2
0001 BILLINPT DAILY,,T1
0002 LUMBER STOCK
0003 LUMBER ORDER
0004 ACTSRCV
0005 ACTSPAY
0006 PAYROLL WEEK,MGT
0007 INVENT CHECK
0008 NEWACTS WEEK
0010 CONTR BLDRS
0011 PRINT LUMBER
0012 STMTS
0013 MENU MENU2
// CEND
// END
```

Statements to be placed in the command source member. See the description of the $MGBLD utility program later in this chapter, for an explanation of the statements in a message source member.

After creating the source member, you can use the CREATE procedure or the $MGBLD utility program to generate the command load member. The following procedure command could be used:

```
CREATE MENU01##,REPLACE,ULIB2
```

Step 2. Create the optional display text load member (the member that specifies the text to be displayed along with each item number). To create the display text load member, you must first create the source member from which the load member will be generated. For this example, assume the source and load members will both be called MENU01DT and will be located in library ULIB2.

You can use the $MAINT utility program to create the display text source member by entering the following statements from the keyboard:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-S,NAME-MENU01DT,
//      TO-ULIB2,RECL-80
MENU01DT,1
0001 CUSTOMER BILLING KEYING
0002 LUMBER IN STOCK
0003 LUMBER ON ORDER
0004 ACCOUNTS RECEIVABLE
0005 ACCOUNTS PAYABLE
0006 PAYROLL
0007 INVENTORY CHECKLIST
0008 NEW ACCOUNTS
0010 BUILDER CONTRACTS
0011 PRINT LUMBER IN STOCK
0012 PROCESS STATEMENTS
0013 DISPLAY MENU2
// CEND
// END
```

After creating the source member, you can use the CREATE procedure or the $MGBLD utility program to generate the display text load member. The following procedure command could be used:

```
CREATE MENU01DT,REPLACE,ULIB2
```

Step 3. Run the $BMENU utility program. Enter the following statements to create the format load member for MENU01:

```
// LOAD $BMENU
// RUN
// MENU INPMSG-MENU01##,MENMSG-MENU01DT,
//      INLIB-ULIB2,REPLACE-YES
// END
```

$BUILD–Alternative Sector Rebuild Utility Program

Function

The $BUILD utility program allows you to display and correct data on the disk after a disk error occurs. $BUILD can be run only from the system console and only when no other jobs are running on the system.

When a disk read or write error occurs, the data is written to an *alternative sector*. Disk alternative sectors are reserved for use in place of defective disk sectors. The $BUILD utility program searches the alternative sectors of the disk for data that was unreadable because of a read error. Each sector containing unreadable data is printed along with the sector logically preceding and the sector logically following it in the file.

The data is displayed on the display screen and printed by the printer in both character and hexadecimal formats, as shown in Figure 4-2. The data is displayed in character format on the first line. If the character cannot be displayed, it is replaced by a blank. The data is also displayed in hexadecimal format on the second and third lines. The leftmost hexadecimal digit of each byte is on the second line, and the rightmost hexadecimal digit is below it on the third line.

After the unreadable data is displayed, you have two options:

- Bypass the data by pressing Cmd key 3. The $BUILD utility then searches for the next alternative sector with unreadable data. The next time $BUILD is run, the bypassed sector will again be displayed.

- Correct the unreadable data by using the function keys to display the portion of the bad sector to be corrected. After shifting the display to the desired position, place the cursor on either the character data line or the hexadecimal data line. Enter the desired data in place of the unreadable data. The display screen provides the following information to help you correct the data:
  - The decimal displacement into the sector of the first character displayed: COL-00001 on the display screen in Figure 4-2
  - The sector number: SS-025462 on the display screen in Figure 4-2
  - The file label: FILENAME-SAMPLE on the display screen in Figure 4-2

After keying corrections in the bad sector (if any corrections are required), press Cmd key 1. The corrected sector will be rewritten on the disk, and $BUILD will search for the next bad sector. The next time $BUILD is run, the corrected sector will not be displayed.

Note: If you restore from diskette the disk file containing the unreadable sector, or if you recreate that file, you do not have to run the $BUILD utility program.

$BUILD is executed when the BUILD procedure is run.

Printer Output

```
    DATE  03/31/77      TIME  11.06                **ALTERNATE SECTOR REBUILD UTILITY**

    SS-025461      FILENAME-SAMPLE
SECTOR BEFORE BAD DATA
1.......10........20........30........40........50........60........70........80........90.......100.......110.......120.....128
        &       J X         &     O #
00470650470600E000006A0650205D07000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00350900360360107000037090ZF0563800000000000C0000C000000000C0C000000000000000000C0CCCCCCC000000000000000000000000000000000
131......140.......150.......160.......170.......180.......190.......200.......210.......220.......230.......240.......250...256

00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000C0CCCCCCC000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000C0C0C0C0000000000000000000000000000000000
    SS-025462      FILENAME-SAMPLE
SECTOR WITH BAD DATA
1.......10........20........30........40........50........60........70........80........90.......100.......110.......120.....128
0#PTFLOG                P$PTF       &               P#ERR                   P#FORMAT    &               P#FROMLIB
D7DECDDC4000000000080000000D5DEC4444000500000000000000D7CDDD4444000401000080000000D7CDDDCE4CCC5CCCCC080000000D7CDDDCCC0006C2C
6B736367000000000000080010-007B736000000100400000000101007B5990C0C0C028CF0CCC0000105007B66941300C70CEC0C000001020078696439200940?C
131......140.......150.......160.......170.......180.......190.......200.......210.......220.......230.......240.......250...256
          P#RESTORE                 P#SETRM     &               P#SFROMLI                   P#TRANS
00800000000007DCEEDDC0004010000800000000D7ECEDD44001500000000000C0C0D7ECDDDDC001602000C0000000000D7EDCCE4400140100008000000000FFFF
00000001060078952369500FB3000000000103007B25394CC0020050000000010100?B269643900340600000C0105007B391520000800200000000010200FFFF
    SS-025463      FILENAME-SAMPLE
SECTOR AFTER BAD DATA
1.......10........20........30........40........50........60........70........80........90.......100.......110.......120.....128
P#TRANSNA   &               PAGGDX      &               PAGGDXH     &               PAGGFB      &               PAGGFP      &
D7EDCDDC001500000080000000DCCCCE4440015010000000000000DCCCCCEC4400150100000C0000000DCCCCC4440025000CCCCC0000000000DCCCCD4440025000
7B391525100A000F000000000102007177470000C001000000001030071774780000FC030C0C0C0C0010400717762000300500000000101007177670000040050
131......140.......150.......160.......170.......180.......190.......200.......210.......220.......230.......240.......250...256
          PALTERSC                  PAPAR       &               PAPPLYPTF                   PASM
00000000000CDECDCEC0024010000800000000DCDD4444002500000800000C0C0DCDDDEDEC0024010C0C800C0000000DCEC4444400270000004000000000FFFF
00000001010071335922300589030000000010300717719000000800000000001030071773873600B8070C0C000C01030071240000000E80C000000C000200FFFF
```

Display Screen Output

```
0#PTFLOG                    P$PTF       &               P#ERR

D7DECDDC4000000000080000000D5DEC4444000500000000000000D7CDD4444000401000080000
6B736367000000000000080010-007B736000000100400000000101007B599000000280F000000001

COL-00001     SS-025462     FILENAME-SAMPLE
```

Figure 4-2. Sample Output of a Disk Sector with Character and Hexadecimal Printout

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $BUILD
// PRINTER NAME-$SYSLIST,SPOOL-NO,LINES-66
// RUN
```

$COPY–Disk Copy/Display Utility Program

Function

The $COPY utility program performs the following functions:

- Copies an entire file from the disk to diskette(s), from diskette(s) to the disk, or from the disk to another location on the disk to:
  - Provide a duplicate of a file.
  - Move a file to a larger disk area.
- Deletes records from a file. Selected records are omitted from the copy; the original file remains unchanged.
- Copies a portion of a file. Selected records can be deleted from the copy.
- Copies all disk files (except #LIBRARY, user libraries, the spool file, system files, offline multivolume files, or a diskette file that was not created by $COPY) to diskette(s) to create a backup copy of the files or to obtain more space on the disk; or restores previously copied files from diskette(s) to the disk. When all files are copied from disk to diskette, $COPY must be requested from the system console and no other jobs can be running.
- Copies all members of a specified file group to diskette(s). File labels of the files that belong to the file group contain a period. The characters preceding the period identify the file group, and the characters following the period identify the file within the group. As for all file labels, the maximum number of characters is 8. Examples of labels of files within a file group are:

  A.INV, A.ACCTS, A.PROLL — Files in file group A

  A1.INV, A1.ACCTS, A1.PROLL — Files in file group A1

  Files with labels that do not contain a period are not part of a file group.

  Note: Files on diskette cannot be processed as members of a file group.

- Copies all disk data files that are not members of a file group (except #LIBRARY, user libraries, and the spool file) to diskette(s).
- Copies an indexed file and puts the records in key order (reorganizes the file) to improve, in some cases, the performance of programs that use the file. Selected records can be deleted from the copy.
- Adds a disk file to an existing diskette file.
- Displays all or part of a file (either on the display screen or printer, depending on the current system list device assigned to the requesting display station).
- Creates an indexed file from an existing sequential, direct, or indexed file and deletes selected records.

Notes:

1. $COPY will not process a disk file that is #LIBRARY, a user library, a spool file, system files, an offline multivolume file, or a diskette file that was not created by $COPY. When you are copying all disk files, all members of a specified file group, or all files not a member of a file group, the receiving diskettes must not contain unexpired files. When you are copying all files from diskette back to disk, the process must start with the first file on the first diskette.
2. $COPY will not copy or display an offline multivolume file extent on disk.
3. Files will not be extended when you copy data from diskette to disk.

$COPY is executed when the DISPLAY, LISTFILE, ORGANIZE, RESTORE, or SAVE procedure is run.

Utility Control Statement Formats

To copy an entire file:

```
// COPYFILE OUTPUT-DISK [,DELETE-{'position,c'
                               'position,cc'
                               'position,Xdd'
                               'position,Xdddd'
                               'position,ideographic-constant'
                               SYSDEL}]

                     [,REORG-{NO}]
                             {YES}

// END
```

To copy a portion of a file:

```
// COPYFILE OUTPUT-DISK [,DELETE-{'position,c'
                               'position,cc'
                               'position,Xdd'
                               'position,Xdddd'
                               'position,ideographic-constant'
                               SYSDEL}]

                     [,REORG-{NO}]
                             {YES}

[// SELECT KEY,FROM-'key'
 // SELECT KEY,FROM-'key',TO-'key'
 // SELECT RECORD,FROM-number
 // SELECT RECORD,FROM-number,TO-number
 // SELECT PKY,FROM-'key'
 // SELECT PKY,FROM-'key',TO-'key']
// END
```

To restore previously copied files from diskette to the disk:

```
// COPYALL TO-F1
// END
```

To copy all disk files (except #LIBRARY, user libraries, and the spool file) to diskette:

```
// COPYALL TO-I1,GROUP-ALL
// END
```

To copy all disk files that are members of a specified file group to diskette:

```
// COPYALL TO-I1,GROUP-file group
// END
```

To copy all disk files that are not members of a file group (except #LIBRARY, user libraries, and the spool file) to diskette(s):

```
// COPYALL TO-I1
// END
```

Note: When the COPYALL statement is used with GROUP-ALL specified, $COPY must have been requested from the system console, and no other jobs can be running.

To copy a sequential, direct, or indexed file to an indexed file:

```
                                  [                 {'position,c'                    }]
                                  [                 {'position,cc'                   }]
// COPYFILE OUTPUT-DISK           [ ,DELETE-        {'position,xdd'                  }]
                                  [                 {'position,xdddd'                }]
                                  [                 {'position,ideographic-constant' }]
// KEY LENGTH-value1,POSITION-value2
// END
```

To add a disk file to an existing diskette file:

```
// COPYADD
// END
```

To display an entire file:

```
// COPYFILE {OUTPUT-PRINT} [,IGC-{NO}]
            {OUTPTX-PRINT}     {YES}
// END
```

To display part of a file:

```
                                  [                 {'position,c'                    }]
                                  [                 {'position,cc'                   }]
// COPYFILE {OUTPUT-PRINT}        [ ,DELETE-        {'position,Xdd'                  }]
            {OUTPTX-PRINT}        [                 {'position,Xdddd'                }]
                                  [                 {'position,ideographic-constant' }]
                                  [                 {SYSDEL                          }]

                  [,IGC-{NO}]
                        {YES}

[// SELECT KEY,FROM-'key'                                  ]
[// SELECT KEY,FROM-'key',TO-'key'                         ]
[// SELECT RECORD,FROM-number                               ]
[// SELECT RECORD,FROM-number,TO-number                     ]
[// SELECT PKY,FROM-'key'                                  ]
[// SELECT PKY,FROM-'key',TO-'key'                         ]
// END
```

Parameters

For the COPYFILE statement:

OUTPUT: Specifies the output device for the copy operation.

- OUTPUT-DISK specifies that the output device is the disk or diskette. The COPYIN and COPYO FILE statements in the OCL sequence for $COPY define the file to be copied (COPYIN) and the new file (COPYO) created by the copy operation. When a file that was copied to a diskette is copied back to the disk, the file retains its original format (label, size, retention type), unless the original format is overridden by the appropriate parameter(s) (LABEL, BLOCKS [or RECORDS], or RETAIN) on the COPYO FILE statement. See *Control Statement Sequence* for the FILE statements required to run $COPY.

  Notes:
  1. $COPY cannot be used to copy basic data exchange diskette files. The TRANSFER procedure or the $BICR utility program must be used to copy basic data exchange diskette files.
  2. If the COPYFILE statement is being used to restore one file from diskette(s) containing all files previously copied from the disk (that is, the COPYALL statement was originally used), the label specified on the COPYIN FILE statement should be the label that identified the disk file when it was copied. For example, if a disk file labeled PAYROLL was one of the disk files copied to a diskette file called #SAVE, specify LABEL-PAYROLL on the COPYIN FILE statement to restore only the PAYROLL file.
  3. If you are copying file groups, DISP-SHR must not be specified on the COPYIN FILE OCL statement.

- OUTPUT-PRINT specifies that the file or part of the file is to be displayed on a system list device for the requesting display station. The COPYIN FILE statement in the OCL sequence for $COPY defines the file to be displayed.

  For information about assigning the system list device, see the description of the SYSLIST procedure in Chapter 2.

  Note: When the system list device is the display screen, you control the record display in the following manner:

  - To display the next record, press Cmd key 1.
  - To display the previous record, press Cmd key 2. (Cmd key 2 is ignored if the file being displayed is a diskette file, SELECT-KEY or SELECT-PKY was specified, or the displayed record is the first record in the file.)
  - To terminate the display, press Cmd key 3.
  - To display the next bytes of the displayed record, press the Shift and Roll Up keys.
  - To display the previous bytes of the displayed record, press Shift and Roll Down keys.

When OUTPUT-PRINT is specified for an indexed file, records are displayed in record order, unless SELECT KEY, SELECT PKY, or REORG-YES is specified. For each record, $COPY displays the record key followed by the contents of the record.

When OUTPUT-PRINT is specified for a sequential or direct file, records are displayed in the order they appear in the file. For each record, $COPY displays the relative record number followed by the contents of the record.

$COPY uses as many lines as it needs to display the contents of a record (100 characters per line are printed; if the display screen is used, 80 characters per screen are displayed). Characters that have no graphic display symbol are displayed as 2-digit hexadecimal numbers in over-and-under format. In the following example, the character following the letter F has a hexadecimal value of B6, but does not have a graphic display symbol:

```
ABCDEF J12345
      B
      6
```

After displaying the last record in a file, $COPY displays a message that states the number of records displayed.

OUTPTX: OUTPTX-PRINT specifies that the file, or part of the file, is to be displayed on the system list device for the requesting display station and that each character should be displayed along with the 2-digit hexadecimal number that represents the character in over-and-under format. (Ideographic characters require a 4-digit hexadecimal representation. For further information, see the description of the IGC parameter, later in this section.)

For information about assigning the system list device, see the description of the SYSLIST procedure in Chapter 2.

The description of the OUTPUT parameter describes the format of the information displayed and describes how you control the record display if the system list device is the display screen.

Note: If the system list device is the display screen, OUTPUT and OUTPTX have the same effect.

DELETE: Specifies records that should be deleted. The DELETE parameter is optional except when REORG-YES is specified for a sequential file. The DELETE parameter can take any of the following forms:

Form	Meaning
'position,c'	Delete any record with the specified character in the specified position. For example, DELETE-'50,S' deletes records with an S in position 50.
'position,cc'	Delete any record with the specified two characters beginning in the specified position. For example, DELETE-'50,SS' deletes records with SS in positions 50 and 51.
'position,Xdd'	Delete any record with the specified character in the specified position. Xdd is the hexadecimal value of the character. For example, DELETE-'50,XE2' deletes records with an S in position 50. (See Appendix F for standard characters and their hexadecimal equivalents.)
'position,Xdddd'	Delete any record with the specified two characters beginning in the specified position. Xdddd is the hexadecimal value of the characters. For example, DELETE'50,XE2E2' deletes records with SS in positions 50 and 51.
'position,ideographic-constant'	This form of the DELETE parameter is valid only on systems with ideographic character support and deletes records with a specified ideographic character beginning in a specified position. The ideographic constant portion of the parameter must be bracketed by the *shift out*, ($^{O}_{E}$) and the *shift in* ($^{O}_{F}$) control characters.

CAUTION
If two non-IGC characters are in the specified position, which happens to have an EBCDIC equivalent to the 2 bytes of the IGC character, that record will be deleted. Care should be used when dealing with files whose records are not all in the same format.

Form	Meaning
SYSDEL	Removes system-deleted records from the file. *Note:* If you specify SYSDEL for a file that is not delete capable, or on a system that is not configured with extended disk data management, the SSP will generate a halt (0, 2, 3 options). If the 0 option is selected, the system ignores the parameter. When a delete-capable file is copied to disk or diskette, it retains its delete-capable attribute. The delete-capable attribute can be changed when a file is copied from a diskette to disk or from disk to disk by use of the DFILE parameter on the COPYO file statement. By using the DFILE parameter, you can change a file from delete capable to not delete capable, and from not delete capable to delete capable.

IGC: The IGC parameter is used for the ideographic version of the SSP and is ignored for nonideographic systems. IGC-YES specifies that the file might contain ideographic characters, and if possible, $COPY should display those characters. When OUTPUT-PRINT is specified along with IGC-YES, the hexadecimal representation of ideographic characters are not displayed, even if the characters are not displayable.

Whenever the SYSLIST device is capable of displaying ideographic characters and the system supports the ideographic feature, the records displayed by $COPY will displayed in ideographic format.

Note: When nonideographic characters are printed, 100 characters are printed on each line. The characters are printed in print positions 1 through 100. When a mixture of ideographic and nonideographic characters are printed, the number of characters printed on a line is variable. Each ideographic character represents 2 bytes of information and requires 2 print positions on the line. If an ideographic character begins in print position 100, position 100 is left blank and the character is printed in positions 0 and 1 on the next line. (Position 0 is staggered 1 position to the left of the normal starting position, which is position 1.) To determine the position of a character within the record, you can use the following algorithm:

$$\begin{pmatrix}\text{position in}\\ \text{record}\end{pmatrix} = 100 \times \begin{pmatrix}\text{number of}\\ \text{previous lines}\end{pmatrix} + \begin{pmatrix}\text{print position of}\\ \text{the character}\end{pmatrix}$$

For example, if an ideographic character begins in position 0 of the third line of printout, the character begins in byte 200 of the record (100 x 2 + 0 = 200).

IGC-NO specifies that the file contents should be displayed just as they would be for a nonideographic system. Any ideographic data in the file is treated as 1-byte alphameric characters.

REORG: REORG-YES specifies that:

- If an indexed disk file is copied, records are copied in the same order as their keys appear in the index.
- If a sequential disk file is copied, records should be deleted. The DELETE parameter is required.

REORG-NO specifies that records are copied in the same order as in the copied file. If the REORG parameter is not specified, REORG-NO is assumed.

For the KEY statement:

LENGTH: Specifies the length of the key in bytes. The length can be any value from 1 through 29, but the sum of the specified length and key position (POSITION) cannot exceed the record length plus 1.

POSITION: Specifies the position of the key in the record. The value specified is the position of the leftmost byte of the key. The position can be any value from 1 through 999.

For the SELECT statement:

KEY or *PKY:* Specifies that a specified portion of an indexed file is to be copied. If PKY is specified, the indexed file contains packed keys.

Note: If a portion of an indexed file is to be copied from diskette, the records in the indexed file must have been saved on the diskette in key sequence. For information on ordering the records of an indexed file in key sequence, refer to the REORG parameter or the ORGANIZE procedure.

FROM-'key': Specifies the key (or the beginning characters of the key) of the first record to be copied or displayed. If packed keys are used, up to 57 numeric characters can be specified. If none of the keys in the file begin with the specified characters, the record with the next higher key is the first record to be copied or displayed. For example, if FROM-'15' is specified, the first key beginning with 15 or larger is the key of the first record to be copied or displayed. If only one record is to be copied or displayed, the FROM and TO parameters must specify the same key.

TO-'key': Specifies the key (or the beginning characters of the key) of the last record to be copied or displayed. If packed keys are used, up to 57 numeric characters can be specified. If none of the keys in the file begin with the specified characters, the record with the next lower key is the last record to be copied or displayed. For example, if TO-'34' is specified, the last key beginning with 34 (or if no keys begin with 34, the last key that begins with a value smaller than 34) is the key of the last record to be copied or displayed.

If the TO parameter is not specified, $COPY uses the last key in the index as the TO key. If only one record is to be copied or displayed, the FROM and TO parameters must specify the same key.

CAUTION

If packed keys are used, the number of characters specified in the FROM parameter must be the same as the number of characters specified in the TO parameter. To ensure that all desired records are selected, you should specify all characters, including leading zeros, in the packed key.

RECORD: Specifies that a portion of the file is to be copied or displayed. When RECORD is specified, the FROM parameter and the TO parameter (if TO is used) must specify relative record numbers.

FROM-number: Specifies the relative record number of the first record to be copied or displayed. For example, if the first record to be copied is the fifth record in the file, FROM-5 should be specified. If only one record is to be copied or displayed, the FROM and TO parameters must specify the same relative record number.

TO-number: Specifies the relative number of the last record to be copied or displayed. For example, if the last record to be copied is the 15th record in the file, TO-15 should be specified. If only one record is to be copied or displayed, the FROM and TO parameters must specify the same relative record numbers.

For the COPYALL statement:

TO: TO-F1 specifies that the files are to be copied from diskette(s) to the disk. If a name other than #SAVE was assigned to the set of saved files (that is, if a label other than #SAVE was specifically assigned on the COPYO FILE statement when the files were copied to diskette), the name associated with the set of saved files must be specified in the LABEL parameter of the COPYIN FILE statement.

TO-I1 specifies that disk files are to be copied to diskette(s). If a label is not specified on the COPYO FILE statement, the name associated with the entire set of saved files is #SAVE.

GROUP: GROUP-ALL specifies that all files (except #LIBRARY, user libraries, and the spool file) are to be copied.

GROUP-file group specifies that all members of the specified file group are to be copied. For example, GROUP-A1 specifies that all files with labels beginning with A1. are to be copied.

If GROUP is not specified, all files that are not members of a file group (except #LIBRARY, user libraries, and the spool file) are to be copied.

Control Statement Sequence

```
// LOAD $COPY
// FILE NAME–COPYIN [,UNIT–F1] ,LABEL–file label
```

$$\left[,\text{RETAIN–}\left\{\begin{matrix}\text{S}\\ \text{J}\\ \underline{\text{T}}\\ \text{P}\end{matrix}\right\}\right]$$

or

```
// FILE NAME–COPYIN, UNIT–I1, LABEL–file label
```

$$\left[,\text{RETAIN–}\left\{\begin{matrix}\text{retention days}\\ \underline{1}\end{matrix}\right\}\right]$$

$$\left[,\text{LOCATION–}\left\{\begin{matrix}\underline{\text{S1}}\\ \text{S2}\\ \text{S3}\\ \text{M1.nn}\\ \text{M2.nn}\end{matrix}\right\}\right]\left[,\text{AUTO–}\left\{\begin{matrix}\text{NO}\\ \underline{\text{YES}}\end{matrix}\right\}\right]$$

```
// FILE NAME–COPYO, UNIT–I1, LABEL–file label
```

$$\left[,\text{RETAIN–}\left\{\begin{matrix}\text{retention days}\\ \underline{1}\end{matrix}\right\}\right],\text{PACK–vol-id}$$

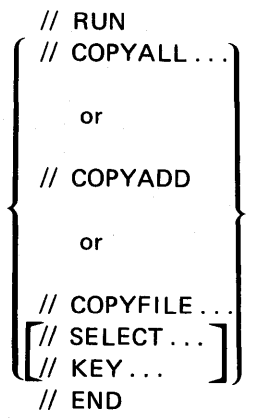
$$\left[,\text{LOCATION–}\left\{\begin{matrix}\underline{\text{S1}}\\ \text{S2}\\ \text{S3}\\ \text{M1.nn}\\ \text{M2.nn}\end{matrix}\right\}\right]\left[,\text{AUTO–}\left\{\begin{matrix}\text{NO}\\ \underline{\text{YES}}\end{matrix}\right\}\right]$$

or

```
// FILE NAME–COPYO [,UNIT–F1] ,LABEL–file label
```

$$\left[\begin{matrix},\text{RECORDS–number}\\ ,\text{BLOCKS–number}\end{matrix}\right]\left[,\text{RETAIN–}\left\{\begin{matrix}\text{S}\\ \text{J}\\ \underline{\text{T}}\\ \text{P}\end{matrix}\right\}\right]$$

$$\left[,\text{LOCATION–}\left\{\begin{matrix}\text{block number}\\ \text{A1}\\ \text{A2}\end{matrix}\right\}\right]\left[,\text{DFILE–}\left\{\begin{matrix}\text{YES}\\ \underline{\text{NO}}\end{matrix}\right\}\right]$$



The COPYIN FILE statement defines the file to be copied, reorganized, or displayed, and the COPYO FILE statement defines the output file created by $COPY.

For information about FILE statement parameters, see the description of the FILE OCL statements in Chapter 1.

Special consideration must be given to specifying the RETAIN parameter on the FILE statements for $COPY. Following is a summary of how the RETAIN parameter affects the retention codes of the file used by $COPY.

$COPY File Retention Summary

The effect that the RETAIN parameter retention code (S, J, T, or P) has on the retention of a disk file for the $COPY program depends on whether:

- The file is an input file or an output file.
- The file is on disk or diskette(s).

Each file that exists on disk, except for new scratch files (retention type S) and job files (retention type J), has a record in the VTOC of system information describing the file, such as file label, file creation date, file organization, and retention code. This record is called a *VTOC format 1*. A disk VTOC format 1 has a retention code of either P (permanent) or T (temporary).

A file being processed also has an *active format 1* in main storage. The active format 1 is created by the FILE statement. An active format 1 has a retention code of S, J, T, or P.

For a file existing on disk (input file), the active format 1 is the same as the existing VTOC format 1; therefore, the retention code is the same unless it is modified by the RETAIN parameter in the FILE statement for the input file (COPYIN). For a file being created on disk (output file), the retention code is specified in the RETAIN parameter (or defaults to T if that parameter is not used) for the output file (COPYO).

An active format 1 with a retention code of P or T is substituted for the VTOC format 1 at the end of the job. An active format 1 with a retention code of S is deleted at the end of the job step. If an active format 1 with a retention code of S identifies a file in the VTOC with a retention code of T, the VTOC format 1 is also deleted at the end of the job step. A file with a retention code of J is deleted at the end of the job.

Note: If the retention code for the disk input file and the retention code for the disk output file are S, neither file will exist at the end of the job.

In the following paragraphs, which summarize the results of the RETAIN parameter on the VTOC format 1 retention code, the FILE statement named COPYIN identifies the input file (file being copied) and COPYO identifies the output file (file being created).

Existing Disk File

The RETAIN parameter is optional in the FILE statement that describes the input file, COPYIN. However, if you are accessing a temporary file and want to delete that file at the end of the job, include the RETAIN parameter with a retention code of S. The following summary shows the effect of the RETAIN parameter retention code on the final VTOC format 1 for a disk input file:

VTOC Format 1 Retention Code for Disk Input File	RETAIN Parameter on COPYIN FILE Statement	Final Active Format 1 for the Disk Input File
P	P	P
P	T	P
P	S	P
T	P	T
T	T	T
T	S	S

Disk to Disk

If the input file, identified by the FILE statement named COPYIN, resides on disk, and the output file, identified by the FILE statement named COPYO, will reside on disk, the VTOC format 1 for the output file becomes whatever is specified in the RETAIN parameter, as shown in the following summary:

Active Format 1 Retention Code for Disk Input File	RETAIN Parameter on COPYO FILE Statement	Final Active Format 1 Retention Code For Output File
P	P	P
P	T	T
P	S	S
P	J	J
T	P	P
T	T	T
T	S	S
T	J	J
S	P	P
S	T	T
S	S	S
S	J	J

Note: The retention code of the input file does not change unless you also code the RETAIN parameter for COPYIN.

CAUTION
You will have neither an input file nor an output file at the end of job step if both of the following are true:

- The input and output file labels and allocations are the same.
- RETAIN-S is specified for the COPYIN FILE OCL statement.

This is because the active format 1 retention code for the output file is S. To prevent both files from being deleted, specify DISP-NEW on the COPYO FILE OCL statement.

Diskette to Disk

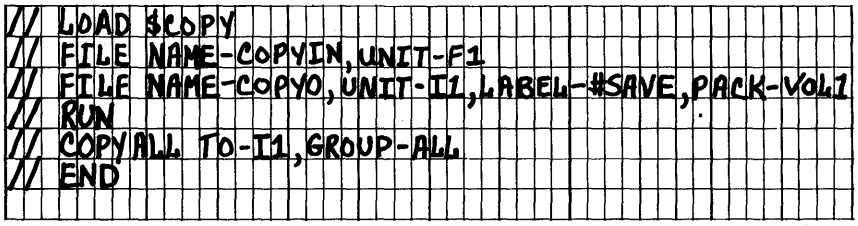
If the input file, identified by the FILE statement named COPYIN, resides on diskette, the retention code for this file is the final retention code of the disk file from which it was created. The output file, identified by the FILE statement named COPYO, will be recreated on disk. The following summary shows the effect of the RETAIN parameter on the VTOC format 1 retention code for the output file:

VTOC Format 1 Retention Code for File on Diskette	RETAIN Parameter on COPYO FILE Statement	Final VTOC Format 1 Retention Code for Disk Output File
P	P	P
P	T	P
P	S	P
P	J	P
T	P	P
T	T	T
T	S	T
T	J	T
S	P	P
S	T	T
S	S	T
S	J	T
J	P	P
J	T	T
J	S	T
J	J	T

Examples

Example A

Copy all disk files to diskette(s):



Example B

Copy a diskette file (JOE) to a disk file (JOEF):

```
// LOAD $COPY
// FILE NAME-COPYIN,UNIT-I1,LABEL-JOE
// FILE NAME-COPYO,UNIT-F1,LABEL-JOEF,
//      BLOCKS-100,RETAIN-P
// RUN
// COPYFILE OUTPUT-DISK
// END
```

Example C

Print from the diskette file JON all records with keys from ADAMS to BAKER:

```
// LOAD $COPY
// FILE NAME-COPYIN,UNIT-I1,LABEL-JON
// RUN
// COPYFILE OUTPUT-PRINT
// SELECT KEY,FROM-'ADAMS',TO-'BAKER'
// END
```

Example D

Copy back to the disk the entire set of files previously copied from the disk to diskette(s):

```
// LOAD $COPY
// FILE NAME-COPYIN,UNIT-I1,LABEL-#SAVE
// FILE NAME-COPYO
// RUN
// COPYALL TO-F1
// END
```

$DDST–Key Sort Utility Program

Function

The $DDST utility program sorts the index keys for a specified disk file. For information about sorting index keys, see *Key Sorting for Indexed Files* in the *Concepts and Design Guide*.

$DDST is executed when the KEYSORT procedure is run.

Utility Control Statement Format

// KEYSORT LABEL–file label [,DATE–{mmddyy / ddmmyy / yymmdd}] [,RETAIN–{P / T / J}]

Parameters

LABEL: The label of the file to be sorted.

DATE: The creation date of the file to be sorted. The format of the specified date must be the same as the format of the session date.

RETAIN: The retention of the file. If no retention is specified, J is used first, then T.

P, T, or J: The retention of the file. By specifying P, T, or J, the user is defining a specific file and no further searching is done. Without retention specified, the order of search is J first, then T, followed by P; the keysort is done on the first file found with name or date that match. The KEYSORT procedure then goes to end of job.

Note: To keysort a T file that has the same label as a J file, RETAIN-T must be specified.

Control Statement Sequence

```
// LOAD $DDST
// FILE NAME–KEYSORT ,...
// RUN
// KEYSORT LABEL–file label [,DATE–{mmddyy / ddmmyy / yymmdd}] [,RETAIN–{P / T / J}]
// END
```

Note: If a FILE OCL statement is used to identify the file to be sorted, the KEYSORT control statement cannot be used.

Example

Sort the index keys in a file called PAYRCD with a creation date of 11-11-79:

```
// LOAD $DDST
// RUN
// KEYSORT LABEL-PAYRCD,DATE-111179
// END
```

$DELET–File Delete Utility Program

Function

The $DELET utility program frees the space occupied by existing temporary or permanent files for use by new files. The space is freed in the following ways:

SCRATCH — For diskette file(s), changes the expiration date of the diskette file(s) to the current job date. For disk file(s), removes the VTOC entry.

REMOVE — Removes the VTOC entry with the option of erasing (overwriting with zeros) the contents of the named file(s) on the disk or diskette.

If you want to delete more than one file, additional control statements must be used. The $DELET utility will process additional control statements under the following conditions:

- The delete operation for a disk file or library (UNIT-F1) completed successfully.
- The delete operation for a single diskette file (UNIT-I1) completed successfully.
- If the System/34 has a diskette magazine drive, the delete operation for multiple diskette files (LABEL-ALL, UNIT-I1) completed successfully.

The END statement must follow the last SCRATCH or REMOVE statement.

Permanent disk files can be deleted only by $DELET. The system library (#LIBRARY), checkpoint active files, or checkpoint active libraries cannot be deleted.

$DELET is executed when the DELETE procedure is run.

Utility Control Statement Formats

To scratch the VTOC entry for a file on a diskette:

```
// SCRATCH UNIT–I1 , LABEL–file label [, PACK–vol-id]

      [                 {S1    }]
      [                 {S2    }]
      [, LOCATION–      {S3    }]
      [                 {M1.nn }]
      [                 {M2.nn }]

// END
```

To scratch the VTOC entry for a file on the disk:

```
// SCRATCH UNIT–F1,LABEL-file label
// END
```

To scratch the VTOC entry for a library on the disk:

```
// SCRATCH UNIT–F1,LABEL–user library name,USERLIBS–YES
// END
```

To scratch the VTOC entry for a diskette file with a specified creation date:

```
// SCRATCH UNIT-I1 , LABEL-file label,DATE-{mmddyy|ddmmyy|yymmdd}
    [,PACK-vol-id ]
    [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}]
// END
```

To scratch the VTOC entry for a disk file with a specified creation date:

```
// SCRATCH UNIT-F1 , LABEL-file label , DATE-{mmddyy|ddmmyy|yymmdd}
    [,USERLIBS-NO]
// END
```

To scratch the VTOC entries for all files on a diskette:

```
// SCRATCH UNIT-I1,LABEL-ALL , PACK-vol-id
    [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}] [,AUTO-{NO|YES}]
// END
```

To scratch the VTOC entries for all files on the disk:

```
// SCRATCH UNIT-F1, LABEL-ALL [,USERLIBS-{YES|NO}] ,GROUP-ALL
// END
```

Note: If UNIT-F1, LABEL-ALL, and GROUP-ALL are all specified, $DELET must be requested from the display station assigned as the system console, and no other job can be running.

To scratch the VTOC entries for all disk files that are members of a specified file group or for all files and user libraries that are members of a specified group.

```
// SCRATCH UNIT–F1, LABEL–ALL [,USERLIBS–{YES}]
                                        {NO }
                  ,GROUP–file group
// END
```

To scratch the VTOC entries for all files that are not members of a file group:

```
// SCRATCH UNIT–F1, LABEL–ALL [,USERLIBS–{YES}]
                                        {NO }
// END
```

To remove the VTOC entry for a file on a diskette:

```
// REMOVE UNIT–I1 , LABEL–file label [,PACK–vol-id]

        [                {S1   }]
        [                {S2   }]
        [,LOCATION–      {S3   }]
        [                {M1.nn}]
        [                {M2.nn}]

// END
```

To remove the VTOC entry for a library on the disk:

```
// REMOVE UNIT–F1,LABEL-user library name,USERLIBS–YES
// END
```

To remove the VTOC entry for a diskette file with a specified creation date:

```
// REMOVE UNIT-I1,LABEL-file label,DATE-{mmddyy|ddmmyy|yymmdd}
    [,PACK-vol-id]

    [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}]

// END
```

To remove the VTOC entry for a disk file with a specified creation date:

```
// REMOVE UNIT-F1,LABEL-file label,DATE-{mmddyy|ddmmyy|yymmdd}
    [,USERLIBS-NO]
// END
```

To remove the VTOC entries for all files on a diskette:

```
// REMOVE UNIT-I1,LABEL-ALL,PACK-vol-id

    [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}] [,AUTO-{NO|YES}]

// END
```

To remove the VTOC entries for all disk files:

```
// REMOVE UNIT-F1,LABEL-ALL [,USERLIBS-{YES|NO}] ,GROUP-ALL
// END
```

Note: If UNIT-F1, LABEL-ALL, and GROUP-ALL are all specified, $DELET must be requested from the display station assigned as the system console, and no other job can be running.

To remove the VTOC entries for all files that are members of a specified file group or for all files and user libraries that are members of a specified group:

```
// REMOVE UNIT–F1, LABEL–ALL [, USERLIBS– {YES} ]
                                          {NO  }

          , GROUP–file group
// END
```

To remove the VTOC entries for all files that are not members of a file group:

```
// REMOVE UNIT–F1, LABEL–ALL [, USERLIBS– {YES} ]
                                          {NO  }
// END
```

To remove the VTOC entry for a diskette file and erase the contents of that file:

```
// REMOVE UNIT–I1 , LABEL–file label , DATA–YES [, PACK–vol-id]

      [                {S1   }]
      [, LOCATION– {S2   }]
      [                {S3   }]
      [                {M1.nn}]
      [                {M2.nn}]

// END
```

To remove the VTOC entry for a user library or disk file and erase the contents of that file:

```
// REMOVE UNIT–F1 ,LABEL

// REMOVE UNIT–F1 , LABEL–file or user library label , DATA–YES
      [, USERLIBS–{YES} ]
                  {NO  }
// END
```

Parameters

For the SCRATCH and REMOVE statements:

UNIT: Specifies the location of the file(s) to be deleted. F1 specifies disk; I1 specifies diskette.

LABEL: If a file or user library label is specified, that file is deleted. If ALL is specified, the GROUP parameter determines which files are deleted. (User libraries are deleted only if USERLIBS-YES is specified.) For diskettes, the files on more than one diskette can be deleted with one $DELET request if the diskettes have the same volume ID.

DATE: The creation date of the file. For a disk file, the format of the specified date must be the same as the format of the session date. For a diskette file, the format of the specified date must be the same as the format of the file creation date. If USERLIBS-YES is specified, DATE cannot be specified.

Notes:

1. If no date is specified and more than one file with the specified label exists on the disk, the operator can delete all files with that label or cancel the job.
2. If no date is specified and more than one file with the specified label exists on a diskette, only the first file with the specified label is deleted.

DATA: DATA-YES specifies that the contents of the deleted file(s) are overwritten with binary zeros. DATA-NO specifies that the contents of the file(s) are unchanged. If the DATA parameter is not specified, DATA-NO is assumed.

PACK: Vol-id of the diskette. The PACK parameter is required when LABEL-ALL and UNIT-I1 are both specified in the REMOVE or SCRATCH statements.

USERLIBS: USERLIBS-YES specifies that user libraries are to be deleted. USERLIBS-NO specifies that user libraries are not to be deleted. If the USERLIBS parameter is not specified, USERLIBS-NO is assumed. If USERLIBS-YES is specified, the DATE parameter cannot be specified.

Note: A library cannot be deleted if it is being used by a display station. Once a user library is specified as active during the sign-on procedure or by a LIBRARY OCL statement entered from the keyboard, that library remains allocated to the display station at least until a different active library or NAME-0 is specified. If a different active library or NAME-0 is specified while the display station is still using one or more members from the previous active library (for example, using a menu from that library), the previous active library remains allocated to the display station until the display station is no longer using members from that library.

GROUP: GROUP-ALL specifies that all files are to be deleted. GROUP-ALL is valid only if $DELET is run from the system console and only if no other jobs are running.

GROUP-file group specifies that files that are members of the specified file group are to be deleted. For example, GROUP-B1 indicates that all files with labels beginning with B1. should be deleted. The USERLIBS parameter specifies whether user libraries with the same group name are also to be deleted.

If the GROUP parameter is not specified, all files that are not members of a file group are to be deleted.

LOCATION: Specifies the location of the diskette from which the file or files are to be deleted:

- LOCATION-S1, LOCATION-S2, or LOCATION-S3 identifies the individual diskette slot that contains the diskette.

  Note: If the file is a multivolume file, the $DELET utility can use all three individual slots. The volumes to be deleted should be placed in order in the slots. The first volume to be deleted must be in the slot specified by the LOCATION parameter. Each time a volume in slot S3 is deleted, a message is displayed if more volumes remain to be deleted. The system operator can then do one of the following:
  - Insert the next volumes beginning in slot S1 and select the option that causes the $DELET utility to resume deleting at slot S1
  - Insert the next diskette in slot S3 and select the option that causes the $DELET utility to resume deleting at slot S1

- LOCATION-M1.nn or LOCATION-M2.nn identifies the magazine location that contains the diskette.

  Note: If the file is a multivolume file, the volumes to be deleted must be placed in order in the magazine. The first volume to be deleted must be in the location specified by the LOCATION parameter. When the last volume in a magazine is deleted, a message is displayed if more volumes remain to be deleted. The system operator can then do one of the following:
  - Insert the next magazine into the current magazine slot and select the option that causes the $DELET utility to resume deleting at the first location in the current magazine slot
  - Ensure that the next magazine is in the other magazine slot and select the option that causes the $DELET utility to resume deleting at the first location in the other magazine slot

AUTO: Controls diskette processing in the magazine drive. The AUTO parameter is valid only if LABEL-ALL is also specified.

AUTO-NO specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, $DELET deletes information only from the diskette in that slot.
- If M1.nn or M2.nn is specified in the LOCATION parameter, the $DELET utility deletes information only from the diskette in the specified magazine location.
- After the information has been deleted from the diskette, message SYS–1624 is displayed to allow another diskette with the same volume ID to be inserted in the specified location.

If the AUTO parameter is not specified, AUTO-NO is assumed.

AUTO-YES specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, the $DELET utility deletes information from the diskettes in the individual slots, beginning with the slot specified by the LOCATION parameter and continuing through slot S3. After the information has been deleted from the diskette in slot S3, the $DELET utility ends.
- If M1.nn or M2.nn is specified in the LOCATION parameter, the $DELET utility deletes information from the diskettes in the specified magazine, beginning with the diskette in the location specified by the LOCATION parameter. After the information has been deleted from a diskette, the $DELET utility continues on to the next slot and checks the vol-id of that diskette. If the vol-id matches the pack parameter, the $DELET utility continues deleting information. However, if vol-id does not match the pack parameter, the SSP issues a halt to allow either a cancel of the $DELET utility or the insertion of a diskette with the correct vol-id. After the information has been deleted from the diskette in the last location in the magazine, message SYS–1637 is displayed to allow the delete operation to continue with the first location of the same magazine slot.

Control Statement Sequence

```
// LOAD $DELET
// RUN
// SCRATCH UNIT-{F1 | I1} ,LABEL-{file label | ALL} [,DATE-{mmddyy | ddmmyy | yymmdd}]
   [,PACK-vol-id] [,USERLIBS-{YES | NO}] [,GROUP-{ALL | file group}]
   [,LOCATION-{S1 | S2 | S3 | M1.nn | M2.nn}] [,AUTO-{NO | YES}]
                              and/or
// REMOVE UNIT-{F1 | I1} ,LABEL-{file label | ALL}[,DATE-{mmddyy | ddmmyy | yymmdd}]
   [,DATA-{YES | NO}] [,PACK-vol-id] [,USERLIBS-{YES | NO}]
   [,GROUP-{ALL | file group}] [,LOCATION-{S1 | S2 | S3 | M1.nn | M2.nn}]
   [,AUTO-{NO | YES}]
// END
```

Examples

Example A

Remove from the disk the VTOC entry for a file labeled JOE (created October 14, 1977):

```
// LOAD $DELET
// RUN
// SCRATCH UNIT-F1,LABEL-JOE,DATE-10/14/77
// END
```

Example B

Remove and erase from the disk all files named JON:

```
// LOAD $DELET
// RUN
// REMOVE UNIT-F1,LABEL-JON,DATA-YES
// END
```

$DUPRD–Diskette Copy Utility Program

Function

The $DUPRD utility program copies a single file on a diskette or all files on a diskette to another diskette to provide a duplicate of the file(s). For systems without a diskette magazine drive, the system reads the data to be copied and then instructs the system operator to insert the receiving diskette into the diskette drive. For systems with a diskette magazine drive, the diskette being copied is placed in slot S1, and the diskette to receive the copy is placed in slot S2. Also, for systems with a diskette magazine drive, COPYI1 can be used to copy the diskettes in magazine slot M1 onto the diskettes in magazine slot M2.

A work space large enough to contain the file(s) to be copied must be available on the disk. When an entire diskette is copied, unused space on the input diskette can be gathered into a single free space on the output diskette. The output diskette must be in the same format as the diskette being copied. The creation date and expiration date of each file created by $DUPRD are the same as the creation date and expiration date of the copied file. See Appendix C for information about diskette formats.

$DUPRD will copy only files that meet the following requirements:

- The file was created on System/34, or the operating system contains blanks.
- The block length is the same as the sector length.
- The record attributes are specified as unblocked and unspanned or as blocked and unspanned.

See Appendix C for information about diskette record attributes.

Diskettes with important files are the diskettes normally copied. Because diskettes can develop surface irregularities as they undergo continued use, it is a good idea to copy your important files soon after they are created.

If $DUPRD encounters a read error on the input diskette, $DUPRD displays the bad sector. The operator can then choose to correct the information in the sector or to have it copied to the output disk as it is. (If the error was in a load member that was copied from a library or if the error was in file control information used by the system, the file will not be usable even if the operator chooses to bypass the error.) The operator can use the following function and command keys:

Key	Function
Roll Up	Scans right
Roll Down	Scans left
Enter/Rec Adv	Updates the sector buffer by adding the changes keyed by the operator (The Enter/Rec Adv key has no effect if a sector other than the bad sector is displayed.)
Cmd key 1	Displays the next sector
Cmd key 2	Displays the preceding sector
Cmd key 3	If pressed while the bad sector is displayed, causes the sector buffer to be copied to the output diskette If pressed when a sector other than the bad sector is displayed, causes the bad sector to be displayed

Utility Control Statement Formats

$DUPRD is executed when the COPYI1 procedure is run.

```
// COPYI1 NAME-{ALL|label} ,PACK-vol-id [,DELETE-{YES|NO}]
      [,PRESERVE-{YES|NO}]
      [,COPIES-{nn|1}] [,LOCATION-{M1|M1.01|S1}]
// END
```

Parameters

NAME: NAME-ALL specifies that all files on an individual diskette are to be copied to another diskette, or, if LOCATION-M1.01 is specified, NAME-ALL specifies that the contents of the diskettes in magazine slot M1 are to be copied onto the diskettes in slotM2.

NAME-label specifies the label of a single file to be copied. NAME-label is invalid with LOCATION-M1 or LOCATION-M1.01.

PACK: Vol-id of the output diskette.

DELETE: The DELETE parameter is valid only with NAME-ALL. DELETE-YES specifies that expired files on the input diskette are not to be copied.

DELETE-NO specifies that expired files are to be copied. If NAME-ALL is specified and the DELETE parameter is not specified, DELETE-NO is assumed.

PRESERVE: PRSERVE-YES specifies that the entire file extent (the space from the beginning of the file to the end of the file) is to be copied even if the file is not full.

PRESERVE-NO specifies that only sectors containing data are to be copied. An end of file indicator is placed immediately after the last sector copied. If the PRESERVE parameter is not specified, PRESERVE-NO is assumed.

COPIES: COPIES-nn specifies the number of copies to be made. nn can be any number from 1 through 99. If the COPIES parameter is not specified, COPIES-1 is assumed.

LOCATION: LOCATION-M1.01 specifies that the diskettes in magazine slot M1 are to be copied onto the diskettes in slot M2. (Specifying LOCATION-M1 is the same as specifying LOCATION-M1.01.)

LOCATION-S1 specifies that data from the diskette in slot S1 is to be copied onto the diskette in slot S2. If the LOCATION parameter is not specified, LOCATION S1 is assumed.

For systems without a diskette magazine drive, the LOCATION parameter is ignored.

Control Statement Sequence

```
// LOAD $DUPRD
// FILE NAME-COPYI1,UNIT-I1, . . .
// RUN
// COPYI1 . . .
// END
```

Examples

Example A

Copy all files on a diskette to another diskette with a volume ID of 123456:

```
// LOAD $DUPRD
// FILE NAME-COPYI1,UNIT-I1
// RUN
// COPYI1 NAME-ALL,PACK-123456
// END
```

Example B

Copy a file on a diskette (with a file label of JIM and a creation date of 01-02-77) to another diskette (with a volume ID of 345678):

```
// LOAD $DUPRD
// FILE NAME-COPYI1,UNIT-I1,DATE-010277
// RUN
// COPYI1 NAME-JIM,PACK-345678
// END
```

$FBLD–File Build Utility Program

Function

The $FBLD utility program creates a disk file that contains no data records. That file can be referenced as an existing file by subsequent jobs and job steps.

$FBLD is executed when the BLDFILE procedure is run.

Utility Control Statement Format

```
// FILE LABEL–file label, ATTRIB–{ SEQUENTL | (S) | INDEXED | (I) | DIRECT | (D) | IFILE | (A) } , RECL–recl ,
    { BLOCKS–value | RECORDS–value } , LOCATION–{ A1 | A2 | location } [ , RETAIN–{ S | J | T | P } ] ,
    [POSITION–key position , LENGTH–key length] [ , DFILE–{ YES | NO } ]
// END
```

Note: Any number of FILE utility control statements can be specified, as long as the number of FILE statements does not exceed the number of files that can be allocated for the job step.

Parameters

LABEL: The label of the file to be created.

ATTRIB: Specifies the organization of the created file:

Value	File Organization
SEQUENTL or S	Sequential
INDEXED or I	Indexed
DIRECT or D	Direct
IFILE or A	Immediate access

RECL: The record length in bytes. Can be any decimal number from 1 through 4096.

BLOCKS: The number of blocks to be allocated for the file. Can be any decimal number from 1 through 8388607, but cannot exceed the amount of disk space available for user files.

RECORDS: The number of records to be allocated for the file. Can be any decimal number from 1 through 8388607, but cannot exceed the amount of disk space available for user files.

LOCATION: Specifies the number of the block where the file is to begin, or indicates a preferred disk placement for the new file.

The following table gives the beginning block number location for disks 2 through 4:

Disk	Beginning Block Number Location
Disk 2	25203
Disk 3	50406
Disk 4	75609

If a preferred logical disk (A1 indicates the first disk and A2 indicates disks 2, 3 and 4) is specified, the system attempts to place the new file on the specified logical disk. If sufficient space does not exist on the preferred logical disk, the system places the file on the other logical disk. No message is issued to the operator.

RETAIN: Specifies the retention type of the new file.

Value	Meaning
S	Scratch file
J	Job file
T	Temporary file
P	Permanent file

For information about file retention types, see the descriptions of the FILE OCL statements in Chapter 1.

Note: If RETAIN-S is specified, the file no longer exists as soon as the step terminates.

POSITION: The starting byte of the key area within the record for an indexed file. The specified value can be from 1 through 999 and cannot be greater than the record length. The sum of the key position and the key length must not exceed the length of the record plus 1. If a key position is specified, the key length must also be specified.

LENGTH: The length of the key area within the record for an indexed file. The specified value can be from 1 through 29 and cannot be greater than the record length. A key length must be specified if a key position is specified. The sum of the key position and the key length must not exceed the length of the record plus 1.

Note: The POSITION and LENGTH parameters must be specified when creating an indexed file. The POSITION and LENGTH parameters are ignored when creating a file other than an indexed file.

DFILE: DFILE-YES specifies that a delete-capable file should be created. If you specify DFILE-NO or omit this parameter, the SSP does not create the file as delete capable. The DFILE parameter is valid only on a system configured with extended disk data management.

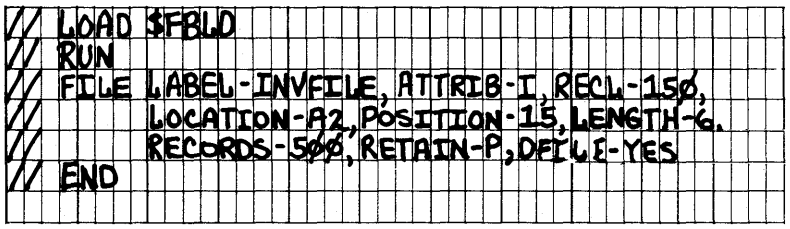
Note: Deleting a record within a delete-capable file does not result in the physical compression of the file. Instead, the record is marked as a deleted record and is inaccessible to all users of the file.

Control Statement Sequence

```
// LOAD $FBLD
// RUN
// FILE . . .
// FILE . . .
   •
   •
   •
// FILE . . .
// END
```

Example

Create a permanent, delete-capable, indexed file that is labeled INVFILE. Records in the file are to be 150 bytes long, and the file must be large enough to contain 500 records. The record keys are 6 bytes long and begin at byte 15 in the record. If possible, the file should be placed on the second disk.



$FREE–Disk Reorganization Utility Program

Function

The $FREE utility program causes all free space on the disk, except free space within files and libraries, to be accumulated into a single area. The location of the area of free space depends upon the disk configuration (single or multiple) and the parameters specified in the COMPRESS utility control statement.

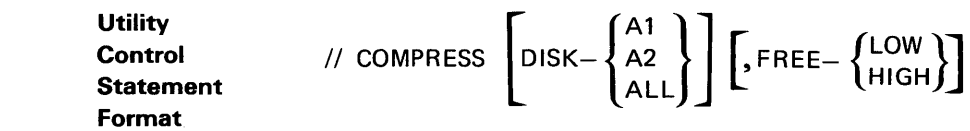
$FREE can be run only from the display station assigned as the system console, and cannot be run (1) while any other jobs are being run, (2) while a user is signed on at any other display station, (3) while remote work stations are varied online, (4) while SSP-ICF subsystems are enabled, (5) while a communication line is enabled, or (6) while extended trace is active.

If a system failure occurs during the running of $FREE, $FREE must be run again to ensure the integrity of data on the disk. If IPL is required after the system failure, $FREE is automatically run as part of the IPL procedure. If IPL is not performed, run the $LABEL utility program to display the disk VTOC. If $FREE must be run, the following message appears as part of the information displayed by $LABEL:

> $FREE MUST BE RUN BEFORE INFORMATION CAN BE OBTAINED FROM THIS FILE.

$FREE must then be the next program run. *No other program except $LABEL should be run until $FREE completes.*

Utility Control Statement Format



Parameters

DISK: Specifies the disk or disks upon which files are to be compressed:

- DISK-A1 specifies that for a multiple-disk configuration, only free space on the first disk is accumulated. DISK-A1 can also be specified for a single-disk configuration.
- DISK-A2 specifies that for a multiple-disk configuration, only free space on disk A2 is accumulated. DISK-A2 is not valid for a single-disk configuration.
- DISK-ALL specifies that for a multiple-disk configuration, the free space on both disks is accumulated at the disk juncture. That is, free space on the first disk is accumulated at the highest block numbers on the disk, and free space on the second disk is accumulated at the lowest block numbers on the disk.

  If DISK-ALL is specified for a single-disk configuration, free space is accumulated at the highest block numbers on the disk.

  If neither the DISK nor the FREE parameters are specified, DISK-ALL is assumed. If the DISK parameter is not specified and the FREE parameter is specified, DISK-A1 is assumed.

FREE: The FREE parameter is valid only if DISK-A1 or DISK-A2 is specified and specifies the location of the accumulated free space:

- FREE-LOW specifies that free space is accumulated at the lowest available block numbers on the disk. For the first disk, the free space immediately follows the system library; for the second disk, the free space begins at the lowest block number.

  If DISK-A2 is specified and the FREE parameter is not specified, FREE-LOW is assumed.

- FREE-HIGH specifies that free space is accumulated at the highest available block numbers on the disk.

  If DISK-A1 is specified and the FREE parameter is not specified, FREE-HIGH is assumed.

Control Statement Sequence

```
// LOAD $FREE
// RUN
[// COMPRESS . . .]
// END
```

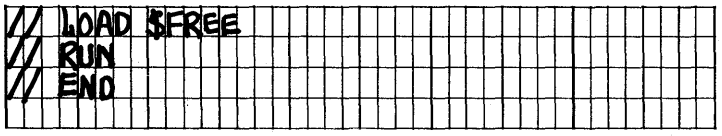
Examples

Example A

For a single-disk configuration, accumulate the free space immediately following the library:

Example B

For a multiple-disk configuration, accumulate the free space on both disks at the disk juncture:



Note: A COMPRESS utility control statement is not required because DISK-ALL is assumed.

$HELP–Help Utility Program

Function

The $HELP utility program is an optional utility program that:

- Aids the operator in executing System/34 procedures
- Provides, on the display screen, quick-reference information for many System/34 functions

You cannot run $HELP by entering the LOAD and RUN OCL statements from the keyboard. $HELP can only be run by the HELP procedure.

$HIST–History File Display Utility Program

Function

The $HIST utility program displays selected entries from the history file or history overflow file on the system list device assigned to the requesting display station.

For information about assigning the system list device, see the description of the SYSLIST procedure in Chapter 2.

Recorded in the history file are:

- All OCL statements, utility control statements, control commands, and procedures executed by the SSP.
- All messages displayed at the display station.
- All operator responses to messages and prompts.
- The end-of-job (*E) entries for each application job or spool print job executed by the SSP. The *E entries have two classifications: end of application job (indicated by *EJ) and end of spool print job (indicated by *EP).
- The work station being used. If the entry comes from a job on the input job queue, no work station ID is associated with the entry.
- The operator's user ID. If the entry comes from a job on the input job queue, no user ID is associated with the entry.
- The job name, in the form wwhhmmss, where ww is the work station ID and hhmmss is the time the job began running. If the entry is an operator control command, a job name is not associated with the entry.
- The time the entry was made in the history file. (The time is based upon the time specified by the system operator during IPL.)

The *EJ entries contain the following information:

- The job's starting time
- The job's ending time
- The amount of time used to run the job
- The date the job was executed
- The operator's user ID
- The work station ID
- The name of the procedure that began the job
- Whether the job was a multiple requestor terminal task (MRT), a nonrequestor terminal task (NRT), or an input job queue task (JOBQ)

The *EP entries contain the following information:

- The job's starting time
- The job's ending time
- The amount of time used to run the job
- The date the job was executed
- The operator's user ID
- The printer's printer ID
- The name of the procedure that created the print job
- The name of the job (wwhhmmss) that created the print job
- The forms number
- A 2-character job completion code: NC–normal completion; CP–system console or subconsole operator canceled the print job; and SP–system console or subconsole operator stopped the print job

Because the history file is limited in size (the size is specified during system configuration), the number of events that can be reflected in the history file at a particular time varies with the length of the file and the length of the entries in the file.

To avoid losing history file entries after the history file has been filled, you can request during IPL that a history overflow file be used. The overflow file contains from 1 to 8 segments, each segment being the same size as the history file. During IPL, you specify the size (number of segments) of the overflow file. When less than 25 sectors remain in the history file, the system issues an informational message to the system console and copies the contents of the history file into an overflow file segment. The system operator can use the HISTORY procedure to print or display the contents of the overflow file. When all entries in a segment are displayed and reset, the system can again use that segment for a copy of the history file.

If an overflow file is not used, history file entries can be lost, because once the file has been filled, each new entry causes the oldest entry to be dropped from the file. When the file is listed, the oldest entry is displayed or printed first.

If $HIST is requested from a display station other than the system console or if SYSTEM or OVERFLOW is not specified on the DISPLAY utility control statement when $HIST is run from the system console, only the history file entries for the requesting display station are displayed.

Indentation of OCL statements is as follows:

Print Pos 1: Any entries beginning with *E
Print Pos 3: Any entries that were displayed prior to entry into the history file
Print Pos 7: All OCL statements generated via procedures
Print Pos 5: All other entries

$HIST is executed when the HISTORY procedure is run.

Utility Control Statement Format

```
                 [ ALLOCATE,RESET                  ]
                 [ DELETE,RESET                    ]
                 [ [ACTIVE]                        ]
                 [ [ALL    ]                       ]
// DISPLAY       [ [jobname] , [number  ]          ] , [CURRENT] , [EONLY    ] ,
                 [ [NOLIST ]   [RESET   ]          ]   [TOTAL  ]   [TEXTONLY ]
                 [ [VIEWED ]   [NORESET ]          ]                [CONTROLS ]

                 [ SYSTEM- {OVERFLOW} ] , [USER-userid] , [WORKSTN-ws-id]
                 [         {YES     }   ]
                 [         {NO      }   ]
```

Parameters

SYSTEM: If SYSTEM-NO is specified, history file entries for only one user or one display station can be displayed. If SYSTEM-YES is specified, the entire history file can be accessed during one execution of $HIST. SYSTEM-OVERFLOW is specified if overflow files are to be accessed.

ACTIVE: Lists history file entries for all active jobs associated with requesting display station.

Note: If SYSTEM-YES or SYSTEM-OVERFLOW is specified, ACTIVE cannot be specified.

ALL: Lists all types of history file entries, including OCL statements generated by procedures. If the DISPLAY statement is not used or if the DISPLAY statement is used and a first parameter is not specified, only items previously displayed to the operator during execution are listed. Items previously displayed to the operator include OCL statements and messages that were logged and displayed as they were entered or issued.

jobname: If a job name is specified, only history file entries for that job are displayed. If the specified job did not originate from the display station executing $HIST, SYSTEM-YES must be specified.

ALLOCATE: Displays and resets the entries in the overflow file (RESET must be specified with ALLOCATE). The overflow file will then be deleted and reallocated. The length of the new overflow file will be the length requested during IPL. The ALLOCATE parameter is used if you change the size of the overflow file during IPL, but the existing file still contains entries that have not been reset.

DELETE: Displays and resets the entries in the overflow file (RESET must be specified with DELETE). The overflow file will then be deleted. The DELETE parameter is used if, during IPL, you request that the overflow file be deleted, but the existing file still contains entries that have not been reset.

VIEWED: Lists only history file entries that are identified as having been displayed during execution. If a parameter is not specified, VIEWED is assumed.

RESET: If SYSTEM-YES is also specified, RESET specifies that any entries displayed cannot be displayed again by $HIST. If SYSTEM-OVERFLOW is also specified, the OVERFLOW file entries cannot be displayed again by the HISTORY procedure. When all entries in an OVERFLOW file segment have been reset, the segment can again be used for a copy of the HISTORY file.

If SYSTEM-YES is not specified, RESET specifies that any entries displayed are unavailable for display by $HIST when it is run from the requesting display station. Those entries can still be displayed if SYSTEM-YES is specified on a DISPLAY utility control statement.

NORESET: Entries that are displayed can be displayed again. If a parameter is not specified, NORESET is assumed.

number: Specifies the number of entries to be displayed. For example, if 23 is specified, the 23 most-recent entries are displayed. The smallest number that can be specified is 1; the largest number is 999. If a number larger than 999 is specified, the entire history file is processed.

CURRENT: Displays only entries that were placed in the history file or overflow file since the last time CURRENT was specified on a DISPLAY utility control statement.

Note: CURRENT is not allowed with ALLOCATE or DELETE.

TOTAL: Displays all specified entries. If a parameter is not specified, TOTAL is assumed.

TEXTONLY: Displays only the text of each entry. Control information (such as user ID, work station ID, job name, and date) is not displayed.

Note: TEXTONLY is ignored and CONTROLS is assumed if SYSTEM-YES or SYSTEM-OVERFLOW is specified. TEXTONLY is not allowed if ALLOCATE or DELETE was specified.

CONTROLS: Displays the control information along with the text for each entry. If a parameter is not specified, CONTROLS is assumed.

EONLY: Display only end-of-job (*E) entries.

USER: If USER is specified, only those entries that have the specified user ID are displayed.

WORKSTN: If WORKSTN is specified, only those entries that have the specified work station ID are displayed.

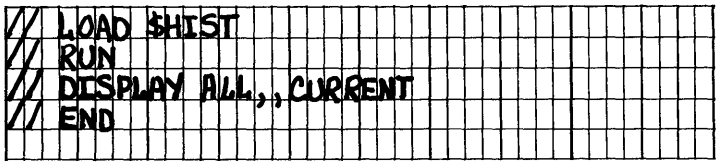
Control Statement Sequence

```
// LOAD $HIST
// RUN
// DISPLAY . . .
// END
```

Examples

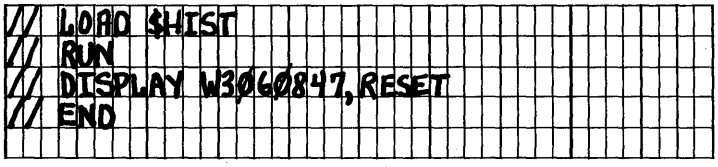
Example A

Display all history file entries for the requesting display station that were added to the history file since the last time CURRENT was specified:



Example B

Display all history file entries for job W3060847:



Note: Because RESET is specified, the entries displayed cannot be displayed again at the requesting display station.

$HSML—History File Scroll Utility Program

Function

The $HSML utility program displays selected entries from the history file on the display screen at the requesting display station.

The $HSML utility program cannot be used to display entries that have been recorded to the history overflow file. To list the entries logged to the history overflow file, you must use the HISTORY procedure or the $HIST utility program.

For information about the entries logged to the history file and the history overflow file, see the description of the $HIST utility program earlier in this chapter.

When the $HSML utility program is run, the most recent entries logged to the history file are displayed first. The following is an example of the initial $HSML display:

```
 HISTORY SCROLL                                WKSTN   USER      JOB NAME    TIME
     // RUN                                     X2    LGA       X2095834  09.59.28
                                                X2    LGA       X2095834  09.59.30
     // DEFINEPN INAME-PHONTEMP,INLIB-#LIBRARY,
     // OUTLIB-#LIBRARY,MODE-NODELAY            X2    LGA       X2095834  09.59.31
     // END                                     X2    LGA       X2095834  09.59.32
  SYS-8621  OPTIONS (   3) PNLM                 X2    LGA       X2095834  09.59.34
                                                X2    LGA       X2095834  09.59.34
  SOURCE MEMBER PHONTEMP NOT FOUND IN SPECIFIED LIB
  3                                             X2    LGA       X2095834  09.59.39
                                                X2    LGA       X2095834  09.59.41
*EJ  09.58.34  09.59.41  00.01.07  07/15/81  LGA        X2  DEFINEPN
  HISTCRT                                       X2    LGA       X2100218  10.02.19
     // LIBRARY NAME-0                          X2    LGA       X2100218  10.02.19
     // MEMBER USER1-##MSG2                     X2    LGA       X2100218  10.02.19
     // * 5039                                  X2    LGA       X2100218  10.02.19
  HISTCRT PROCEDURE EXECUTING                   X2    LGA       X2100218  10.02.20
     // LOAD $HSML                              X2    LGA       X2100218  10.02.20
     // RUN                                     X2    LGA       X2100218  10.02.21
     // DISPLAY SYSTEM-NO                       X2    LGA       X2100218  10.02.22
     // END                                     X2    LGA       X2100218  10.02.22
                  NO MORE ENTRIES IN HISTORY FILE
       CMD7-END OF JOB     CMD8-DISPLAY OLDEST     CMD9-DISPLAY NEWEST
   DIRECTION- B   SCAN DATA-                     WKSTN-     USER-
```

Scan values may be entered at the bottom of the $HSML display. If any values are entered at the bottom of this display, the $HSML utility program searches the history file for entries matching the scan values. A scan direction of forward (F) or backward (B) can be specified; the default value is B. SCAN DATA applies only to the text portion of the history file entries; WKSTN applies only to the control information portion of the entries. The following scan values can be specified:

SCAN DIRECTION	One alphabetic character, either F for a forward scan, or B for a backward scan; the default value is B.
SCAN DATA	A string of 1 through 20 alphameric characters to be searched for within the text portion of the history file entries.
WKSTN	A 2-character, alphameric work station ID.
USER	A 1- to 8-character alphameric user ID can be entered only if SYSTEM-YES was specified. USER does not appear if SYSTEM-NO was specified.

When the Enter/Rec Adv key is pressed after scan values are keyed into the above fields, the $HSML utility program searches the history file for the specified values. If multiple scan values are specified, all specified conditions must be met for the scan to be successful.

If a forward scan is specified, the scan begins with the first history file entry on the currently displayed screen. If the scan is successful, a screen of history file entries is displayed with the first entry containing the specified characters. All entries that satisfy the scan are shown in high intensity and are underlined. To continue the scan operation with the same values, press the Enter/Rec Adv key. The scan continues with the next entry after the last currently displayed entry. If the specified characters are not found in the history file, a message is displayed.

If a backward scan is specified, the scan begins with the last entry on the currently displayed screen and searches backward through the history file. If the scan is successful, a screen of history file entries is displayed with the first entry containing the specified scan characters. To continue the scan in a backward direction, press the Enter/Rec Adv key. The scan continues with the entry previous to the first currently displayed entry. If the specified characters are not found in the history file, a message is displayed.

The $HSML utility program provides the following functions through the use of the command and function keys:

Key	Function
Roll Up	Pages forward through the history file entries. The roll factor is initially set to the number of entries that can be displayed on the screen at one time. When the last, or the most recent, entry in the history file is displayed, a last entry message is displayed. If the Roll Up key is used when the last entry is displayed, the current screen is redisplayed. The roll factor can be changed using Cmd key 3. If SYSTEM-YES was specified in the DISPLAY statement, history file entries are displayed as they are logged to the history file. The newly logged entries are marked with the symbol > when displayed.
Roll Down	Pages backward through the history file entries. The roll factor is initially set to the number of entries that can be displayed on the screen at one time. When the first, or the oldest, entry in the history file is displayed, a first entry message is displayed. If the Roll Down key is used when the first entry is displayed, the current screen is redisplayed. The roll factor can be changed using Cmd key 3.

Key	Function
Cmd key 3	Used to change the roll factor of the Roll Up and the Roll Down keys. The following screen is displayed when Cmd key 3 is pressed:

```
 HISTORY SCROLL                                WKSTN   USER      JOB NAME    TIME
      // RUN                                    X2    LGA       X2095834  09.59.28
                                                X2    LGA       X2095834  09.59.30
      // DEFINEPN INAME-PHONTEMP,INLIB-#LIBRARY,
      // OUTLIB-#LIBRARY,MODE-NODELAY           X2    LGA       X2095834  09.59.31
      // END                                    X2    LGA       X2095834  09.59.32
  SYS-8621  OPTIONS (   3) PNLM                 X2    LGA       X2095834  09.59.34
                                                X2    LGA       X2095834  09.59.34
  SOURCE MEMBER PHONTEMP NOT FOUND IN SPECIFIED LIB
  3                                             X2    LGA       X2095834  09.59.39
                                                X2    LGA       X2095834  09.59.41
*EJ  09.58.34  09.59.41  00.01.07  07/15/81  LGA        X2  DEFINEPN
  HISTCRT                                       X2    LGA       X2100218  10.02.19
      // LIBRARY NAME-0                         X2    LGA       X2100218  10.02.19
      // MEMBER USER1-##MSG2                    X2    LGA       X2100218  10.02.19
      // * 5039                                 X2    LGA       X2100218  10.02.19
  HISTCRT PROCEDURE EXECUTING                   X2    LGA       X2100218  10.02.20
      // LOAD $HSML                             X2    LGA       X2100218  10.02.20
      // RUN                                    X2    LGA       X2100218  10.02.21
      // DISPLAY SYSTEM-NO                      X2    LGA       X2100218  10.02.22
      // END                                    X2    LGA       X2100218  10.02.22
                    NO MORE ENTRIES IN HISTORY FILE
    -ROLL FACTOR      PRESS ENTER, ROLL UP, ROLL DOWN
```

To change the roll factor for future roll up or roll down operations, key in the new number of entries to roll, then press the Enter/Rec Adv key. Valid entries are 1 through 999. If a value is not keyed into this field, the $HSML utility program assumes the number of entries that can be displayed on a full screen.

To change the number of entries to roll for one operation only of the roll keys, key in the new number of entries to roll, and then use the Roll Up or the Roll Down key. This option does not change the roll factor for future roll up or roll down operations. Valid entries are 1 through 999. If a value is not keyed into this field, the $HSML utility program assumes the number of entries that can be displayed on a full screen.

Key	Function
Cmd key 4	Used to specify whether the following control information for each entry to the history file is displayed: • Work station ID • User ID • Job name • Time If the control information is currently displayed, use Cmd key 4 to remove the control information. If the control information is not currently displayed, use Cmd Key 4 to display the control information.
Cmd key 5	Used to specify whether nonviewed entries to the history file are displayed. Viewed entries are those that were displayed prior to entry into the history file. Nonviewed entries include *E entries and all OCL statements generated via procedures. If nonviewed entries are currently displayed, use Cmd key 5 to remove the nonviewed entries. If only viewed entries are currently displayed, use Cmd key 5 to also display nonviewed entries.
Cmd key 6	Used to define what values must be met before an entry to the history file is selected to be displayed. For example, the following screen is displayed if Cmd key 6 is used:

```
                    CHANGE SELECTION CRITERIA                 Optional-*

          Cancels or changes the scroll display selection criteria

Work station ID . . . . . . . . . . . . . . . . . . . . . . . .          *

User ID . . . . . . . . . . . . . . . . . . . . . . . . . . . .          *

Jobname . . . . . . . . . . . . . . . . . . . . . . . . . . . .          *

Start time  . . . . . . . . . . . . . . . . . . . . . . hh.mm.ss         *

Stop time . . . . . . . . . . . . . . . . . . . . . . . hh.mm.ss         *




CMD6-To cancel above selection and return to scroll display.
CMD7-To return to scroll display without any change to selection criteria.
```

Specify the following values by which the system selects the entries to be displayed:

Work station ID: A 2-character, alphameric work station ID.

User ID: A value can be entered in this field only if SYSTEM-YES was specified in the DISPLAY statement. If a user ID is entered in this field, the ID must consist of 1 through 8 alphameric characters.

If SYSTEM-YES was not specified in the DISPLAY statement, the default value for the user ID is the user ID of the requesting display station. In this case, the user ID cannot be changed, and the operator cannot display entries with other user IDs.

Job name: An 8-character, alphameric job name.

Start time: All entries made to the history file including and after the specified time will be displayed. The start time must be entered in the form hh.mm.ss, where hh is hours ($00 \leqslant hh \leqslant 23$), mm is minutes ($00 \leqslant mm \leqslant 59$), and ss is seconds ($00 \leqslant ss \leqslant 59$).

Stop time: All entries made to the history file including and before the specified time will be displayed. The stop time must be entered in the form hh.mm.ss, where hh is hours ($00 \leqslant hh \leqslant 23$), mm is minutes ($00 \leqslant mm \leqslant 59$), and ss is seconds ($00 \leqslant ss \leqslant 59$).

The above fields are initially set to null values, except for user ID. All history file entries matching the values entered above are displayed until you again use Cmd key 6 to change these values. You may enter multiple selection values; in this case, the entries must match all these values in order to be displayed.

Key	Function
Cmd key 7	Ends the $HSML utility program.
Cmd key 8	Displays the oldest, or the first, history file entry.
Cmd key 9	Displays the newest, or the last, history file entry. History file entries will also be displayed as they are logged to the history file.
Cmd key 10	Used to control whether only the history file entries beginning with *EJ (end-of-job) or *EP (end-of-print) are displayed. For information about the *E entries, refer to the description of the $HIST utility program in Chapter 4. If all history file entries, including the *E entries, are currently displayed, use Cmd key 10 to display only the *E entries. If only the *E entries are currently displayed, use Cmd key 10 to display all history file entries, including the *E entries.

$HSML is executed when the HISTCRT procedure is run. The $HSML utility program creates a workfile containing the history file entries to be displayed. The label given to the workfile is HFSWKxx, where xx is the work station ID of the requesting display station. The workfile has a retention of J, which specifies that the workfile will be scratched at end of job.

Utility Control Statement Format

// DISPLAY [SYSTEM- {NO / YES}]

Parameters

SYSTEM: If YES is specified, then all entries in the system history file are displayed. If security is active, SYSTEM-YES can be entered only by an operator with a security classification of system operator or higher. As entries are logged to the history file after the $HSML utility program is run, the new entries are marked with the symbol > when displayed. This allows the operator to view the history file entries as the system processes the corresponding jobs.

If NO is specified, only those entries with the user ID of the requesting display station are displayed. If the SYSTEM parameter is not specified, SYSTEM-NO is assumed.

Control Statement Sequence

```
// LOAD $HSML
// RUN
// DISPLAY . . .
// END
```

Examples

Example A

Display all system history file entries:

```
// LOAD $HSML
// RUN
// DISPLAY SYSTEM-YES
// END
```

Example B

Display only those entries with the user ID of the requesting display station:

```
// LOAD $HSML
// RUN
// DISPLAY
// END
```

$IDSET–Switched Line Remote ID Specification Utility Program

Function

The $IDSET utility program specifies a list of remote IDs for a switched communication line that is using the SSP-ICF BSCEL subsystem. The $IDSET utility is normally called by the DEFINEID procedure.

The first time you run $IDSET, the system file #IBSRID is created, which is 4 sectors long and may contain up to 55 remote IDs. After #IBSRID has been created, $IDSET can be used to update or modify #IBSRID.

The $IDSET utility displays two screens. The initial screen is the command screen that defines the operating instructions for conducting a successful data entry session. The second screen permits the entry of the initial ID list or modification of the existing IDs, including addition, deletion, deactivation, and reactivation. At end of job, all deleted IDs are removed and the file is condensed.

The following editing functions are provided to simplify data entry:

- Page function: The screen may be paged forward or backward.
- Command screen display: At any time during data entry, the command screen can be redisplayed.

The editing functions are specified by the following command keys:

Key	Function
Enter/Rec Adv key	Page forward
Cmd key 1	Redisplay command screen
Cmd key 4	Page backward
Cmd key 9	End job
Cmd key 19	End job without update

The editing functions are valid in either update or display modes. However, if $IDSET is called by the CNFIGICF or ENABLE procedures, then specific modes are assumed as follows:

- CNFIGICF: Update (initialize if it is the first time).
- ENABLE: Update (initialize if it is the first time).

The $IDSET utility must be run from the system console and will reference only the sytem file #IBSRID. If the BSCEL subsystem has already been enabled and you want to modify the list of remote IDs, then perform the following:

1. Disable the BSCEL subsystem via the DISABLE procedure.

2. Modify the file #IBSRID via the DEFINEID procedure.

3. Reenable the subsystem via the ENABLE procedure.

Therefore, if modification of the remote ID is necessary, it is recommended that you perform this operation before the BSCEL subsystem is enabled.

For the BSCEL subsystem to use the remote IDs defined by the DEFINEID procedure, the multiple remote IDs option must be selected when you configure the BSCEL subsystem using the CNFIGICF or ENABLE procedures. For further information, see the *Interactive Communications Feature Reference Manual.*

Utility Control Statement Format

To specify a switched line remote ID:

```
// DEFINEID MODE– { UPDATE  }
                  { DISPLAY }
                  { DELETE  }
```

Parameters

UPDATE: The $IDSET will permit the user to initially specify a list of remote IDs, modify existing IDs, add new IDs, remove old IDs, de-activate IDs without deletion from the file, or reactivate previously de-activated IDs.

DISPLAY: The $IDSET utility displays the existing remote ID list. No modification is permitted. The display option is invalid if no IDs are defined. DISPLAY is the default when you run $IDSET using OCL statements.

DELETE: This option indicates that the system file #IBSRID is to be deleted. The delete option is invalid if no IDs are defined.

Control Statement Sequence

```
// LOAD $IDSET
// RUN
// DEFINEID MODE– { UPDATE  }
                  { DISPLAY }
                  { DELETE  }
// END
```

Example

Delete the system file #IBSRID:

```
// LOAD $IDSET
// RUN
// DEFINEID MODE-DELETE
// END
```

$IEDS–Subsystem Termination Utility Program

Function

The $IEDS utility program terminates an enabled subsystem configuration and can only be run from the system console. If there are active sessions, $IEDS issues a halt with four options. The options and their functions are:

0 Hold the DISABLE procedure; no new sessions are to start. The subsystem will continue the DISABLE procedure when the current sessions terminate.

1 Retry the DISABLE procedure and check again for active sessions.

2 Immediately terminate the active sessions and continue the DISABLE procedure.

3 Terminate the DISABLE procedure and ignore the DISABLE request. The DISABLE procedure goes to end of job, however, the subsystem remains.

$IEDS frees the subsystem queue space in main storage (if any) associated with the subsystem, provided another subsystem of the same type is not using it. If there are no other interactive communications feature subsystems running, $IEDS also frees the main storage space required for SSP-ICF common queue space and SSP-ICF control. If there are no other enabled subsystems of this type, the subsystem task is terminated.

$IEDS is normally called by the DISABLE procedure.

Utility Control Statement Format

To terminate a subsystem:

```
// DISABLE SUBSYS-name
```

Parameters

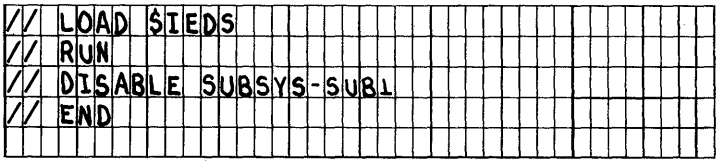
SUBSYS: The member name of the subsystem to disable.

Control Statement Sequence

```
// LOAD $IEDS
// RUN
// DISABLE SUBSYS-name
// END
```

Example

Terminate the subsystem configuration named SUB1:



$IENBL–Subsystem Initialization Utility Program

Function

The $IENBL utility program initializes an interactive communications feature (SSP-ICF) subsystem, and must be run before an SSP-ICF subsystem can be used. This utility program can only be run from the system console. When $IENBL is run with no SSP-ICF tasks currently running, it loads the SSP-ICF control routines and allocates the SSP-ICF common queue space. $IENBL also defines the subsystem environment, loads the requested subsystems (if they are not already loaded), and gives the subsystem control. When the subsystem is loaded, additional subsystem queue space is also allocated if required.

The enable function may allocate nonswappable main storage for SSP-ICF control routines, SSP-ICF common queue space, SSP-ICF subsystem queue space, and for the subsystem code. Allocating nonswappable storage decreases the user-available swappable storage. Enable will not allow the available swappable user storage to become less than the current maximum region size (including the region size for the enable function).

The subsystem environment is defined by a composite of the following:

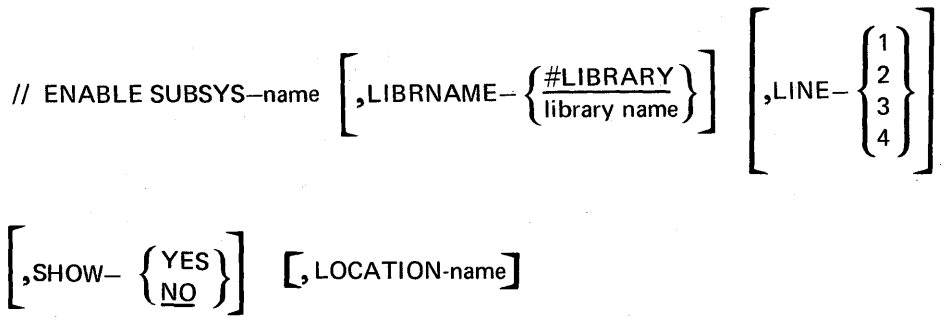
- ENABLE parameters
- Subsystem configuration variables
- System console display station communications variables (that is, $SETCF parameters)

After you run $IENBL, communications via the subsystem may take place.

The $IENBL utility program is normally called by the ENABLE procedure.

Utility Control Statement Format

To initialize an SSP-ICF subsystem:



Parameters

SUBSYS: The member name given to the subsystem configuration.

LIBRNAME: The name of the library containing the subsystem configuration. The default is #LIBRARY.

LINE: The number of the communications line to use (1, 2, 3, or 4).

SHOW: The SHOW option displays some of the parameters configured for a particular subsystem, and the operator has the option of modifying some of these parameters. For further information about the parameters displayed by the $IENBL utility, see the *SSP-ICF Reference Manual.*

LOCATION: The name of the remote location to be activated.

The SHOW-NO option does not display the configured parameters for a particular subsystem. This option is the default.

Control Statement Sequence

```
// LOAD $IENBL
// RUN
// ENABLE SUBSYS–name [,LIBRNAME– {#LIBRARY / library name}]
   [,LINE– {1 / 2 / 3 / 4}] [,SHOW– {YES / NO}] [,LOCATION-location name]
// END
```

Example

Initialize the SSP-ICF subsystem SUB1 for location SITEA, which resides in the user library SUBLIBR. If the SSP-ICF subsystem SUB1 is already enabled, add remote location SITEA to the subsystem. This example uses communications line 1, and displays a screen indicating the values that are in effect for this ENABLE:

```
// LOAD $IENBL
// RUN
// ENABLE SUBSYS-SUB1,LIBRNAME-SUBLIBR,LINE-1,
//        SHOW-YES,LOCATION-SITEA
// END
```

$INIT–Diskette Labeling and Initialization Utility Program

Function

The $INIT utility program performs three functions:

- Initializes (formats) diskettes
- Deletes files from diskettes
- Renames diskettes by changing volume label information

Note: $INIT can process a diskette 1 diskette on a diskette drive designed for the diskette 2D diskette, but $INIT cannot process a diskette 2D diskette on a diskette drive designed for a diskette 1 diskette.

Initializing a Diskette

To initialize a diskette 1 diskette, $INIT formats the diskette in the 128-byte or 512-byte format. To initialize a diskette 2D diskette, $INIT formats the diskette in the 256-byte or 1024-byte format. For information about diskette formats, see Appendix C.

During the initialization process, the diskette is checked for files whose expiration dates have not been met. If one or more unexpired files exist on the diskette, a message is displayed and the operator can cancel the job or delete all unexpired files on the diskette. If the operator continues the job, unexpired files are deleted and the remainder of the diskette is checked for defective tracks. $INIT checks for defective tracks, marking (flagging) any cylinder containing a defective track.

Cylinders flagged as defective are renumbered. If cylinder 0 or more than two other cylinders contain defective tracks, a message is displayed and initialization is terminated by $INIT. (The diskette is not usable.)

When initialization is complete, cylinders 1 through 74 on a diskette 1 diskette contain 128-byte or 512-byte sectors consisting of blanks; the volume ID and owner ID fields in the volume label on track 0 are replaced by the PACK and ID parameter values specified in the VOL utility control statement; and, the diskette VTOC indicates that an empty file, labeled DATA, occupied cylinders 1 through 73 (or 74).

When a diskette 2D diskette is initialized, cylinders 1 through 74 contain 256-byte or 1024-byte sectors consisting of hex F6.

Note: All diskettes for a multivolume file must be initialized in the same format.

Deleting Files from a Diskette

If the DELETE option of $INIT is requested, a message is displayed when any unexpired files exist on the inserted diskette. The operator can then cancel the job or delete the unexpired files. If the operator deletes the files, the VTOC for the diskette is set to indicate that an empty file, DATA, occupies cylinders 1 through 73. The volume ID specified in the VOL utility control statement is compared to the volume ID in the diskette volume label on track 0. They must be identical for deletion to occur. The owner ID specified in the VOL control statement is not compared with owner-id information in the volume label.

Renaming a Diskette Volume

If the RENAME option of $INIT is requested, only the volume label is changed. The vol-id and owner-id fields are replaced by the contents of the PACK and ID parameters in the VOL utility control statement. If a vol-id (PACK parameter) is not specified, the program date in year-month-day format is used. If owner-id (ID parameter) is not specified, OWNERID is used.

Diskette Defects Encountered during Output Processing

If a defect is discovered during diskette output processing, the system makes two more attempts (called retries) to write the bad sector. If the retries are not successful, the file is closed at the beginning of the operation during which the error occurred (normally at the start of a track). A message notifies the operator that the diskette contains a defect. The operator can then insert another diskette and continue the operation (which results in a multivolume file), or terminate the job and restart with a different diskette. The portion of the diskette that follows the error cannot be used.

For a diskette that contains an error encountered during an output operation, you must run $INIT or the INIT procedure before the portion of the diskette that follows the defect can be used. However, if $INIT discovers more than two defective tracks, the diskette is not usable.

Utility Control Statement Formats

To initialize a diskette:

```
// UIN OPTION– {FORMAT  }
               {FORMAT2 }

      [                 {S1   }  ]
      [                 {S2   }  ]
      [                 {S3   }  ]
      [, LOCATION–      {M1.01}  ] [, RECL– {number}  ]
      [                 {M2.01}  ]         {80    }  ]

// VOL [PACK– {vol-id      }] [, ID– {owner-id  }]
       [     {program date }]        {OWNERID  }]
// END
```

To delete files on a diskette:

```
// UIN OPTION–DELETE

      [                 {S1   }  ]
      [                 {S2   }  ]
      [                 {S3   }  ]
      [, LOCATION–      {M1.01}  ]
      [                 {M2.01}  ]

// VOL [PACK– {vol-id      }] [, ID– {owner-id  }]
       [     {program date }]        {OWNERID  }]
// END
```

To rename a diskette:

```
// UIN OPTION–RENAME

      [                 {S1   }  ]
      [                 {S2   }  ]
      [                 {S3   }  ]
      [, LOCATION–      {M1.01}  ]
      [                 {M2.01}  ]

// VOL [PACK– {vol-id      }] [, ID– {owner-id  }]
       [     {program date }]        {OWNERID  }]
// END
```

Note: The LOCATION parameter is ignored for systems without a diskette magazine drive.

Parameters

For the UIN statement:

OPTION: Specifies the $INIT function to be performed:

- OPTION-FORMAT specifies that a diskette 1 diskette is to be initialized in the 128-byte format or that a diskette 2D diskette is to be initialized in the 256-byte format.

  If OPTION-FORMAT is specified for one diskette in a multivolume file, it must be specified for all diskettes in the file.

- OPTION-FORMAT2 specifies that a diskette 1 diskette is to be initialized in the 512-byte format or that a diskette 2D diskette is to be initialized in the 1024-byte format.

  If OPTION-FORMAT2 is specified, a basic data exchange file cannot be placed on the diskette. Also, if OPTION-FORMAT2 is specified for one diskette in a multivolume file, it must be specified for all diskettes in the file.

- OPTION-DELETE specifies that all files are to be deleted from the diskette.

- OPTION-RENAME specifies that the volume ID and owner ID fields in the diskette volume label (track 0) are to be changed. If OPTION is not specified or if the UIN statement is not used, OPTION-RENAME is assumed.

LOCATION: Specifies the location of the diskette or diskettes to be processed:

- LOCATION-S1 specifies that the diskette in slot S1 should be processed. If LOCATION is not specified, LOCATION-S1 is assumed.

- LOCATION-S2 specifies that the diskette in slot S2 should be processed.

- LOCATION-S3 specifies that the diskette in slot S3 should be processed.

- LOCATION-M1.01 specifies that the diskettes in magazine M1 should be processed. Specifying LOCATION-M1 is the same as specifying LOCATION-M1.01.

- LOCATION-M2.01 specifies that the diskettes in magazine M2 should be processed. Specifying LOCATION-M2 is the same as specifying LOCATION-M2.01.

RECL: Specifies the block length in bytes. The maximum block length is the number of bytes in a sector, for example, if OPTION-FORMAT is specified for a diskette 1 diskette, the maximum block length is 128 bytes. If the RECL parameter is not specified, a block length of 80 bytes is assumed.

For the VOL statement:

PACK: During a diskette initialization or rename operation, the value specified by the PACK parameter is written into the volume ID field of the volume label. The specified volume ID can contain up to 6 alphameric characters. If the PACK parameter is not specified, the program date in year-month-day format is written in the volume ID field.

During a delete operation, the volume ID specified by the PACK parameter (or the session date if PACK is not specified) must be the same as the volume ID of the inserted diskette, or the delete operation is not performed.

ID: During a diskette initialization or rename operation, the value specified by the ID parameter is written into the owner ID field of the volume label. The specified owner ID can contain up to 14 characters. Any characters except quotes ('), commas (,), hyphens (-), and leading or embedded blanks can be specified. If ID is not specified, OWNERID is written into the owner ID field.

During a delete operation, $INIT ignores the ID parameter.

Note: The owner ID field is displayed when $LABEL is used to display the diskette VTOC. The SSP does not use the owner ID field for any other purpose.

Control Statement Sequence

```
// LOAD $INIT
// RUN
[ // UIN OPTION- { FORMAT   } ]
                 { FORMAT2  }
                 { DELETE   }
                 { RENAME   }

[ , LOCATION- { S1   } ] [ , RECL- { number } ]
              { S2   }             { 80     }
              { S3   }
              { M1.01 }
              { M2.01 }

[ // VOL [ PACK- { vol-id       } ] [ , ID- { owner-id  } ] ]
                { program date }            { OWNERID  }
// END
```

Examples

Example A

Change a diskette volume ID to JIM:

```
// LOAD $INIT
// RUN
// VOL PACK-JIM
// END
```

Example B

Initialize a one-sided diskette in 128-byte-per-sector format. The new volume ID is APRO and the new owner ID is FRANT:

```
// LOAD $INIT
// RUN
// UIN OPTION-FORMAT
// VOL PACK-APRO,ID-FRANT
// END
```

$LABEL–VTOC Display Utility Program

Function

The $LABEL utility program lists any of the following information on the system list device assigned to the requesting display station:

- The VTOC entry for an individual disk file
- The VTOC entries for all disk files in order by file location or by file name
- The VTOC entry for an individual diskette file
- The VTOC entries for all files on a diskette or a set of diskettes in order by file location or by file name

If $LABEL displays a disk VTOC within a job that contains a RESERVE OCL statement or that uses J files, $LABEL also lists a job file display following the disk VTOC display. The job file display contains entries for space used by the job's RESERVE OCL statement and J files.

For information about assigning the system list device, see the description of the SYSLIST procedure in Chapter 2.

The listed information is an up-to-date record of the contents of the disk or diskette. Some common reasons for wanting such information are:

- To check the contents of a diskette before reinitializing it to ensure that it does not contain active files (files that have not reached their expiration dates)
- To find out how much disk space is available for new files

  Note: If other programs are running when $LABEL is run, those programs might use space that is shown as available by the $LABEL listing.

- To obtain specific file information, such as the file label, retention classification (permanent or temporary), or the space allocated for the file

$LABEL is executed when the CATALOG procedure is run.

Sample Disk VTOC Displays

The following displays are disk VTOC displays. The first display shows the disk VTOC displayed by file location and the second display shows the disk VTOC displayed sequentially by file name. Following the displays is a list explaining the items on the displays.

Disk VTOC Display by File Location

```
PACK - QASCP     OWNER ID - OWNERID            D I S K   V T O C   D I S P L A Y                    DATE - 29/02/80
DEVICE CAPACITY -  13.2  MEGABYTES                    B Y   L O C A T I O N                          TIME - 08.48

 STATUS CODES    0 - CHECKPOINTED FILE OR LIBRARY      1 - DELETE CAPABLE FILE          2 - IMMEDIATE ACCESS FILE

                 3 - SECURED FILE OR LIBRARY           4 - PREVIOUSLY SECURED FILE OR LIB

----------------------------FILE----------------------------  -----ALLOCATION-----  RECORD  -----RECORDS-----  --KEY---
 LABEL      DATE    RETAIN  ORG  TYPE        STATUS   LOCATION  AMOUNT  FORMAT   PREF  LENGTH   USED  AVAILABLE  POS  LEN

#SYSWORK  29/02/80    P         SYSTEM                   240      180  RECORDS            256
#SYSHIST  29/02/80    P         SYSTEM                   258     2040  RECORDS            256
#SYSTASK  29/02/80    P         SYSTEM                   462     7020  RECORDS            256
#LIBRARY  00/00/00    P         SYSTEM         0        1164     1900  BLOCKS               0
##JOBQ    29/02/80    P         SYSTEM                  3064        4  BLOCKS             256
FILES     29/02/80    T    S                           3068       10  BLOCKS              80        0       320
FILESD    29/02/80    T    S                    1      3078       20  RECORDS            160        0        32
FILEI     29/02/80    P    I                    2      3080      200  RECORDS            256        0       209      1   10
FILEDD    29/02/80    P    D                    1      3102      200  RECORDS             64      200         0
CRLIB02R  00/00/00    P              LIBRARY    0,4     3107      200  BLOCKS     A1       0
INVENTRY  29/04/79    P    I                  2,3      3307        5  RECORDS            44        3        48      2    2
SALES     26/04/79    P    I                1,2,3      3308      800  RECORDS     A1     336       10       795      2   10
CRD921    10/05/79    P    S                           3418        1  BLOCKS             10        1       255
CRD902    27/04/79    T    S                  0,1      3419       10  RECORDS           500        4         6
MASTER    25/04/79    T    D                    3      3444     1000  RECORDS            57     1032         0
USILIB    00/00/00    P              LIBRARY           3467       50  BLOCKS              0
CKPTFOUT  00/00/00    T              CKRECORD   0      3517       56  BLOCKS              0
CRFRPG02  00/00/00    T              CKRECORD   0      3573       56  BLOCKS              0
CRFRPG14  00/00/00    T              CKRECORD   0      3629       56  BLOCKS              0
CKPTALL   00/00/00    P              LIBRARY           5112      200  BLOCKS     A2       0
CRLIB01R  00/00/00    P              LIBRARY           5312      100  BLOCKS     A2       0
CRD921    08/05/79    P    S                           5412        1  BLOCKS     A2      10        1       255

SPACE UNOCCUPIED BY  P  OR  T  FILES  --  LOCATION - BLOCKS  -  EACH 2560 BYTES
                                            3421       23
                                            3685     1427
                                            5413       11
USER VTOC ENTRIES - USED  17 / AVAILABLE 183
```

```
PACK - QASCP     OWNER ID - OWNERID            J C B   F I L E   D I S P L A Y                    DATE - 29/02/80
  JOBNAME -  JOB01                                                                                  TIME - 08.48

 STATUS CODES    0 - CHECKPOINTED FILE OR LIBRARY      1 - DELETE CAPABLE FILE          2 - IMMEDIATE ACCESS FILE

                 3 - SECURED FILE OR LIBRARY           4 - PREVIOUSLY SECURED FILE OR LIB

----------------------------FILE----------------------------  -----ALLOCATION-----  RECORD  -----RECORDS-----  --KEY---
 LABEL      DATE    RETAIN  ORG  TYPE        STATUS   LOCATION  AMOUNT  FORMAT   PREF  LENGTH   USED  AVAILABLE  POS  LEN

#RESERVE  29/02/80    J     S  RESERVE                 3685       60  BLOCKS              80
CKPTOUT   29/02/80    J     S  RESERVE                 3685     5000  RECORDS              6      250      4870
*****   END OF VTOC DISPLAY   *****
```

Disk VTOC Display by File Name

```
PACK - CASCP      OWNER ID - OWNERID              D I S K    V T O C    D I S P L A Y                    DATE - 29/02/80
DEVICE CAPACITY -  13.2  MEGABYTES                        B Y    F I L E N A M E                          TIME - 08.48
 STATUS CODES    0 - CHECKPOINTED FILE OR LIBRARY       1 - DELETE CAPABLE FILE          2 - IMMEDIATE ACCESS FILE
                 3 - SECURED FILE OR LIBRARY            4 - PREVIOUSLY SECURED FILE OR LIBR
```

FILE LABEL	DATE	RETAIN	ORG	TYPE	STATUS	LOCATION	ALLOCATION AMOUNT	ALLOCATION FORMAT	ALLOCATION PREF	RECORD LENGTH	RECORDS USED	RECORDS AVAILABLE	KEY POS	KEY LEN
##JOBQ	29/02/80	P		SYSTEM		3064	4	BLOCKS		256				
#LIBRARY	00/00/00	P		SYSTEM	0	1164	1900	BLOCKS		0				
#SYSHIST	29/02/80	P		SYSTEM		258	2040	RECORDS		256				
#SYSTASK	29/02/80	P		SYSTEM		462	7020	RECORDS		256				
#SYSWORK	29/02/80	P		SYSTEM		240	180	RECORDS		256				
CKPTALL	00/00/00	P		LIBRARY		5112	200	BLOCKS	A2	0				
CKPTFOUT	00/00/00	T		CKRECORD	0	3517	56	BLOCKS		0				
CRD902	27/04/79	T	S		0,1	3419	10	RECORDS		500	4	6		
CRD921	08/05/79	P	S			5412	1	BLOCKS	A2	10	1	255		
CRD921	10/05/79	P	S			3418	1	BLOCKS		10	1	255		
CRFRPG02	00/00/00	T		CKPECORD	0	3573	56	BLOCKS		0				
CRFRPG14	00/00/00	T		CKRECORD	0	3629	56	BLOCKS		0				
CRLIB01R	00/00/00	P		LIBRARY		5312	100	BLOCKS	A2	0				
CRLIB02R	00/00/00	P		LIBRARY	0,4	3107	200	BLOCKS	A1	0				
FILEDD	29/02/80	P	D		1	3102	200	RECORDS		64	200	0		
FILEI	29/02/80	P	I		2	3080	200	RECORDS		256	0	209	1	10
FILES	29/02/80	T	S			3068	10	BLOCKS		80	0	320		
FILESD	29/02/80	T	S		1	3078	20	RECORDS		160	0	32		
INVENTRY	29/04/79	P	I		2,3	3307	5	RECORDS		44	3	48	2	2
MASTER	25/04/79	T	D		3	3444	1000	RECORDS		57	1032	0		
SALES	26/04/79	P	I		1,2,3	3308	800	RECORDS	A1	336	10	795	2	10
USILIB	00/00/00	P		LIBRARY		3467	50	BLOCKS		0				

```
SPACE UNOCCUIPED BY  P  OR  T  FILES  --    1461 BLOCKS
SPACE CURRENTLY AVAILABLE  --    1401 BLOCKS
USER VTOC ENTRIES - USED  17 / AVAILABLE 183
```

```
PACK - CASCP      OWNER ID - OWNERID              J O B    F I L E    D I S P L A Y                    DATE - 29/02/80
  JOBNAME -  JOB01                                                                                       TIME - 08.48
 STATUS CODES    0 - CHECKPOINTED FILE OR LIBRARY       1 - DELETE CAPABLE FILE          2 - IMMEDIATE ACCESS FILE
                 3 - SECURED FILE OR LIBRARY            4 - PREVIOUSLY SECURED FILE OR LIBR
```

FILE LABEL	DATE	RETAIN	ORG	TYPE	STATUS	LOCATION	ALLOCATION AMOUNT	ALLOCATION FORMAT	ALLOCATION PREF	RECORD LENGTH	RECORDS USED	RECORDS AVAILABLE	KEY POS	KEY LEN
#RESERVE	29/02/80	J	S	RESERVE		3685	60	BLOCKS		80				
CKPTOUT	29/02/80	J	S	RESERVE		3685	5000	RECORDS		6	250	4870		

```
*****  END OF VTOC DISPLAY  *****
```

Item	Description
PACK	The volume identification in the disk volume label. The disk volume label (VOL1) is located on track 0.
OWNER ID	The owner identification.
DATE	The current session date in the current format.
DEVICE CAPACITY	The capacity in megabytes (1 megabyte = 1 million bytes) of the disk used by the system.
JOBNAME (Job File Display only)	The name of the job within which $LABEL was executed.
TIME	The time, based on the time specified by the system operator during IPL.
FILE LABEL	The label of the file described by the VTOC entry.
FILE DATE	The creation date of the file described.
FILE RETAIN	The retention classification of a file: P for permanent, T for temporary, J for job, S for scratch or step.
FILE ORG	The file organization: S for sequential, I for indexed, D for direct, blank for a system or library file.
FILE TYPE	Indicates the type of file: • LIBRARY indicates the file is a user library. • LIBRFILE indicates a library file created by $MAINT. • SYSTEM indicates the file is a system file. • Blanks indicate a data file. • CKRECORD indicates a checkpoint record file. • RESERVE indicates either a reserve area, or an S or a J file within a reserve area.

Item	Description
FILE STATUS	Indicates the current status of the file: 0 for an active checkpointed file 1 for a delete-capable file 2 for an immediate access to added records file 3 for a secured file or library 4 for a previously secured file or library *Notes:* 1. Either any combination or none of the above conditions, can represent the status. 2. If there are any checkpoint active scratch or job files in the VTOC, $LABEL displays the file labels of these files as #CHECKPT. The file retention is displayed as S or J, and the status is displayed as a checkpointed file. 3. Scratch and job file entries do not normally exist in the disk VTOC. However, while a job step is being checkpointed, scratch and job files exist in the VTOC until the job step completes normal termination.
FILE LOCATION	The block number, in decimal, of the beginning of data in a file and the disk identifier.
ALLOCATION AMOUNT	The number, in decimal, of blocks or records allocated for a file. *Note:* If the RECORDS parameter is used to allocate space, the number shown might be greater than the number requested because the system allocates space in blocks and rounds up to the next higher block whenever part of a block is required.

Item	Description
ALLOCATION FORMAT	The format in which a file was created; either BLOCKS or RECORDS.
ALLOCATION PREF	The preferred disk (if a preferred disk was specified when the file was created).
RECORD LENGTH	The length, in decimal number of bytes, of the records in a file. The record length given for libraries is 00.
RECORDS USED	The number, in decimal, of records currently contained in a data file.
RECORDS AVAILABLE	The number, in decimal, of records for which there is still room in a data file.
KEY POS	The position, in decimal, of the leftmost byte of the key in the records in an indexed file.
KEY LEN	The length, in decimal, of the keys in an indexed file.
SPACE UNOCCUPIED	The location (block number) and number of blocks unoccupied by P or T files if VTOC entries are displayed by location. The location and number of blocks are given for all space unoccupied by P or T files. The total number of blocks unoccupied by P or T files if VTOC entries are displayed by filename. If no space is available, the heading NO DISK SPACE AVAILABLE ON THIS SYSTEM appears.
SPACE CURRENTLY	The total number of blocks currently available to the user. This is the space not occupied by any file. This is displayed if VTOC entries are displayed by filename.
USER VTOC ENTRIES	The number of VTOC entries currently used and the number currently available. *Note:* Because other programs can run concurrently with $LABEL, the listed amount of available space might not be available when you try to use it.

Sample Diskette VTOC Display

Diskette VTOC Display by File Location

```
                    D I S K E T T E    D I S P L A Y    B Y    L O C A T I O N

  VOLUME ID - SAVQAS     OWNER ID - USERID                                    DATE  02/29/80      TIME  08.47

 SPACE AVAILABLE ON THIS VOLUME IS  1732 SECTORS -EACH  128 BYTES

USER VTOC ENTRIES - USED   6 / AVAILABLE  13

    FILE       FILE     SECTORS     FILE       FILE     RECORD     EXPRIATION       MVF    SEQUENCE      CREATING
    NAME       DATE     IN FILE     TYPE     LOCATION   LENGTH       DATE          FILE    NUMBER        SYSTEM

  FILE.B     02/29/80       1     COPYFILE        1        256     PROTECT                            IBMSYSTEM34
  FILE.A     02/29/80      21     COPYFILE        2         80     PROTECT                            IBMSYSTEM34
  FILE.C     02/29/80       1     COPYFILE       23         80     PROTECT                            IBMSYSTEM34
  PUBSLIB    02/29/80      29     LIBRFILE       24         08     PROTECT                            IBMSYSTEM34
  DATA       02/29/80      20     EXCHANGE       53         80     PROTECT                            IBMSYSTEM34
  DATAI      02/29/80      40     IFORMAT        73        128     03/10/80                           IBMSYSTEM34
*****   END OF VTOC DISPLAY   *****
```

Diskette VTOC Display by File Name

```
                    D I S K E T T E    D I S P L A Y    B Y    F I L E N A M E

  VOLUME ID - SAVQAS     OWNER ID - USERID                                    DATE  02/29/80      TIME  08.47

 SPACE AVAILABLE ON THIS VOLUME IS  1732 SECTORS -EACH  128 BYTES

USER VTOC ENTRIES - USED   6 / AVAILABLE  13

    FILE       FILE     SECTORS     FILE       FILE     RECORD     EXPRIATION       MVF    SEQUENCE      CREATING
    NAME       DATE     IN FILE     TYPE     LOCATION   LENGTH       DATE          FILE    NUMBER        SYSTEM

  DATA       02/29/80      20     EXCHANGE       53         80     PROTECT                            IBMSYSTEM34
  DATAI      02/29/80      40     IFORMAT        73        128     03/10/80                           IBMSYSTEM34
  FILE.A     02/29/80      21     COPYFILE        2         80     PROTECT                            IBMSYSTEM34
  FILE.B     02/29/80       1     COPYFILE        1        256     PROTECT                            IBMSYSTEM34
  FILE.C     02/29/80       1     COPYFILE       23         80     PROTECT                            IBMSYSTEM34
  PUBSLIB    02/29/80      29     LIBRFILE       24         08     PROTECT                            IBMSYSTEM34
*****   END OF VTOC DISPLAY   *****
```

Item	Description
VOLUME ID	The volume ID given in the diskette volume label. The volume label for basic data exchange format diskettes is described in *The IBM Diskette General Information Manual*. Also see Appendix C of this manual.
OWNER ID	The owner identification given in the diskette volume label. The volume label for basic data exchange format diskettes is described in the *The IBM Diskette General Information Manual*. Also see Appendix C of this manual.
DATE	The current session date in the current format.
TIME	The time, based upon the time specified by the system operator during IPL.
SPACE AVAILABLE ON THIS VOLUME	The number, in decimal, of sectors available on diskette. The number represents space following the last active (unexpired) file on the diskette.
USER VTOC ENTRIES	The number of VTOC entries currently used and the number currently available.
FILE NAME	The label specified when the file described was created.
FILE DATE	The creation date of a file.
SECTORS IN FILE	The number, in decimal, of sectors contained in a file. *Note:* Diskette files created by some System/34 utility programs have a control record at the beginning of the file. Consequently, the number of sectors might be larger than expected. When the file is returned to the disk, the control information is dropped and the sector count returns to the original number.

Item	Description	
FILE TYPE	The type of file:	
	APARFILE	Created by the $FEAPR utility program
	BACKUP	Created by the $BACK utility program
	COPYFILE	Created by the $COPY utility program
	EXCHANGE	A basic data exchange file
	IFORMAT	An I exchange file
	IGC EXTN	An IGC extended character file
	LIBRFILE	Created by the $MAINT utility program
	OLMVFILE	An offline multivolume file
	PROFILE	Created by the $PRSV utility program
	UNKNOWN	The file is not one of the preceding types
FILE LOCATION	The sector number, in decimal, of the beginning of data in a file.	
RECORD LENGTH	The length, in decimal number of bytes, of the records in a file.	
EXPIRATION DATE	The expiration date of a file. PROTECT indicates that a diskette file is permanent (RETAIN-999).	
MVF FILE	An indicator to specify whether a file is a multivolume file: blank for a single-volume file, CONTINUED for any volume but the last volume of a multivolume file, LAST for the last volume of a multivolume file.	
SEQUENCE NUMBER	The diskette sequence number within a multivolume file if the diskette is part of a multivolume file.	
CREATING SYSTEM	The 13-character identifier of the system that created the file. For files created by System/34, the field contains IBMSYSTEM/34.	

To display the following:

- VTOC entry for a particular disk file
- VTOC entries for the entire disk by file location
- VTOC entries for the entire disk in ascending alphabetical sequence by file name

```
// DISPLAY UNIT-F1,LABEL-{ALL / filename} [,SORT- {LOCATION / NAME}]
```

To display the following:

- VTOC entry for a particular diskette file
- VTOC entries for a particular diskette or set of diskettes by file location
- VTOC entries for a particular diskette or set of diskettes in ascending alphabetical sequence by file name

```
// DISPLAY UNIT-I1,LABEL-{ALL / filename} ,

LOCATION-{S1 / S2 / S3 / M1.nn / M2.nn} ,AUTO-{YES / NO} [,SORT- {LOCATION / NAME}]
```

Note: The LOCATION and AUTO parameters are ignored for systems without a diskette magazine drive.

Parameters

UNIT: UNIT-I1 specifies that diskette VTOC information is to be listed. UNIT-F1 specifies that disk VTOC information is to be listed.

LABEL: LABEL-ALL specifies that the entire contents of the VTOC are to be listed. If a file label is specified, only the entry for the specified file is listed. If more than one file exists with the specified label, $LABEL lists the VTOC entries for all of those files.

LOCATION: Specifies a diskette location.

S1, S2, or S3 identifies the first individual diskette slot to be used by $LABEL. If the LOCATION parameter is not specified, LOCATION-S1 is assumed.

M1.nn or M2.nn identifies the first magazine location to be used by $LABEL. M1 indicates the first magazine, and M2 indicates the second magazine. nn is a decimal value from 01 to 10 that identifies the location of the diskette in the magazine. (Specifying M1 is the same as specifying M1.01; specifying M2 is the same as specifying M2.01.)

AUTO: Controls diskette processing in the diskette slot or the magazine drive.

AUTO-NO specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, $LABEL uses only the specified slot. If LABEL-ALL is specified, $LABEL ends when the diskette in the specified slot is cataloged.
- If M1.nn or M2.nn is specified in the LOCATION parameter, $LABEL starts with the specified location and ends after going through the last position of the specified magazine.

AUTO-YES specifies:

- If S1, S2, or S3 is specified in the LOCATION parameter, $LABEL starts with the specified slot and ends after cataloging the diskette in slot S3.
- If M1.nn or M2.nn is specified in the LOCATION parameter, $LABEL starts with the specified location and ends after cataloging the last diskette in magazine M2.

If AUTO is not specified, AUTO-YES is assumed.

SORT: Controls the sequence in which the VTOC entries are displayed.

SORT-LOCATION specifies: The VTOC entries are to be sorted and displayed by physical file location.

SORT-NAME specifies: The VTOC entries are to be sorted and displayed by file name in ascending alphabetic sequence.

If the SORT parameter is specified for LABEL-filename, it is ignored. If SORT is not specified, SORT-LOCATION is assumed.

Control Statement Sequence

```
// LOAD $LABEL
// RUN
// DISPLAY UNIT-{I1 | F1} ,LABEL-{ALL | file label}
// END
```

Examples

Example A

List the VTOC entry for a disk file labeled PAYR:

```
// LOAD $LABEL
// RUN
// DISPLAY UNIT-F1,LABEL-PAYR
// END
```

Example B

List all VTOC entries for a diskette in ascending alphabetic sequence by file name:

```
// LOAD $LABEL
// RUN
// DISPLAY UNIT-I1,LABEL-ALL,LOCATION-S1,AUTO-NO,SORT-NAME
// END
```

$LOADI–Reload Library Utility Program

Function

The $LOADI utility program initiates IPL from a diskette file produced as output from the BACKUP procedure or the $BACK utility program.

Space allocations for the system library (#LIBRARY) and the library directory within the system library file are displayed on the display screen during execution of $LOADI. The allocations can be altered at the time they are displayed. Unaltered allocations remain as they were when the diskette file was created by the BACKUP procedure or the $BACK utility program. In addition to changing the space allocations for the system library, the operator can also:

- Change the size of the task work area
- Specify that the disk VTOC be rebuilt
- Specify that the system configuration record be rebuilt
- Change the size of the history file
- Change the number of possible VTOC entries

For detailed operator instructions, see the *Operator's Guide*.

The $LOADI utility is the only means of increasing the size of the system library directory. When using $LOADI, however, be aware that system library members that exist on the disk but do not exist in the back-up system library are lost when the back-up system library is returned to the disk. If system library members on the disk should be saved, use the FROMLIBR procedure to copy them before executing $LOADI; then, use TOLIBR to replace them in the library after the library is reloaded. The size of the reloaded system library might have to be increased for the additional members.

$LOADI is executed when the RELOAD procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $LOADI
// FILE NAME-#LIBRARY,UNIT-I1
// RUN
```

$MAINT–Library Maintenance Utility Program

Function

The $MAINT utility program performs the functions required for creating and maintaining libraries on System/34. For a description of libraries and their contents, see the *Concepts and Design Guide*.

$MAINT can be used to perform any of the following general functions:

- *Allocate:* Allocate a library or change the size of an existing library or library directory.
- *Copy:* Copy library members to, from, within, or between libraries, or display the contents of a library or file created by $MAINT.
- *Delete:* Delete members from a library.
- *Compress:* Remove unusable space within a library.

Notes:

1. You should not delete, rename, or copy SSP members, even though $MAINT provides the capability to do so.
2. To increase the size of the system library (#LIBRARY) or the system library directory, use the RELOAD procedure or the $LOADI utility. The $MAINT utility can be used to decrease the size of the system library or the system library directory.

Following are descriptions of the specific functions performed by $MAINT.

Allocate Functions

$MAINT allocate functions are requested via an ALLOCATE utility control statement. Specific allocate functions performed by $MAINT are:

- Creates a library with a specified size.
- Increases the size of an existing library. The size of the library can be increased only if unused disk space immediately follows the library.
- Decreases the size of an existing library. The number of blocks deleted cannot exceed the amount of unused space in the library.
- Changes the size of a user library directory. (The directory size can be increased only if unused disk space immediately precedes the library. The directory size can be decreased only if unused space exists in the directory.)

Copy Functions

$MAINT copy functions are requested via a COPY utility control statement. Specific copy functions performed by $MAINT are:

- Reader-to-library operations: Replaces or adds a member read from the system input device.

- Library-to-library operations:
  - Copies a specified member and, optionally, changes the name of the member. The member can be copied to the same library or to a different library.
  - Copies members of a specified type, or all types, that have names beginning with certain characters. Members can be copied to the same library or to a different library.
  - Copies members of a specified type, or all types, omitting members that have a certain name or names beginning with certain characters, or omitting all SSP members. Members can be copied to the same library or to a different library. (If members are copied within a library, the new members must be given different names.)

- Library-to-file operations (record mode or sector mode):
  - Copies to a disk or diskette file a member having a specified name or all members having a specified name.
  - Copies to a disk or diskette file members of a specified type, or all types, that begin with certain characters.
  - Copies to a disk or diskette file all members of a specified type or all types.
  - Copies to a disk or diskette file all members of a specified type, omitting members that have a certain name or names beginning with certain characters, or omitting all SSP members.
  - Copies to a disk or diskette file all members of all types, omitting members that have a certain name or names beginning with certain characters.
  - Copies to a disk or diskette file all members that have had a PTF (program temporary fix) applied (sector mode only).
  - Adds library members to an existing file that contains library members. The added members need not be from the same library as the members already in the file.
  - Copies a member(s) to a basic data exchange diskette file.

- File-to-library operations:
  - Copies a member or members from a file to a library. (The copied members can replace members with the same name that already exist in the library.)

    Note: Ordinarily, you should not replace a member that is being used. If it is necessary to replace a member in use, do not compress the library until the replaced member is no longer being used.

  - Copies members that have a specified PTF log number in that library. The PTF is logged in the PTFLOG member in the system library.
  - Copies a member in a basic data exchange diskette file directly to a library.

- File-to-printer operations: Prints the type and the name of all members in a record-mode file created by $MAINT. Prints the directory entry and the disk or diskette address in the file of all members in a sector-mode file created by $MAINT.

- Library-to-printer operations: When statements are printed, blanks are reinserted into statements from source and procedure members. Load and subroutine members are printed in hexadecimal format.
  - Prints a member having a specified name or all members having a specified name.
  - Prints all members of a certain type.
  - Prints members of a specified type, or all types, that begin with certain characters.
  - Prints members of a specified type or all types, omitting members that have a certain name or have names beginning with certain characters, or omitting all SSP members.
  - Prints directory entries for members of a specified type.
  - Prints directory entries for members of a specified type or all types, omitting entries for members that have a certain name or that have names beginning with certain characters, or omitting all SSP members.
  - Prints all directory entries and the library status.
  - Prints the library status.

  Note: Files are not extended when you copy members from a library to disk unless you are in record mode.

Figure 4-3 shows a sample library status listing. Figure 4-4 shows the printout of a library directory entry. Figure 4-4 is followed by an explanation of the fields shown.

Note: In the column at the right, the first of the two numbers is a decimal number; the second number is its hexadecimal equivalent. For example, in the entry 298/00012A, 298 is the decimal value of the hexadecimal number 00012A.

```
USERLIB1 STATUS                    DATE   79/09/20      TIME   17.57

START SECTOR OF LIBRARY                  502521/07AAF9
END SECTOR OF LIBRARY                    503020/07ACEC
TOTAL NUMBER OF LIBRARY BLOCKS              50/000032
START SECTOR OF DIRECTORY                502522/07AAFA
END SECTOR OF DIRECTORY                  502530/07AB02
NUMBER OF DIRECTORY SECTORS                 10/00000A
ACTIVE DIRECTORY ENTRIES                    21/000015
AVAILABLE DIRECTORY ENTRIES                 59/00003B
START SECTOR OF LIBRARY MEMBERS          502531/07AB03
END SECTOR OF LIBRARY MEMBERS            503020/07ACEC
ACTIVE LIBRARY MEMBER SECTORS              298/00012A
AVAILABLE MEMBER SECTORS                   188/0000BC
NEXT AVAILABLE MEMBER SECTOR             502833/07AC31
```

Figure 4-3. Printout of Library Status

USERLIB1 DIRECTORY DATE 79/09/20 TIME 17.57

YPE	NAME	DISK ADDR	TOTAL	NUM TEXT/RECORD	ATTRIBUTES	LINK ADDR/NUM STMT	RLD DISP	ENTRY ADDR	PROG SIZE	MRT	LEVEL
O	#PTFLOG	502531/07AB03	1/0001	0/00	80080000	0/0000	0/00	0/0000	0/00	0	5
O	AGGDX	502532/07AB04	15/000F	15/0F	40000000	0/0000	13/00	0/0000	16/10	0	3
O	AGGDXAOW	502547/07AB13	5/0005	5/05	10080000	0/0000	0/00	0/0000	0/00	0	1
O	AGGDXROW	502552/07AB18	5/0005	5/05	10080000	0/0000	0/00	0/0000	0/00	0	1
O	CPTEST	502561/07AB21	8/0008	8/08	10080000	0/0000	0/00	0/0000	0/00	0	1
O	CPTEST##	502569/07AB29	2/0002	0/00	00080000	0/0000	0/00	0/0000	0/00	0	1
O	FKMRT	502592/07AB40	31/001F	31/1F	40010000	0/0000	239/EF	0/0000	10/0A	4	3
O	TESTIME	502571/07AB2B	17/0011	17/11	40000000	51200/C800	151/97	51200/C800	4/04	0	1
O	TIMET09	502588/07AB3C	4/0004	4/04	40000000	51200/C800	72/48	51206/C806	0/00	0	3
O	WTCDPY	502623/07AB5F	18/0012	18/12	40000000	0/0000	120/78	0/0000	10/0A	8	1
P	FKMRT	502641/07AB71	1/0001	80/50	00000000	2/0002				YES	1
P	MYPROC1	502643/07AB73	5/0005	110/6E	00000000	36/0024					1
P	MYPROC2	502648/07AB78	4/0004	110/6E	00000000	25/0019					4
P	PRNTFILE	502652/07AB7C	2/0002	72/48	00000000	17/0011					3
P	STAT	502654/07AB7E	1/0001	64/40	00000000	2/0002					4
P	WTCDPY	502642/07AB72	1/0001	80/50	00000000	2/0002				YES	1
R	SUBR08	502655/07AB7F	2/0002	0/00	44000000						4
S	FEDGA	502657/07ABR1	92/005C	80/50	00000000	683/02AB					0
S	FEDGB	502749/07ABDD	82/0052	80/50	00000000	597/0255					3
S	SRC01	502831/07AC2F	1/0001	40/28	00000000	20/0014					5
S	SRC02	502832/07AC30	1/0001	96/60	00000000	25/0019					5

Figure 4-4. Information in Printout of Library Directory

Following is an explanation of the fields shown in Figure 4-4.

FIELD	DESCRIPTION
TYPE	Type of library member described by the entry: S Source member P Procedure member O Load member R Subroutine member
NAME	The name of the library member.
DISK ADDR	The sector address (in decimal and hexadecimal) of the first sector of the library member.
TOTAL	The total number of sectors (in decimal and hexadecimal) in the library member.
NUM TEXT/ RECORD	For source and procedure members, the record length of the member (in decimal and hexadecimal). For load members, the number of text sectors contained in the member, excluding RLDs (relocation dictionaries), which are the parts of a load member that are used for adjusting main storage addresses when the member is moved to main storage. For subroutine members, the category value assigned when the subroutine was assembled.
ATTRIBUTES	Four bytes (32 bits) of *attributes* giving detailed characteristics of the member. Displayed in hexadecimal.

Bit	Meaning When On (set to 1)
	Byte 0:
0	This is an SSP member. This bit is used to prevent SSP members from being deleted.
1	For a load member (O member), the program can execute privileged SVC (supervisor call) instructions. For a procedure member (P member), the statements generated by the procedure are not logged in the history file.
2	The program is noninquirable.

Bit	Meaning When On (set to 1)
	Byte 0 (continued):
3	For a load member (O member), the member is a screen format load member. For a procedure member (P member), only data (not parameters) can be passed on the INCLUDE OCL statement that invokes the procedure.
4	This program requires that $WORK and $SOURCE be allocated. $SOURCE must be filled from the keyboard or a source member.
5	This SSP module is not part of the basic SSP system.
6	A PTF (program temporary fix) has been applied to this program.
7	This load member contains overlays.
	Byte 1:
0	This is a dedicated program (the program cannot be run while other jobs are executing).
1	This is an NEP (never-ending program).
2	This module has a cross-reference format index table. Valid for SSP load members only.
3	This program can be loaded only from the system console.
4	This is a nonexecutable member (it cannot be loaded with a LOAD OCL statement).
5	This program requires that a new load address be calculated at load time to ensure that the program is placed in main storage at a point beyond its own common region.
6	This program reads utility control statements.
7	This program contains a where-to-go table. It is used by the transient cross-reference resolver program (#MAXRF). Valid only for SSP load members.

Bit	Meaning When On (set to 1)
	Byte 2:
0	This program requires that $WORK2 be allocated.
1	This program is nonswappable.
2	This program cannot be run if any other jobs are running or if a user is signed on at any other display station.
3	This program requires the scientific instruction set.
4	This member is a configuration record.
5-7	Reserved.
	Byte 3:
0	This program requires the BASIC instruction set.
1	This member is an IBM pad member.
2-7	Reserved.

LINK ADDR/ NUM STMT

For load members, the main storage address (in decimal and hexadecimal) assigned to the member when it is linked in main storage with other load members.

For source or procedure members, the number of statements in the member.

RLD DISP

For load members only. The displacement (in decimal and hexadecimal) of the first RLD (relocation dictionary) in the first sector containing RLDs.

ENTRY ADDR

For load members only. The main storage address (in decimal and hexadecimal) of the entry point of the member.

PROG SIZE

For load members only. The number of sectors (in decimal and hexadecimal) required to run the program contained in the member.

MRT

For load members, the maximum number of requesting display stations that can be be attached.

For procedure members, YES indicates a multiple requestor terminal procedure.

LEVEL

The release level of the SSP when the member was created or when the member was updated via SEU or SDA. For members copied from an IBM System/32 sector-mode file, the version level is 232.

Delete Functions

$MAINT delete functions are requested via a DELETE utility control statement. Specific delete functions performed by $MAINT are:

- Deletes non-SSP members of a specified type, or non-SSP members of all types, that have a specified name.
- Deletes non-SSP members of a specified type, or non-SSP members of all types, that have names beginning with certain characters.
- Deletes all non-SSP members of a specified type or non-SSP members of all types.
- Deletes non-SSP members of a specified type or non-SSP members of all types, except members that have a certain name or that have names beginning with certain characters.
- Deletes specified members, including SSP members.

Notes:

1. When $MAINT is used to delete SSP members or to delete all non-SSP members from #LIBRARY, $MAINT can be run only from the system console, and cannot be run (a) while any other jobs are being run, (b) while a user is signed on at any other display station, (c) while remote work stations are varied online, (d) while SSP-ICF subsystems are enabled, (e) while a communication line is enabled, or (f) while extended trace is active. When non-SSP members are deleted, system tasks may be active.
2. When $MAINT is being used to delete all members from a user library, there can be no other users of the library.
3. Ordinarily, you should not delete a member that is being used. If it is necessary to delete a member in use, do not compress the library until the deleted member is no longer being used.

Compress Function

The $MAINT compress function is requested via a COMPRESS utility control statement. When the compress function is requested, $MAINT moves all members to the front of the library. All unused space is accumulated at the end of the library.

Note: When $MAINT is used to compress a user library, no other jobs can be using that library. When $MAINT is used to compress the system library, no other programs can be running on the system.

$MAINT is executed when the BLDLIBR, CONDENSE, FROMLIBR, LIBRLIBR, LISTFILE, LISTLIBR, REMOVE, or TOLIBR procedure is run.

Checkpoint Active Libraries

The following library functions of $MAINT are not allowed for checkpoint active libraries because they require only one user:

- Condensing the library
- Removing all non-SSP members of a specified type or all types
- Replacing library members into the same space when copying
- Reusing the space at the end of the library

Utility Control Statement Formats

For Allocate Functions

```
// ALLOCATE [LIBRSIZE–blocks | INCREASE–blocks | DECREASE–blocks] [,LIBRNAME–{library name | #LIBRARY}]
          [,STATUS–{CREATE | CHANGE}][,LOCATION–{A1 | A2 | block number}]
          [,DIRSIZE–sectors]
// END
```

For Copy Functions

To copy from reader to library:

```
// COPY FROM–READER,LIBRARY–{P | S},NAME–name,TO–{F1 | library name}
     [,RETAIN–{P | R}] [,RECL–{number | 120}] [,MRT–YES]
     [,HIST–{YES | NO}] [,PDATA–{YES | NO}]
      •
      •
      •
// CEND
// END
```

Note: When copying from the reader, the statements to be copied must immediately follow the COPY utility control statement and must be terminated with a CEND statement.

To copy from library to library:

```
// COPY FROM-{F1 | library name},LIBRARY-{S | P | O | R | ALL},NAME-{ALL | characters.ALL | name}
        ,TO-{F1 | library name},[NEWNAME-{characters | name}][,RETAIN-{P | R}]
        [,OMIT-{name | characters.ALL | SYSTEM}]
// END
```

To copy from a library to a file in record mode:

```
// COPY FROM-{F1 | library name},TO-DISK,FILE-file name,{RECL-number | ADD-YES}
        ,NAME-{name | characters.ALL | ALL},LIBRARY-{S | P | ALL}
        [,OMIT-{name | characters.ALL | SYSTEM}] [,BASIC-{YES | NO}]
        [,SVATTR-{YES | NO}]
// END
```

Note: Source and procedure member copies made in record mode are preceded by a COPY statement and followed by a CEND statement. For further information about record-mode files and for the formats of the COPY and CEND statements, see *Storing Library Members in Disk or Diskette Files* in Chapter 6.

To copy from a library to a file in sector mode:

```
// COPY FROM-{F1 | library name},TO-DISK,FILE-file name
        ,NAME-{name | characters.ALL | ALL},LIBRARY-{S | P | O | R | ALL} [,ADD-{YES | NO}]
        [,OMIT-{name | characters.ALL | SYSTEM}] [,PTF-{YES | NO}]
// END
```

To copy a file to a library:

```
// COPY FROM-DISK,TO-{F1 | library name} [,RETAIN-{P | R}]
        ,FILE-file name [,PTF-{NO | number}] [,OMIT-NEW]
// END
```

To copy from a file to the system list device:

```
// COPY FROM–DISK,TO–PRINT,FILE–file name
// END
```

To copy from a library to the system list device:

```
// COPY FROM–{F1 | library name},LIBRARY–{S | P | O | R | ALL | SYSTEM}
        ,NAME–{DIR | name | characters.ALL | ALL},TO–PRINT [,OMIT–{name | characters.ALL | SYSTEM}]
// END
```

For Delete Functions

```
// DELETE LIBRARY–{S | P | O | R | ALL} [,NAME–{ALL | characters.ALL | name}]
        [,LIBRNAME–{library name | #LIBRARY}] [,OMIT–{characters.ALL | name}]
        [,RETAIN–S]
// END
```

For the Compress Function

```
// COMPRESS [LIBRNAME–{library name | #LIBRARY}]
// END
```

Parameters

For the ALLOCATE statement:

LIBRSIZE: Specifies the number of blocks to be allocated for the library (1 block = ten 256-byte sectors).

INCREASE: Specifies the number of blocks by which the library size is to be increased. INCREASE cannot be specified if DIRSIZE is specified.

DECREASE: Specifies the number of blocks by which the library size is to be decreased. DECREASE cannot be specified if DIRSIZE is specified.

LIBRNAME: Specifies the name of the library to be allocated or whose size is to be changed. If LIBRNAME is not specified, the system library (#LIBRARY) is assumed.

STATUS: STATUS-CREATE specifies the creation of a user library. STATUS-CHANGE specifies that the amount of space allocated to a user library or the system library is to be changed. If the STATUS parameter is not specified, STATUS-CHANGE is assumed.

LOCATION: Specifies a preferred disk placement or specifies the starting block number for a new library. LOCATION is valid only when STATUS-CREATE is specified.

If a preferred disk is specified, the SSP allocates the new library as follows:

- If you specify disk A1, the library is allocated in the first segment (lowest address) on disk 1 that is large enough to contain the library. If not enough space is available on disk A1, the SSP attempts to allocate the library on disk A2.

- If you specify disk A2 for a multiple-disk configuration, the SSP allocates the library in the last segment (highest address) of available storage on A2 that is large enough to contain the file. Although disks 2 through 4 are physically separate, the system treats these disks as one logical disk: A2 is the logical name. For example:
  - On a three-drive system, the library is placed in the last available segment of disk 3
  - On a four-drive system, the library is placed in the last segment of disk 4

If not enough space is available on the last physical disk of logical disk A2, the SSP attempts to find space on the other disks that make up A2. If not enough space is available on logical disk A2, the SSP attempts to allocate the library on disk A1.

If a block number is specified, the SSP attempts to allocate the library at the specified location. The following table gives the beginning block number location for disks 2 through 4:

Disk	Beginning Block Number Location
Disk 2	25203
Disk 3	50406
Disk 4	75609

If not enough space is available at the specified block number location, the SSP does not allocate the library.

DIRSIZE: Specifies the number of sectors for the directory of the new library or the new directory size if you are changing the size of the directory for a user library. The minimum directory size is 2 sectors. The maximum size is 256 sectors. The first sector in the directory is the library control sector. The remaining sectors contain directory entries. Each sector except the last sector can contain nine entries; the last sector can contain only eight entries. DIRSIZE cannot be specified with STATUS-CHANGE if LIBRSIZE, INCREASE, or DECREASE is specified.

For the COPY statement:

FROM: Specifies the device from which the member or members are to be copied:

- FROM-READER indicates copy from the system input device. When FROM-READER is specified, the statements to be copied must immediately follow the COPY utility control statement and must be terminated with a CEND statement.

  Note: For the ideographic version of the SSP, the statements to be copied can contain ideographic characters.

- FROM-F1 or FROM-#LIBRARY indicates copy from the system library.

- FROM-library name indicates copy from the specified library.

- FROM-DISK indicates copy from disk or diskette. A FILE OCL statement must define the file.

TO: Specifies the device to which the member or members are to be copied:

- TO-F1 or TO-#LIBRARY indicates copy to the system library.

- TO-library name indicates copy to the specified library.

- TO-DISK indicates copy to disk or diskette. A FILE OCL statement must define the file.

- TO-PRINT indicates copy to the system list device.

LIBRARY: Specifies the type of the member or members to be copied:

- LIBRARY-S indicates source members.

- LIBRARY-P indicates procedure members.

- LIBRARY-O indicates load members.

- LIBRARY-R indicates subroutine members.

- LIBRARY-ALL indicates all member types unless:
  - Members are copied from library to file in record mode. In this case, LIBRARY-ALL specifies source (S) and procedure (P) members.
  - The directory area is being printed (NAME-DIR). In this case, LIBRARY-ALL specifies that status information and directory entries are to be printed.

- LIBRARY-SYSTEM indicates that status information is printed for the system library. LIBRARY-SYSTEM is valid only if NAME-DIR is specified.

NAME: Specifies the name or names of the members to be copied or listed:

- NAME-DIR specifies that the library directory entries are to be listed.
- NAME-name specifies the name or names of members to be copied or listed.
- NAME-characters.ALL specifies that members whose names begin with the specified characters are to be copied or listed. Up to 7 characters can be specified. For example, NAME-PAY.ALL copies or prints all members whose names begin with PAY. (Duplicate members with different names can be created in a library-to-library copy. See the description of the NEWNAME parameter.)
- NAME-ALL specifies that all members, except SSP members, are to be copied or listed when LIBRARY-ALL is specified, or that all members of the type specified by the LIBRARY parameter are to be copied.

RETAIN: The RETAIN parameter is used when a member is copied into a library:

- RETAIN-P specifies that if a member with the same name as the new member already exists in the library, the system displays a message. The operator must then decide if the duplicate member should be replaced. If the RETAIN parameter is not specified, RETAIN-P is assumed.
- RETAIN-R specifies that if a member with the same name as the new member already exists in the library, the existing member is replaced and no operator message is displayed.

RECL: Specifies the record length, in bytes, of a source or procedure member. The record length can be from 40 through 120. If FROM-READER is specified, RECL defaults to 120 if omitted. If TO-DISK is specified, RECL indicates that the copy of a file is in record mode, not sector mode.

- Record mode is specified by the RECL parameter used with the TO-DISK parameter. Record mode can be specified only for source and procedure members. Source and procedure member copies made in record mode are preceded by a COPY statement and followed by a CEND statement.

  For the formats of the COPY and CEND statements, see *Storing Library Members in Disk or Diskette Files* in Chapter 6.

  The member itself is in expanded format; that is, the data is not compressed–all blanks are included.

- Sector mode is specified by the TO-DISK parameter without the RECL-number parameter. A sector mode copy can be specified for any type of library member (source, procedure, load, or subroutine). In sector mode, copies are in hexadecimal format and consist of control information and PTF (program temporary fix) numbers for any PTFs that have been applied to a member, followed by the member as it exists in the library.

ADD: ADD-YES specifies that a member or members are being added to a file that already contains members. If ADD-YES is not specified, ADD-NO is assumed.

Notes:

1. When adding a member to a disk file, the file must contain enough unused space to hold the member. When adding a member to a diskette file, the file must be the last active (unexpired) file on the diskette.
2. The RECL parameter is not allowed if ADD-YES is specified. For record mode, the record length is determined by the record length of the existing file.
3. If ADD-YES is specified, BASIC-YES should not be specified. When you add records to a basic exchange file, do not specify BASIC-YES.

NEWNAME: Specifies the name for a new library member, or specifies the beginning characters of the name for a new library member or members. If the beginning characters are specified, the number of characters must be the same as the number of characters specified in the NAME-characters.ALL parameter.

NEWNAME is valid only for a library-to-library copy. It is required for a copy within a library and optional for a copy from one library to another. NEWNAME cannot be specified if NAME-ALL is specified.

OMIT: OMIT specifies the name of a member or members to be omitted from the copy or display:

- OMIT-characters.ALL specifies that all members whose names begin with the specified characters are to be omitted from the copy or display. Up to 7 characters can be specified.
- OMIT-SYSTEM indicates that all SSP members are omitted.
- OMIT-NEW is valid only for a file-to-library operation. OMIT-NEW indicates that only members that have the same name as existing library members are copied to the library.

FILE: Specifies the name of the file that was specified on the FILE OCL statement for the input or output file.

PTF: PTF specifies that members with PTFs applied should be copied or that members with specified PTF numbers should be copied:

- PTF-YES specifies that only members that have PTFs applied are to be copied. PTF-YES is valid only when copying from a library to a file.
- PTF-number specifies that only members with the specified PTF number (00001 through 65535) are copied to the library. PTF-number is valid only when copying in sector mode from file to library.
- PTF-NO specifies that PTFs have no particular significance. A member is copied if the name is the same as the specified name. If PTF is not specified, PTF-NO is assumed.

MRT: MRT-YES specifies that the procedure member being created is a multiple requestor terminal procedure. MRT-YES is valid only for a reader-to-library copy of a procedure member (FROM-READER and LIBRARY-P must be specified).

HIST: Specifies whether OCL statements generated by a procedure member when the procedure is executed should be logged to the history file.

HIST-YES indicates that statements are logged to the history file. If HIST is not specified, HIST-YES is assumed.

HIST-NO indicates that OCL statements are not logged to the history file.

PDATA: PDATA-YES specifies that data (not parameters) can be passed on the INCLUDE OCL statement that causes the procedure to be executed. The data starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. The data is passed to the first program in the procedure on the first input operation from the requesting display station. Every MRT procedure has this attribute whether or not the attribute was selected when the procedure was created.

PDATA-NO specifies that parameters can be passed on the INCLUDE OCL statement that causes the procedure to be executed. If the PDATA parameter is not specified, PDATA-NO is assumed.

Note: For the ideographic version of the SSP, data passed on the INCLUDE OCL statement can contain ideographic characters; however, parameters passed on the INCLUDE OCL statement cannot contain ideographic characters.

BASIC: BASIC-YES specifies that the output file is a basic data exchange diskette file. BASIC-YES is valid only for a library-to-file copy in record mode when you are creating a new file. If the BASIC parameter is not specified, BASIC-NO is assumed.

Note: If ADD-YES is specified, BASIC-YES should not be specified. When you add records to a basic exchange file, do not specify BASIC-YES.

SVATTR: SVATTR-YES specifies that the attributes (SSP, MRT, PDATA, and HIST) should be retained when the member is copied. If the SVATTR parameter is not specified, SVATTR-NO is assumed.

For the DELETE statement:

LIBRARY: Specifies the type of member or members to be deleted:

- LIBRARY-S indicates source members.
- LIBRARY-P indicates procedure members.
- LIBRARY-O indicates load members.
- LIBRARY-R indicates subroutine members.
- LIBRARY-ALL indicates all member types (S, P, O, and R).

NAME: Specifies the name or names of members to be deleted:

- NAME-name specifies the name or names of members to be deleted.
- NAME-characters.ALL specifies that members whose names begin with the specified characters are to be deleted. Up to 7 characters can be specified. For example, NAME-PAY.ALL deletes all members whose names begin with PAY.
- NAME-ALL specifies that all members, except SSP members, are to be deleted when LIBRARY-ALL is specified, or that all members of the type specified by the LIBRARY parameter are to be deleted. If NAME-ALL and LIBRARY-ALL are specified, only non-SSP members are deleted. In order to delete SSP members, RETAIN-S must be specified.

LIBRNAME: The name of the library containing the member or members to be deleted. If LIBRNAME is not specified, the system library (#LIBRARY) is assumed.

OMIT: OMIT-name specifies that a member or members with the specified name are not to be deleted.

OMIT-characters.ALL specifies that all members whose names begin with the specified characters are not to be deleted. Up to 7 characters can be specified.

RETAIN: If RETAIN-S is specified, all SSP members identified by other parameters in the DELETE statement are deleted, unless both LIBRARY-ALL and NAME-ALL are specified.

If the RETAIN parameter is not specified, SSP members are not deleted.

For the COMPRESS statement:

LIBRNAME: The label of the library to be compressed. If LIBRNAME is not specified, the system library (#LIBRARY) is assumed.

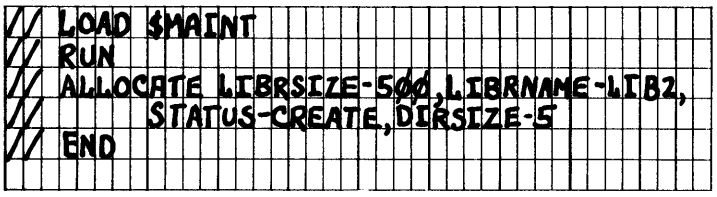
Control Statement Sequence

```
// LOAD $MAINT
[// FILE . . . ]   A FILE statement is required only when copying from a
                   library to a file or from a file to a library
                   or when performing a file-to-printer operation.
// RUN
{// ALLOCATE . . .
 // COPY . . .
 // DELETE . . .
 // COMPRESS . . .}
// END
```

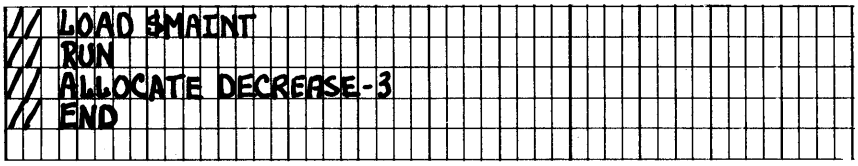
Examples

Allocate Examples

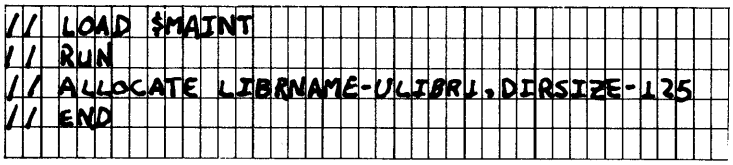
Allocate 500 blocks for a new library called LIB2. The directory should be 5 sectors long:



Decrease the size of the system library by 3 blocks:



Change the directory size:



Copy Examples

Reader-to-library operation:

Create a new system library P member called MRTPROC that is an MRT procedure:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-P,NAME-MRTPROC,
//     TO-F1,RECL-80,MRT-YES
        •
        •                STATEMENTS IN MRTPROC
        •
        •
// CEND
// END
```

Library-to-library operation:

Copy a load member called ACCT from the system library to a user library called NEWLIB:

```
// LOAD $MAINT
// RUN
// COPY FROM-F1,LIBRARY-0,NAME-ACCT,TO-NEWLIB
```

Library-to-file operation:

Copy (in sector mode) a procedure member named PAYROLL from a library called LIBR1 to a disk file labeled PAY. PAY is 30 sectors long and is to be retained permanently:

```
// LOAD $MAINT
// FILE NAME-PAY,UNIT-F1,BLOCKS-3,RETAIN-P
// RUN
// COPY FROM-LIBR1,TO-DISK,FILE-PAY,
//     NAME-PAYROLL,LIBRARY-P
// END
```

Copy (in record mode) a system library source member named SAM, with a record length of 80 bytes, to a file labeled BOB. BOB is 20 sectors long and is to be retained temporarily:

```
// LOAD $MAINT
// FILE NAME-BOB,UNIT-F1,BLOCKS-2,RETAIN-T
// RUN
// COPY FROM-F1,TO-DISK,FILE-BOB,RECL-80,
//    NAME-SAM,LIBRARY-S
// END
```

Copy (in record mode) a system library source or procedure member named SAM, with a record length of 80 bytes, to a basic data exchange diskette file labeled BOB. BOB is to be retained for 30 days. The vol-id of the diskette is 111222:

```
// LOAD $MAINT
// FILE NAME-BOB,UNIT-I1,RETAIN-30,PACK-111222
// RUN
// COPY FROM-F1,TO-DISK,FILE-BOB,RECL-80,
//    NAME-SAM,LIBRARY-S,BASIC-YES
// END
```

Copy (in sector mode) all members named PAYDAY from a library labeled ULIB to a disk file named PAYROLL. PAYROLL is 60 sectors long, starts at location 1500, and is a temporary file:

```
// LOAD $MAINT
// FILE NAME-PAYROLL,UNIT-F1,BLOCKS-6
//    LOCATION-1500,RETAIN-T
// RUN
// COPY FROM-ULIB,TO-DISK,FILE-PAYROLL,
//    NAME-PAYDAY,LIBRARY-ALL
// END
```

Copy (in record mode) system library source and procedure members named PAYDAY, with a record length of 120 bytes, to a disk file named PAY. PAY is 80 sectors long and is to be retained permanently:

```
// LOAD $MAINT
// FILE NAME-PAY,UNIT-F1,BLOCKS-8,RETAIN-P
// RUN
// COPY FROM-F1,TO-DISK,FILE-PAY,RECL-120,
//    NAME-PAYDAY,LIBRARY-ALL
// END
```

Copy (in sector mode) all system library members whose names begin with a dollar sign ($) to a file named UTIL. UTIL has a retention period of 90 days and is on a diskette whose volume ID is UTIL:

```
// LOAD $MAINT
// FILE NAME-UTIL,UNIT-I1,RETAIN-90,PACK-UTIL
// RUN
// COPY FROM-F1,TO-DISK,FILE-UTIL,
//    NAME-$.ALL,LIBRARY-ALL
// END
```

Copy (in record mode) source and procedure members from a library labeled LIB2 to a disk file named RPSD. Copy only those members with names beginning with RPU and with a record length of 80 bytes. RSPD is 50 sectors long and is classified as a permanent file:

```
// LOAD $MAINT
// FILE NAME-RPSD,UNIT-F1,BLOCKS-5,RETAIN-P
// RUN
// COPY FROM-LIB2,TO-DISK,FILE-RPSD,RECL-80,
//    NAME-RPU.ALL,LIBRARY-ALL
// END
```

Copy (in sector mode) all system library members whose names start with PA to a disk file named PAYR. PAYR is 60 sectors long and is classified as a temporary file:

```
// LOAD $MAINT
// FILE NAME-PAYR,UNIT-F1,BLOCKS-6,RETAIN-T
// RUN
// COPY FROM-F1,TO-DISK,FILE-PAYR,
//    NAME-PA.ALL,LIBRARY-ALL
// END
```

Add all system library members whose names begin with the characters PA to a disk file named PAYR. Omit those members whose names start with PAY:

```
// LOAD $MAINT
// FILE NAME-PAYR,UNIT-F1
// RUN
// COPY FROM-F1,TO-DISK,FILE-PAYR,NAME-PA.ALL,
//    LIBRARY-ALL,OMIT-PAY.ALL,ADD-YES
// END
```

File-to-library operation:

Copy library member(s) from a disk file named SAVED to a library labeled ULIB. If the file contains a member whose name and type are the same as a member currently existing in the library, the existing member is not replaced unless the operator replaces the member after being notified (by a message on the display screen) that a duplicate member exists (that is, RETAIN-P is assumed on the COPY control statement):

```
// LOAD $MAINT
// FILE NAME-SAVED,UNIT-F1
// RUN
// COPY FROM-DISK,TO-ULIB,FILE-SAVED
// END
```

Copy library member(s) into the system library from a diskette file named LMT. The operator is not notified if LMT contains a member whose name and type are the same as a member currently existing in the library:

```
// LOAD $MAINT
// FILE NAME-LMT,UNIT-I1
// RUN
// COPY FROM-DISK,TO-F1,RETAIN-R,FILE-LMT
// END
```

Copy library member(s) from a diskette file labeled FILE1 into a user library named ULIB1. Only members already existing in ULIB1 are copied from FILE1. The operator is not notified when the member(s) is replaced:

```
// LOAD $MAINT
// FILE NAME-FILE1,UNIT-I1
// RUN
// COPY FROM-DISK,TO-ULIB1,FILE-FILE1,
//      RETAIN-R,OMIT-NEW
// END
```

File-to-printer operation:

Print the type and name of all members in a diskette file that are labeled FILE1:

```
// LOAD $MAINT
// FILE NAME-FILE1,UNIT-I1
// RUN
// COPY FROM-DISK,TO-PRINT,FILE-FILE1
// END
```

Library-to-printer operation:

Print the status information in the system library directory area and print all entries in the directory:

```
// LOAD $MAINT
// RUN
// COPY FROM-F1,LIBRARY-ALL,NAME-DIR,TO-PRINT
// END
```

Delete Examples

Delete a non-SSP source member named PAYROLL from a library named LIBR1:

```
// LOAD $MAINT
// RUN
// DELETE LIBRARY-S,NAME-PAYROLL,LIBRNAME-LIBR1
// END
```

Delete from the system library all non-SSP members whose names begin with the characters INV:

```
// LOAD $MAINT
// RUN
// DELETE LIBRARY-ALL,NAME-INV.ALL
// END
```

Delete all non-SSP procedures from the system library:

```
// LOAD $MAINT
// RUN
// DELETE LIBRARY-P,NAME-ALL
// END
```

Compress Example

Compress the system library:

```
// LOAD $MAINT
// RUN
// COMPRESS
// END
```

$MGBLD–Create Message Member Utility Program

Function

The $MGBLD utility program creates a message load member in a library. A message load member is the special type of library load member from which the SSP retrieves the text associated with the message identification code (MIC) specified by a calling program. Input to $MGBLD is a library source member called a message source member.

Message Source Member

The message source member is put into a library like any other source member. For example, it might be copied (from reader to library or from library to library) by the $MAINT utility program; or, it might be entered by a program such as the source entry utility (SEU) described in the *Source Entry Utility Reference Manual.*

Three types of statements can be used in the message source member. Two of these are required: a *message control statement* and one or more *message text statements*. *Comment statements* are optional.

Message Control Statement

The message control statement specifies the name of the message load member to be created and whether a first- or second-level message load member is being created. The message control statement must be the first statement in a message source member. The format of the message control statement is:

```
load-name[,level]
```

load-name: The name to be given to the message load member created.

level: A value of 1 specifies that a first-level message load member is to be created (maximum of 75 characters per message text statement, excluding the MIC). A value of 2 specifies that a second-level message load member is to be created (maximum of 225 characters per message text statement, excluding the MIC). If the level parameter is not specified, 1 is assumed.

Message Text Statement

The format of the message text statement is:

```
MIC text
```

MIC: The message identification code. The MIC must be specified as a 1- through 4-character decimal number from 0 through 9999 and must be left-justified within the first four positions of the message text statement. The MIC must be in ascending order, relative to the MIC for the preceding message text statement, or the same MIC can be specified on consecutive message text statements to concatenate the text area. The number of statements that can be concatenated is restricted to the minimum number required to specify up to 225 characters of message text area.

text: The text area of the message text statement. The text area of each message text statement starts at position 6 and extends to the end of the message text statement (the length of the statement depends on the record length of the message source member). The text for a message is the series of characters from the start of the text area to the last nonblank character of the text area for the MIC. If text cannot be contained in one message text statement, it can be continued on following message text statement(s) specifying the same MIC. The text area on following statement(s) is appended to the text area of the preceding statement before trailing blanks for the total text specified are dropped. A message text of one blank will be generated for a message text statement containing a blank text area.

Comment Statement (optional)

The format of the comment statement is:

```
* comment
```

Comment statements must have an asterisk (*) as the first character. Comment statements can be interspersed with the message text statements. These statements, intended to provide additional information about the message, do not become part of the message load member.

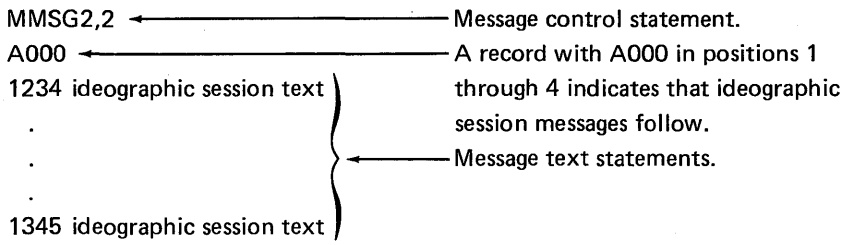
Considerations for the Ideographic Version of the SSP

For systems with ideographic character support, you can create:

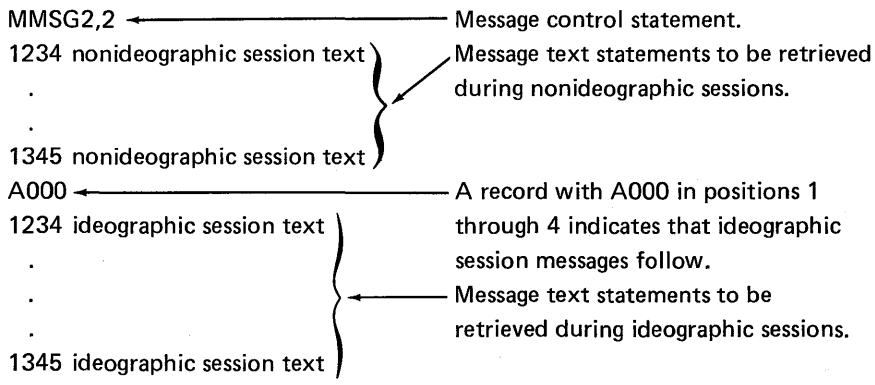
- A message member that contains messages to be retrieved during nonideographic sessions
- A message member that contains messages to be retrieved during ideographic sessions
- A message member that contains two groups of message texts: one group from which messages are retrieved during nonideographic sessions, and an alternative group from which messages are retrieved during ideographic sessions

If the message member contains only messages for nonideographic sessions, the message source member is created just as for systems without ideographic character support.

If the message member contains only messages for ideographic sessions, the source member should be as follows:



If the message member contains two groups of message texts, the source member should be as follows:



If the user is signed on to an ideographic session, the message is retrieved from the ideographic session portion of the message member. If the message is not found, the system retrieves the message from the nonideographic session portion of the message member.

If the job is submitted from the job queue or from a nonideographic session, the system retrieves the message from the nonideographic portion of the message member.

For the ideographic version of the SSP, the following special considerations exist when you concatenate message text statements:

- If a *shift in* character is in one of the last two positions in a record and if a *shift out* character is in the first position of the continuation record, the *shift in/shift out* pair and the intervening blank, if one exists, are deleted when the two records are concatenated.
- If the last two positions in the first record contain a *shift out* character followed by a *shift in* character, the *shift out/shift in* pair is deleted when the two records are concatenated.

$MGBLD is executed when the CREATE procedure is run.

Utility Control Statement Format

// MGBLD SOURCE–source name [,SSP–{YES / NO}] [,REPLACE–{YES / NO}]

[,LIBRARY–{library name / #LIBRARY}]

Parameters

SOURCE: Specifies the name of the library message source member that contains the message control statement and message text statements (MIC and text) required to create a message load member.

SSP: SSP-YES specifies that the message load member created is identified as an SSP member and cannot be deleted by the REMOVE procedure.

SSP-NO specifies that the message load member is *not* identified as an SSP member. If the SSP parameter is not specified, SSP-NO is assumed.

REPLACE: REPLACE-YES specifies that the message load member replaces an existing member with the same name.

REPLACE-NO specifies that an existing member with the same name is not replaced. An error message is displayed if an attempt is made to replace a member. If the REPLACE parameter is not specified, REPLACE-NO is assumed.

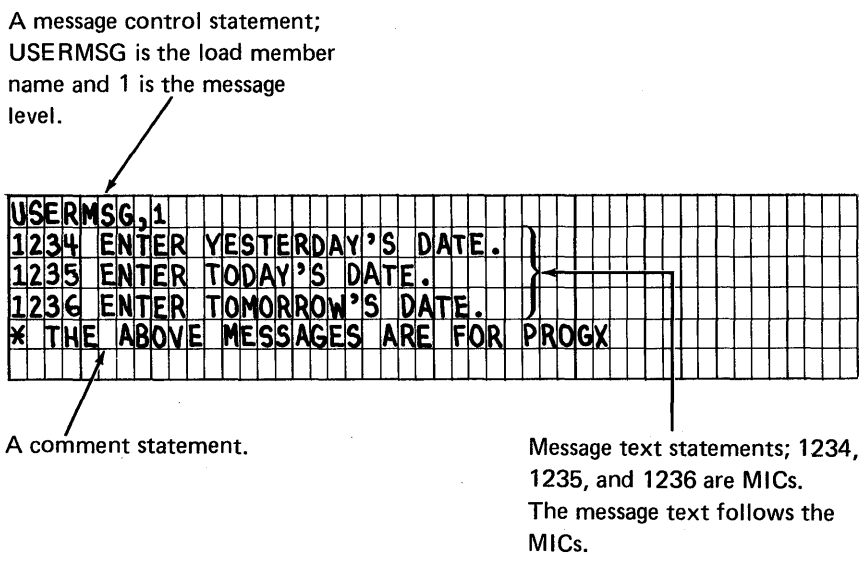
LIBRARY: Specifies the name of the library containing the message source member. The message load member created is also placed in the specified library. If SSP-YES is specified, the source member must be the system library (#LIBRARY). If LIBRARY is not specified, #LIBRARY is assumed.

Control Statement Sequence

```
// LOAD $MGBLD
// RUN
// MGBLD SOURCE-sourcename, . . .
// END
```

Example

Using $MAINT, enter from the keyboard into the system library a first-level message source member named USERMI. (See the description of the reader-to-library copy function of $MAINT.) USERMI should contain the following statements:



To create a message source member named USERMI from the above statements, you would enter from the keyboard:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-S,NAME-USERMI,
//      TO-F1,RETAIN-P,RECL-40
USERMSG,1
1234 ENTER YESTERDAY'S DATE.
1235 ENTER TODAY'S DATE.
1236 ENTER TOMORROW'S DATE.
* THE ABOVE MESSAGES ARE FOR PROGX
// CEND
// END
```

To create a message load member named USERMSG from the above source member (USERMI), you could use the CREATE procedure by entering:

```
CREATE USERMI
```

You could use the $MGBLD utility program by entering:

```
// LOAD $MGBLD
// RUN
// MGBLD SOURCE-USERMI
// END
```

$MMSP–Stop Monitoring Line Utility Program

Function

The $MMSP utility program stops the auto communications line monitoring function.

Note: The communications line being monitored could have been previously placed in auto monitor mode in one of three ways:
- By the STARTM procedure
- If an SSP-ICF BSC subsystem disabled
- If a batch BSC job terminated

$MMSP is executed when the STOPM procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

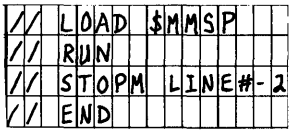
```
// LOAD $$MMSP
// RUN
// STOPM LINE#-{1|2|3|4}
// END
```

Parameter

LINE#: Specifies that auto monitor mode is discontinued for this line. Line numbers 1 through 4 are valid entries.

Example

To stop monitoring line number 2:



$MMST–Start Monitoring Line Utility Program

Function

The $MMST utility program starts automatic monitoring a BSC multipoint line that has the MLCA (Multiline Communications Adapter). The MLCA responds to all host polls or selects on a BSC multipoint line with a negative acknowledgement. A nonswappable BSC interrupt handler is not required in main storage for MLCA. Therefore, System/34 can be placed in the host's polling list without any tasks in System/34 being started until necessary.

Auto monitoring continues until stopped by the STOPM procedure, or until either a batch BSC job is started or an SSP-ICF BSC subsystem is enabled on that line.

$MMST is executed when the STARTM procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

// LOAD $MMST
// RUN
// STARTM LINE#– {1 2 3 4} ,CODE– {A E} ,STATION-station address
// END

Parameters

LINE#: The line number that is placed in auto monitor mode. Line numbers 1 through 4 are valid entries. This line must be a multipoint line.

CODE: Specifies transmission code. Valid entries are E (EBCDIC) and A (ASCII).

STATION: The 2-character, hexadecimal multipoint station address (either the poll or select sequence can be specified).

Example

To start monitoring line number 2:

```
// LOAD $MMST
// RUN
// STARTM LINE#-2,CODE-E,STATION-03
// END
```

$PACK–Disk Reorganization Utility Program

Function

The $PACK utility program causes all free space within the user area on disk to be accumulated into a single area. For a single disk configuration, free space is accumulated at the highest block numbers on the disk. For a multiple-disk configuration, the free space on both disks is accumulated at the disk juncture; that is, free space on disk A1 is accumulated at the highest block numbers on disk A1 and free space on disk A2 is accumulated at the lowest block numbers on disk A2.

$PACK can be run only from the display station assigned as the system console, and cannot be run (1) while any other jobs are being run, (2) while a user is signed on at any other display station, (3) while remote work stations are varied online, (4) while SSP-ICF subsystems are enabled, (5) while a communications line is enabled, or (6) while extended trace is active.

$PACK is executed when the COMPRESS procedure is run.

Notes:

1. If LOCATION was specified in the FILE statement for a file moved by $PACK, the location specified is not valid after $PACK moves the file. Use the CATALOG procedure to display the VTOC entries for files moved by $PACK to determine the new location of the files.
2. If you want to (1) accumulate free space on only one disk in a multiple-disk environment or (2) accumulate free space immediately following the system library accumulate free space in the highest block numbers on disk A2, you can use the $FREE utility program described earlier in this chapter.

Utility Control Statement Format

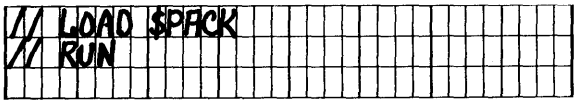
Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PACK
// RUN
```

Example

Compress the free space on disk:



$PDSR–Primary SDLC Retry Count Reset Utility Program

Function

$PDSR utility program displays the configured Primary SDLC error retry count values for each communication line and allows the values to be changed. The value is the number of multiples-of-7 retries that Primary SDLC uses to attempt to contact a secondary station. If the secondary station does not respond during the specified number of retries, a permanent time-out error is reported.

Some intelligent modems such as the IBM 3865 require more time to equalize than the error retry count allows. Thus, a permanent time-out error may be reported when the modem is equalizing (not a permanent error situation). Increasing the error retry count value from the default (a value of 1, or 7 retries) prevents a permanent time-out error from occuring.

It is recommended that the default value of 1 be used unless a problem has been identified. This value only affects the operation of Primary SDLC and its associated SNA task (Remote Work Station Support, SNA Peer Support–Primary only, System/34 Finance Support, or Link Station Test).

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PDSR
// RUN
```

Example

To display and modify the Primary SDLC error retry count values enter:

```
// LOAD $PDSR
// RUN
```

The following screen is displayed that allows you to modify the error retry count values. Remember that the value entered for a line number specifies the number of multiples of 7. For example, a value of 2 specifies 14 retries.

```
                PRIMARY SDLC ERROR RETRY COUNTS


                LINE        VALUE

                  1           1

                  2           1

                  3           1

                  4           1


The value is the number of multiples of 7 retries used by Primary SDLC
to contact a secondary station.  If the secondary station does not
respond within the specified number of retries, a permanent timeout
error is reported.  Valid values are 1-5.
```

$PNLM–Phone List Utility

Function

The $PNLM utility program provides a way to create, update, or delete one or more phone number lists, each containing up to 120 phone numbers. A phone list contains the phone number or numbers you want the System/34 to call automatically. Each list is stored as a load member in either #LIBRARY or a user library, and as many lists can be defined as there is space available. Refer to the *Interactive Communications Feature Reference Manual* and to the *Data Communications Reference Manual* for additional information on how the phone list can be used.

The $PNLM utility program is executed when the DEFINEPN procedure is run.

A source member containing the phone list is required for the DEFINEPN procedure. If directed to do so, the DEFINEPN procedure first calls SEU so that you can create the source member. If you do not use SEU to create this source member, a source member containing the phone list must be given to the $PNLM utility program.

The phone list source member contains the following information for each phone number:

- The phone number, including separator (SEP) and end of number (EON) characters. The phone number can be 1 through 22 characters, including any SEP or EON characters. The phone number must be placed in columns 1 through 22 and must be left-justified.

  An apostrophe (X'7D') represents an SEP character. The SEP character, if included, is used with some autocall units to indicate that the hardware waits for the next dial tone before continuing to dial the rest of the phone number. For example, to call a number outside of your private exchange, you might have to dial 9 and then wait for a dial tone before dialing the rest of the phone number. In this case, you would put a 9 in column 1, an apostrophe (the SEP character) in column 2, and the rest of the number in columns 3 through 22.

  An asterisk (X'5C') represents an EON character. The EON character, if included, must be the last character of the phone number. The EON character tells the autocall unit that the preceding characters make up the entire number as it is to be dialed.

  Note: Use of the separator (SEP) and end of number (EON) characters may be restricted by the autocall unit you are using. Either or both of these characters may not be allowed. Some countries may require that the SEP or EON characters appear in a certain order within a phone list. The user should be aware of such restrictions when creating a phone list. Check with your autocall supplier to find out which characters are allowed and what special requirements your autocall unit may have.

- A retry count specifying the total number of times an attempt to call this number should be made. Allowable values are 1 through 255; the default value is 1. The retry count must be placed in columns 23 through 25, right-justified, and zerofilled.

  Note: Many countries limit the number of attempts to the same telephone number in the case of unsuccessful calls. Where this limitation exists, it is typically in the range of 2 through 12 attempts. You should contact the PTT (telephone company) for the maximum value that applies.

- A timer (or connection) wait value indicating the time to wait for the distant station to be connected after the last digit has been dialed. Allowable values are 3 through 126 seconds; the default value is 20 seconds. The timer wait value must be placed in columns 26 through 29, right-justified, and zerofilled.

  Note: You should be aware of any PTT (telephone company) or autocall unit restrictions on the timer wait value. For example, the timer wait value for autocall units conforming to the CCITT Standard V.25 is typically restricted to the range of 10 through 40 seconds.

Using the source member, the $PNLM utility program prints each entry in the phone list, flagging any statements in error. A load member is built in the user-specified library if no errors occur.

Utility Control Statement Format

// DEFINEPN INAME–source member name

[,INLIB–{,source library name / #LIBRARY}] [,OUTNAME–{,load member name / source member name}]

[,OUTLIB–{,load library name / #LIBRARY}] ,[MODE–{DELAY / NODELAY}]

Control Statement Sequence

// LOAD $PNLM
// RUN
// DEFINEPN INAME-source member name,INLIB-source member library,
// OUTNAME–load member name, OUTLIB–load library name

[,MODE–{DELAY / NODELAY}]

// END

Example

To create a phone list called DMSLIST to be stored in a library called DRMLIB:

```
// LOAD $PNLM
// RUN
// DEFINEPN INAME-DMSLIST,OUTLIB-DRMLIB
// END
```

$POST–Data Exchange Utility Program

Function

The $POST utility program converts a disk file to a basic data exchange file on diskette, converts a special E format diskette file to a sequential or indexed disk file, adds a special E format diskette file to a sequential disk file, or displays the contents of a special E format diskette file. All diskette files that are input for $POST must be in the 128-byte format for diskette 1 diskettes (one-sided diskettes), in the 256-byte format for diskette 2D diskettes (two-sided double-density diskettes), or special E formatting diskette 2D diskettes. All diskette files created by $POST are in the 128-byte (for diskette 1) or the 256-byte (for diskette 2D) basic data exchange format.

For information about diskette formats, see Appendix C.

When a special E format diskette file is added to an existing disk file, the records in the diskette file are either truncated or padded with hexadecimal zeros (hex 00) to conform to the record length of the disk file. When a new disk file is created from a special E format diskette file, the record length of the disk file is set to that of the diskette file. When a new basic data exchange diskette file is created from a disk file, the record length of the diskette file is set to that of the disk file or to 128 (for diskette 1) or 256 (for diskette 2D), whichever is smaller.

$POST processes records consecutively during file conversion. If input for $POST is an indexed disk file, records are read sequentially by key.

$POST is executed when the POST procedure is run.

Utility Control Statement Formats

To create a basic data exchange diskette file from a disk file or to convert a special E format diskette file to a disk sequential file:

```
[// TRANSFER]
 // END
```

To add the data in a special E format diskette file to a disk sequential file or to add data in a disk file to a basic data exchange diskette file:

```
// TRANSFER ADD-YES
// END
```

To create an indexed file on the disk from a special E format diskette file:

```
// TRANSFER [ADD-NO],KEYLEN-value,KEYLOC-value
// END
```

To create a sequential disk file from a special E format diskette file and to detect the special end-of-data condition generated by the 5260 Retail System:

```
// TRANSFER EOD-YES
// END
```

To display the contents of a special E format diskette file:

```
// DISPLAY [FROM-{value 1 / 1}][,TO-{value 2 / 8,388,607}]
// END
```

Parameters

For the POST statement:

ADD: ADD-YES specifies that records in the input file are added to the output file. The first record from the input file is placed after the last record in the existing output file. If the input file (specified on the COPYIN FILE statement) is on a diskette(s), records are added to the disk file until the end of either the diskette file or the disk file is reached. However, the add operation is not started unless the space available on the disk file is large enough to contain the records in the file segment on the diskette currently inserted. If the input file is on disk, then records are added to the diskette file until the end of the disk file is reached. The output diskette file must be physically the last file on the diskette.

KEYLEN: The length of the record keys when an indexed file is to be created on the disk. The value can be any decimal number from 1 through 29.

Note: KEYLEN must be specified with KEYLOC, and the sum of their values must not exceed the record length plus 1 or the physical record length of the basic data exchange file plus 1, whichever is less.

KEYLOC: The relative displacement of the start position of the record key in the records. The value can be any decimal number from 1 through 128 for diskette 1, or 1 through 256 for diskette 2D.

Note: KEYLOC must be specified with KEYLEN, and the sum of their values must not exceed record length plus 1 or the physical record length of the basic data exchange file plus 1, whichever is less.

EOD: EOD-YES specifies that the special end-of-data condition (see note that follows) is to be detected when a special E format diskette file is copied or added to a disk file.

EOD-NO specifies that the special end-of-data condition (see note that follows) is to be ignored when a special E format diskette file is copied or added to a disk file. If the EOD parameter is not specified, EOD-NO is assumed.

Note: The 5260 Retail System indicates an end-of-data condition with a record of blanks. If a basic exchange diskette file is being copied and EOD-YES is specified, data may be lost.

For the DISPLAY statement:

FROM: Value 1 specifies the first record to be displayed. If FROM is not specified, the first record in the file is assumed.

TO: Value 2 specifies the last record to be displayed. If TO is not specified, the last record in the file is assumed. The maximum value that can be specified is 8388607.

Control Statement Sequence

```
// LOAD $POST
// FILE NAME–COPYIN,UNIT–{I1 / F1},LABEL–from file label
[// FILE NAME–COPYO,UNIT–{I1 / F1},LABEL–to file label . . . ]
// RUN
[// TRANSFER]
[// DISPLAY ]
// END
```

Notes:

1. If a new disk file is to be created from a multivolume diskette file, the COPYO FILE statement must be given, and the required RECORDS (or BLOCKS) parameter must be large enough to contain the entire diskette file.
2. If a new disk file is to be created from a single-volume diskette file (with the same space requirement as the diskette file), do not specify the COPYO FILE statement.
3. If a new disk file larger than the diskette file is to be created, the COPYO FILE statement must be specified with the required RECORDS (or BLOCKS) parameter.
4. If a file is being created on diskette, the COPYO FILE statement with a PACK parameter is required.

Example

Create a basic data exchange diskette file (JOEB1) from a disk file (JOE):

```
// LOAD $POST
// FILE NAME-COPYIN,UNIT-F1,LABEL-JOE
// FILE NAME-COPYO,UNIT-I1,LABEL-JOEB1,PACK-9
// RUN
// TRANSFER
// END
```

$PRES–Resource Security Utility Program

Function

The $PRES utility program is used to activate or de-activate resource (file and library) security and to define, modify, or delete the resource security file. Other security officers designated by the master security officer can use the $PRES utility to modify the resource security file. The $PRES utility can be run only from the system console when it is being used as a display station.

The *Installation and Modification Reference Manual* describes the resource security function and the user access codes that can be assigned for files and libraries that are to be protected. That manual also describes the displays issued by the $PRES utility program and the operator actions required.

$PRES is executed when the PRSRC procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PRES
// RUN
```

$PRLT–Security Report Utility

Function

The $PRLT utility program is used to select the type or format of the security file listing. A temporary file is used to select or sort records from the security file. The listings are printed on the syslist printer.

The PRLIST procedure can be used to select the type or format of the security file listing.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PRLT
// RUN
```

$PRMN–Menu Security Utility Program

Function

The $PRMN utility program is used to assign a menu to a user and, optionally, to restrict that user to requesting jobs from only that menu. The $PRMN utility can be run only by the master security officer or another designated security officer and only from the system console when it is being used as a display station.

The *Installation and Modification Reference Manual* describes the menu security display and the operator actions required.

$PRMN is executed when the PRMENU procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PRMN
// RUN
```

$PROF–Password Security Utility Program

Function

The $PROF utility program is used by the master security officer to activate or deactivate password or badge security and to define, modify, delete, or increase the size of the password security file. Other security officers designated by the master security officer can use the $PROF utility program to modify the password security file. The $PROF utility program can be run only from the system console when it is being used as a display station.

CAUTION

When the size of the password security file is changed, the file is reallocated. The contents of the file are lost. Therefore, you should save the contents of the file on a diskette before reallocating the file. You can use the PRSAVE procedure to save the contents of the password security file. You can use the PRESTOR procedure to place the contents in the newly allocated system security file.

The *Installation and Modification Reference Manual* describes the password security file and the operator classes that can be defined. That manual also describes the displays issued by the $PROF utility program and the operator actions required.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PROF
// RUN
```

$PRON–Resource Owner Utility Program

Function

The $PRON utility program is used by the owner of a file or library to display the resource security file records for files or libraries for which the operator is classified as an owner. The operator can change the resource security file records for any file or library which he owns. The $PRON utility program can be run only if password security is active and if the resource security file exists.

For a description of the displays issued by the $PRON utility program and of the operator actions required, see the description of the PRSRCID procedure in Chapter 2. For information about the resource security function and the user access codes that can be assigned, see the *Installation and Modification Reference Manual*.

The $PRON utility is executed when the PRSRCID procedure is run.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $PRON
// RUN
```

$PRST–Security File Restore Utility Program

Function

The $PRST utility program copies the system security files from diskettes onto the disk. (The $PRSV utility can be used to copy the system security files to diskettes.) $PRST can be run only by the master security officer and only if the following requirements are met:

- The $PROF utility program was used to select password security.
- The system security files are already allocated.
- The current security files are large enough to hold the files being copied from diskette ($PROF and $PRES can be used to increase the size of the current security files).
- The utility is run from the system console and no other jobs are running.

If both the password security file and the resource security file are defined on disk and in the diskette file, both files are restored. If only the password security file is defined on disk, only the password security file will be restored even though the diskette file may also contain a copy of a resource security file. If both files are defined on disk but only the password security file is in the diskette file, a warning message is displayed. The operator can then choose to restore just the password security file or to cancel $PRST without restoring either file.

For information about the contents of the system security files and the operator classes that can be assigned, see the description of security in the *Installation and Modification Reference Manual.*

Utility Control Statement Format

Utility control statements are not used.

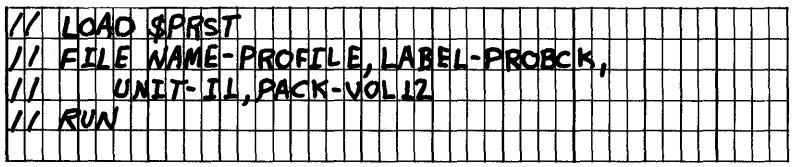
Control Statement Sequence

```
// LOAD $PRST
// FILE NAME-PROFILE,LABEL-file label,UNIT-I1,
//      PACK-vol-id
// RUN
```

Note: On the FILE OCL statement, the LABEL parameter identifies the label of the diskette file that contains the copy of the system security files and the PACK parameter identifies the volume ID of the diskette that contains the file.

Example

Copy the system security files back to disk from a diskette file labeled PROBCK. The diskette volume ID is VOL12:



$PRSV–Security File Save Utility Program

Function

The $PRSV utility program saves the system security files on diskette(s). $PRSV can be run only if the $PROF utility program was used to select the password security function. In addition, $PRSV can be run only by the system master security officer and only from the system console when it is being used as a display station.

If a resource security file is defined on disk, both the password security file and the resource security file are copied to the diskette file. If a resource security file is not defined on disk, only the password security file is copied to the diskette file.

Note: For information about the contents of the system security files and the operator classes that can be assigned, see the description of security in the *Installation and Modification Reference Manual.*

Utility Control Statement Format

Utility control statements are not used.

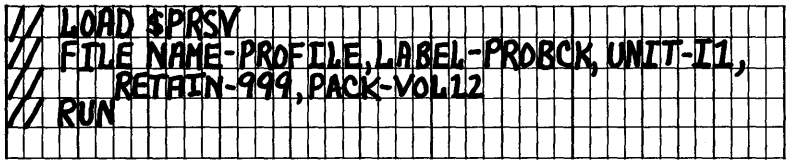
Control Statement Sequence

```
// LOAD $PRSV
// FILE NAME-PROFILE,LABEL-file label,UNIT-I1,
//       RETAIN-999,PACK-vol-id
// RUN
```

Note: On the FILE OCL statement, the LABEL parameter identifies the label of the diskette file to contain the copy of the system security files and the PACK parameter identifies the volume ID of the diskette to be used.

Example

Copy the system security files to a diskette file labeled PROBCK. The diskette volume identification is VOL12:



$RENAM–File Rename Utility Program

Function

The $RENAM utility program changes the label of a specified permanent or temporary disk file or library. Only the label is changed; the creation date of the file is not changed. $RENAM ensures that no disk file or library already exists with the new label. (A library can be renamed only if it was created by release 3 or a later release of System/34, or processed by the rebuild function during IPL.)

$RENAM is executed when the RENAME procedure is run.

Utility Control Statement Format

```
// RENAME LABEL–label1,NEWLABEL–label2 [,DATE– {mmddyy}]
                                              {ddmmyy}
                                              {yymmdd}
// END
```

Parameters

LABEL: Specifies the current label of the file whose label is to be changed.

NEWLABEL: Specifies the new label to be assigned to the file.

DATE: Specifies the creation date of the file whose label is to be changed. The date must be specified in the format of the session date. If DATE is not specified and more than one file exists with the label specified by the LABEL parameter, the file with the latest creation date is relabeled.

Control Statement Sequence

```
// LOAD $RENAM
// RUN
[// RENAME . . .
// RENAME . . .
     •
     •
     •
// RENAME . . .]
// END
```

Note: $RENAM does not prevent more than one label change for a given file.

Example

Change the label of the last file created that has the label FILEA. The new label should be FILEB:

```
// LOAD $RENAM
// RUN
// RENAME LABEL-FILEA,NEWLABEL-FILEB
// END
```

$RSTRT–Program Restart Utility Program

Function

The $RSTRT utility program restarts a program after a system failure, if the program was checkpointed before the failure occurred. If you do not want to restart the program, the $RSTRT utility removes the checkpoint record file and checkpoint active status from the system.

For further information about checkpoint/restart, its restrictions and its programming considerations, see the *Concepts and Design Guide*.

If resource security is active, the operator who executes the $RSTRT utility program must be on the list of users for the user files and libraries for the job step being restarted as well as all job steps that follow the checkpointed job step.

Utility Control Statement Format

// CHKPT LABEL–label [,RESTART–{NO / YES}]

Parameters

label: The label of the file that contains the checkpoint records.

RESTART: RESTART-YES specifies that the information saved in the latest checkpoint record should be restored and the task restarted. RESTART-YES is the default.

RESTART-NO specifies that the checkpointed task is not to be restarted, and the checkpoint active status is to be removed from the system.

Control Statement Sequence

```
// LOAD $RSTRT
// RUN
// CHKPT LABEL–label [,RESTART–{NO / YES}]
// END
```

Example

Restart a checkpointed task contained in the checkpointed file CHKFIL:

$SETCF–Set Configuration Utility Program

Function

The $SETCF utility program changes items in the display station configuration record or the communications configuration record for the requesting display station. Each display station that is defined as a command display station during system configuration has an associated display station configuration record and communications configuration record. Information is recorded in the display station configuration records during system configuration (see the *Installation and Modification Reference Manual* for more information). After system configuration, a display station configuration record or communications configuration record can be changed by $SETCF. In general, items that can be changed by $SETCF include display station environment items, BSC environment items, items that override user program BSC specifications, and SDLC environment items.

The following display station environment items can be changed by $SETCF:

- Number of lines printed per page
- Print belt image
- Session date format
- Session date
- Region size
- Library assigned to the display station
- Printer for display station output
- Printer forms number

The following BSC environment items can be changed by $SETCF (for information about BSC, see the *Data Communications Reference Manual*):

- Bits-per-second rate
- Business machine clocking
- Error retry count
- Switched network backup
- IBM modem test
- Non-US answer tone
- Line number

The following BSC specifications can be overridden by $SETCF (for information about BSC, see the *Data Communications Reference Manual*):

- Local BSC switched line ID
- Remote BSC switched line ID
- Tributary station address
- Line type
- Switch type
- Wait time
- Blank compression
- Record separator
- 3740 multiple file support

The following SDLC environment items can be changed by $SETCF (for information about SDLC, see the *Data Communications Reference Manual*):

- Bits-per-second rate
- Business machine clocking
- Line number
- Switched network backup
- IBM modem test
- Non-US answer tone

The following SDLC line items can be set by $SETCF (for information about SDLC, see the *Data Communications Reference Manual*):

- Secondary station address
- Line type
- Switch type
- Line number
- Exchange ID

$SETCF does not affect SDLC items for remote work stations or for SSP-ICF.

$SETCF is executed when the ALTERBSC, ALTERSDL, OVERRIDE, SPECIFY, or SET procedure is run.

Utility Control Statement Formats

To change the display station environment:

```
// SETCF [LINES–number] [,IMAGE–{YES}] [,FORMAT–{MDY}]
                            {NO }          {DMY}
                                           {YMD}

[,DATE– {mmddyy}] [,LIBRARY–{#LIBRARY}]
        {ddmmyy}             {name   }
        {yymmdd}             {0      }

[,PRINTER–{SYS }] [,FORMSNO–nnnn] [,RGSIZE–nn]
          {ws-id}
```

Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. Changes made by the SETCF statement remain in effect until:
   a. The items are changed again by the $SETCF utility program or the SET procedure.
   b. The system library is reloaded.
   c. The system is configured again.

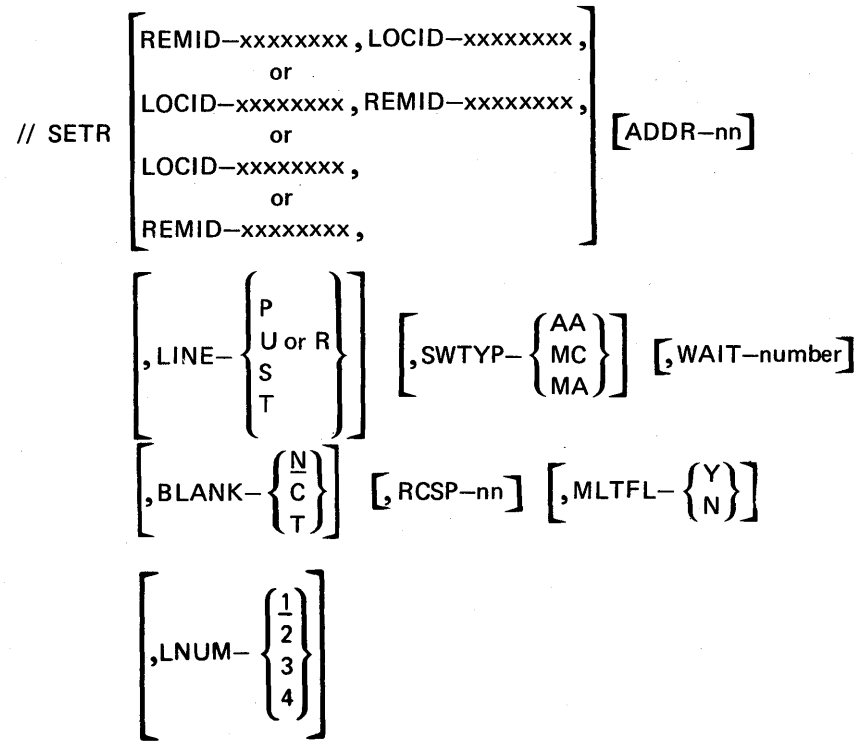
To change the BSC environment:

```
// SETB [BRATE-{F}] [,CLOCK-{Y}] [,ERC-number]
                {H}       {N}
     [,SLINE-{Y}] [,TEST-{Y}] [,TONE-{Y}]
             {N}        {N}       {N}
     [,LNUM-{1}]
            {2}
            {3}
            {4}
```

Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If any parameters are omitted, the corresponding values in the display station communications configuration record are not changed.
3. Changes made by the SETB statement remain in effect until:
   a. The items are changed again by the $SETCF utility program or the ALTERBSC procedure.
   b. The system library is reloaded. When the system library is reloaded:
      - CLOCK, TEST, and TONE are set to the values specified during system configuration.
      - The switched network backup line is not used (SLINE-N).
      - The full rated speed of the modem is used (BRATE-F).
      - The error retry count (ERC) for the program is used.
   c. The system is configured again.
4. If the $SETCF utility is run in inquiry mode, configuration changes apply only until the inquiry ends.
5. When you use the SSP-interactive communications feature BSC (SSP-ICF BSC) support, the $SETCF utility must be run on the system console. In this case the parameters are fixed; that is, once you enable a subsystem, the changes made by $SETCF will not go into effect until you disable the subsystem and, then, enable it again. The IMS/VS IRSS parameter does not apply to SSP-ICF BSC.
6. When you use the SSP-ICF BSC support and the system library is reloaded, the ERC defaults to 7. It is possible, however, to override the ERC value with the $IENBL utility.

To override BSC specifications:



Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If LINE-S is specified, SWTYP must be specified; if SWTYP is specified, LINE-S must be specified unless SLINE-Y was previously specified on an ALTERBSC procedure command or SETB utility control statement. Although SWTYP is required with LINE-S, SSP-ICF BSC does not use its value.
3. Changes made by the SETR statement remain in effect until:
   a. The items are changed by the $SETCF utility program or the OVERRIDE procedure.
   b. The system library is reloaded. When the system library is reloaded:
      – LINE is set to the value specified during system configuration.
      – REMID, LOCID, ADDR, SWTYP, WAIT, and RCSP are all set to the values specified in the user program.
      – Blank compression is not used (BLANK-N).
      – 3740 multiple files are not used (MLTFL-N).
4. The LOCID and REMID parameters are reset to 0 when the next SETR control statement that does not contain these keywords is executed.

To change the SDLC environment:

```
// SETS [BRATE-{F|H}] [,CLOCK-{Y|N}] [,LNUM-{1|2|3|4}]
     [,SLINE-{Y|N}] [,TEST-{Y|N}] [,TONE-{Y|N}]
```

Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If any parameters are omitted, the corresponding values in the display station communications configuration record are not changed.
3. For information on making a switched line connection for an inoperable point-to-point line to a remote work station, see the description of data communications problem determination in the *Operator's Guide*.

To change the SDLC line parameters:

```
// SETP [ADDR-nn] [,LINE-{P|S|T}] [,SWTYP-{AA|MC|MA}]
     [,LNUM-{1|2|3|4}] [,ID-nnnnn]
```

Notes:

1. Although each parameter is optional, at least one parameter must be specified.
2. If LINE-S is specified, SWTYP must be specified. If SWTYP is specified, LINE-S must be specified. If LINE-P or LINE-T is specified, SWTYP defaults to 0.
3. For information on making a switched line connection for an inoperable point-to-point line to a remote work station, see the description of data communications problem determination in the *Operator's Guide*.
4. When using the SSP-ICF SNA support, the $SETCF utility must be run on the system console before an SSP-ICF SNA subsystem is enabled if the line type to be used by that subsystem is different than the line type specified during hardware configuration.

Parameters

For the SETCF statement:

LINES: Specifies the number of lines to be printed per page. The maximum number is 112, and the minimum number is 1.

For information about how the value specified determines the actual number of lines printed per page, see the description of the FORMS OCL statement in Chapter 1.

IMAGE: Specifies whether the print belt image is to be changed.

- IMAGE-YES specifies that the print belt image is to be changed. If IMAGE-YES is specified, an IMAGE OCL statement must have preceded the RUN OCL statement for $SETCF.

  Note: The print belt image does not change again until it is changed by $SETCF or until the system is reconfigured.

- IMAGE-NO specifies that the print belt image is not changed. If IMAGE is not specified, IMAGE-NO is assumed.

FORMAT: Specifies the format of the session date. MDY specifies month-day-year, DMY specifies day-month-year, and YMD specifies year-month-day.

DATE: Specifies the session date.

LIBRARY: Specifies the name of the user library assigned to the display station. If 0 or #LIBRARY is specified, no user library will be active.

PRINTER: Assigns a printer for output from the display station. The assignment can be overridden for the session by a PRINTER OCL statement.

- PRINTER-SYS assigns the system printer for output from the display station.

- PRINTER-ws-id specifies the work station ID of the printer to be assigned for output from the display station.

FORMSNO: Specifies the printer forms number to be used for work station output. The printer forms number can be overridden for the session by a SYSLIST or PRINTER OCL statement.

RGSIZE: Specifies the default region size in number of K bytes. (If an odd number is specified, the SSP uses the next higher even number of K bytes as the region size.) The default region size can be overridden by the REGION OCL statement.

For the SETB statement:

BRATE: BRATE-F specifies that the full rated speed of the modem should be used. BRATE-H specifies that only half the rated speed of the modem should be used.

CLOCK: CLOCK-Y specifies that System/34 must provide the programmed clocking facility. CLOCK-N specifies that the modem has the clocking facility.

ERC: The number of error retries to be attempted. Any decimal number from 1 through 255 can be specified. If the system library is reloaded, the error retry count from the executing program is used.

SLINE: SLINE-Y specifies that a switched network backup line is used as backup (standby) for the nonswitched primary line. SLINE-N specifies that the switched network backup line is not used.

Note: After specifying SLINE-Y, the operator can use the OVERRIDE procedure to specify manual call, manual answer, or automatic answer. Otherwise, the connection defaults to manual call or manual answer depending upon the first line operation performed by the user program. If the first operation is a transmit operation, manual call is assumed; if the first operation is a receive operation, manual answer is assumed.

TEST: TEST-Y specifies that an IBM modem is being used. The automatic wrap test includes modem testing when a permanent error occurs.

TEST-N specifies that a non-IBM modem is being used. The automatic wrap test does not include modem testing.

TONE: TONE-Y specifies that a non-US special tone is required for manual answer and automatic answer. TONE-N specifies that a non-US special tone is not required.

LNUM: LNUM specifies the line number to which the parameters apply. If LNUM is not specified, LNUM-1 is assumed.

IRSS: IRSS-Y specifies that the *IBM System/34 IMS/VS IRSS Facility PRPQ*, 5799-AXK, is being used. IRSS-N specifies that the IMS/VS IRSS Facility is not being used.

For the SETR statement:

LOCID-xxxxxxxx,: The hexadecimal equivalent of the local station switched line ID. Either 2, 4, 6, or 8 hexadecimal digits can be specified. If this station's ID is longer than 4 characters, it must be specified in the user program. If LOCID is not specified, the value in the user program is used.

Note: If either the LOCID or REMID parameter is specified, it must appear before any other parameters.

REMID-xxxxxxxx,: The hexadecimal equivalent of the remote station switched line ID. Either 2, 4, 6, or 8 hexadecimal digits can be specified. If the remote station's ID is longer than 4 characters, it must be specified in the user program. If REMID is not specified, the value in the user program is used.

Note: If either the LOCID or REMID parameter is specified, it must appear before any other parameters.

ADDR: The hexadecimal equivalent of one of the pair of tributary station addressing characters. If ADDR is not specified, the value in the user program is used.

For the polling and addressing characters for System/34 tributary stations, see Appendix G.

LINE: Specifies the line facility:

- LINE-P specifies point-to-point nonswitched line.
- LINE-U or LINE-R indicates that the line type specified in the user program should be used.
- LINE-S specifies point-to-point switched line.
- LINE-T specifies tributary station on a multipoint line.

If LINE is not specified, the value in the communications configuration record remains unchanged.

SWTYP: Switch type:

- SWTYP-AA specifies that if a switched line (LINE-S or SLINE-Y) is specified and the modem is in automatic answering mode, then System/34 automatically answers the call.
- SWTYP-MC specifies that if a switched line (LINE-S or SLINE-Y) is specified, the System/34 operator initiates the call manually.
- SWTYP-MA specifies that if a switched line (LINE-S or SLINE-Y) is specified, the System/34 operator answers the call manually.

WAIT: The number of seconds that BSC will wait with no message being sent or received before it indicates a permanent error occurred. Any decimal number from 1 through 999 can be specified. If this parameter is omitted, zero is written in the display station communication configuration record. Zero indicates that the value for wait time is taken from the user program.

BLANK: Specifies if 3780-format blank compression or truncation is used:

- BLANK-N specifies no blank compression or truncation. If BLANK is not specified, BLANK-N is assumed.
- BLANK-C specifies blank compression.
- BLANK-T specifies blank truncation.

RCSP: RCSP specifies the record separator.

If RCSP is not specified and if the user program is an RPG II program, the following statements are true:

- If BLANK-C or BLANK-T is specified, a record separator of hex 1E is used.
- If BLANK-C or BLANK-T is not specified, no record separator is used.

If RCSP is not specified and if the user program is an assembler program, the following statements are true:

- If a record separator is specified in the user program, that record separator is used.
- If BLANK-C or BLANK-T is specified and a record separator is not specified in the user program, a record separator of hex 1E is used.
- If BLANK-C or BLANK-T is not specified and a record separator is not specified in the user program, no record separator is used.

MLTFL: MLTFL-Y specifies that multiple 3740 files can be transmitted or received with null records separating the files.

MLTFL-N specifies that 3740 multiple files are not supported. If MLTFL is not specified, MLTFL-N is assumed.

LNUM: LNUM specifies the line number to which the parameters apply. If LNUM is not specified, LNUM-1 is assumed.

For the SETS statement:

BRATE: BRATE-F specifies that the full rated speed of the modem is used. BRATE-H specifies that only half the rated speed of the modem is used.

CLOCK: CLOCK-Y specifies that System/34 must provide the programmed clocking facility. CLOCK-N specifies that the modem must provide the clocking facility.

LNUM: Specifies the line number to which the parameters apply. If LNUM is not specified, LNUM-1 is assumed.

SLINE: SLINE-Y specifies that a switched network backup line is used as backup (standby) for the nonswitched primary line. SLINE-N specifies that the switched network backup line is not used.

Note: After specifying SLINE-Y, the operator can use the SPECIFY procedure to specify manual call, manual answer, or automatic answer.

TEST: TEST-Y specifies that an IBM modem is being used. The automatic wrap test includes modem testing when a permanent error occurs. TEST-N specifies that a non-IBM modem is being used. The automatic wrap test does not include modem testing when a permanent error occurs.

TONE: TONE-Y specifies that a non-US special tone is required for manual answer and automatic answer. TONE-N specifies that a non-US special tone is not required when a permanent error occurs.

For the SETP statement:

ADDR: Specifies the hexadecimal SDLC address of the secondary station.

LINE: Specifies the line facility:

- LINE-P specifies point-to-point nonswitched line.
- LINE-S specifies point-to-point switched line.
- LINE-T specifies secondary station on a multipoint line.

SWTYP: Specifies the switch type:

- SWTYP-AA specifies that if a switched line (LINE-S) is specified and the modem is in automatic-answering mode, System/34 automatically answers the call.
- SWTYP-MC specifies that if a switched line (LINE-S) is specified, the System/34 operator initiates the call manually.
- SWTYP-MA specifies that if a switched line (LINE-S) is specified, the System/34 operator answers the call manually.

If SWTYP is not specified, the value from the configuration record is used.

LNUM: Specifies the line (1 or 2) to which the parameters in this procedure apply. If not specified, LNUM defaults to line 1.

ID: Specifies the 5-character hexadecimal number used in an exchange of identification between the host system and the System/34 secondary SDLC station.

Control Statement Sequence

```
// LOAD $SETCF
[// IMAGE . . .]
```

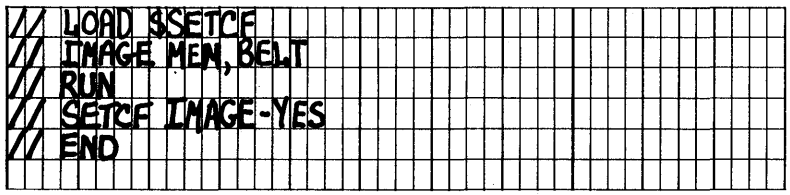
An image statement is required to modify the print belt image for the display station.

```
// RUN
{// SETCF . . .
 // SETB . . .
 // SETR . . .
 // SETS . . .
 // SETP . . .}
// END
```

Examples

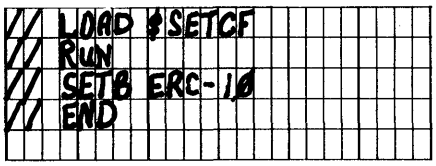
Example A

Replace the current print belt image with the image in the source member named BELT:



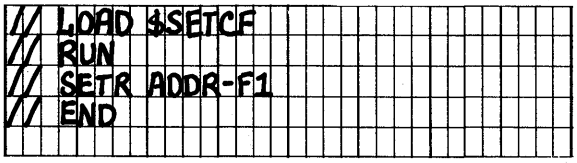
Example B

Change the current BSC error retry count to 10:



Example C

Change the EBCDIC addressing characters for an existing tributary station to 11 (see Appendix G for a list of valid addressing characters):



$SFGR–Screen Format Generator Utility Program

Function

The $SFGR utility program processes user-generated or compiler-generated display screen format specifications and produces a display screen format (or formats) in a specified load member.

For information about display screen formats and how they are created, see *Using Display Screen Formats* in Chapter 6.

In addition to the display screen format, $SFGR generates lists of:

- Source specifications
- Diagnostic or informational messages
- Indicators used
- Fields that might require output from the user program (in the order in which those fields appear in the output record area)
- Input fields on the display (in the order in which they appear in the input record area)

Display Screen Format Specifications

Input to $SFGR is a library source member that contains display screen format specifications. Before you code these specifications, you should lay out the entire display on the display layout sheet portion of the *IBM 5251 Display Station Keyboard Template Assignment Sheet and Display Screen Layout Sheet*, shown in Figure 4-5. Then, you can code the specifications on the *IBM System/34 Display Screen Format Specifications* (see Figures 4-6 and 4-7). The *IBM WSU/$SFGR Debugging Template*, can be used for debugging listings of display screen format specifications.

Unless indicated otherwise, alphameric entries should be left-adjusted, numeric entries should be right-adjusted, and leading zeros are not required except for indicators 01-09.

For information about generating the display screen format specifications, as well as a description of the programming considerations to keep in mind when you generate them, see *Using Display Screen Formats* in Chapter 6. This section describes, in detail, the contents of the specifications themselves. Following is a detailed description of each entry on the specification sheet.

Display Control Specification

The first record coded in the display screen specifications is called the display control specification. The display control specification provides information about the display screen format that, in general, is unrelated to the fields being defined. One display control specification is required and must be the first record in the display screen format specifications for a format. Figure 4-6 shows the portion of the display screen format specifications used for coding the display control specification.

IBM®

5251 Display Station Keyboard Template Assignment Sheet and Display Screen Layout Sheet

Format Name _____ Description _____

Job Name _____ Sheet ____ of ____

Originated by _____ Date _____

Display Mode	13	14	15	16	17	18	19	20	21	22	23	24	Clear
	1	2	3	4	5	6	7	8	9	10	11	12	Test Request

Keyboard Template Assignments

1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		

Address comments concerning this form to IBM Corporation, Department 245, Rochester, Minnesota 55901.

File No. S5250/S34-89

GX21-9271-0 UM/050*
Printed in U.S.A.

*Number of forms per pad could vary slightly.

Note: This side of the form can be used as a work sheet for assigning command key functions in the user program.

Figure 4-5 (Part 1 of 2). Keyboard Template Assignment Sheet and Display Screen Layout Sheet

Display Screen Layout Sheet

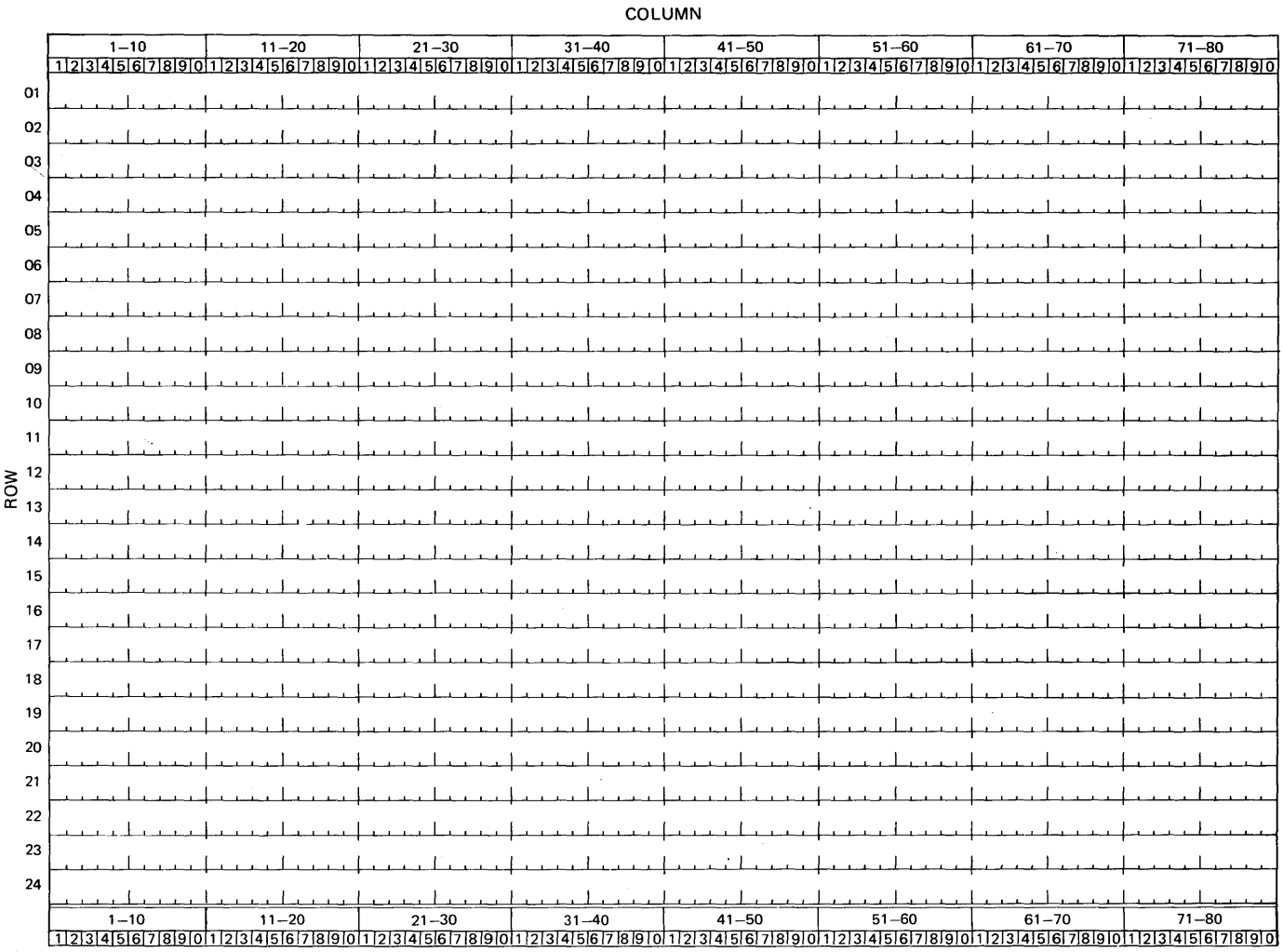


Figure 4-5 (Part 2 of 2). Keyboard Template Assignment Sheet and Display Screen Layout Sheet

IBM

System/34 Display Screen Format Specifications

Second Edition

Use this coding sheet only to define display screen formats for WSU and $SFGR. This coding sheet could contain typographical errors.

GX21-9253- U/M 050*
Printed in U.S.A.
*No. of sheets per pad may vary slightly.

S	Sequence Number	Form Type	Format Name	Format ID (WSU Only)	Start Line Number	Number of Lines to Clear	Lowercase	Return Input	Reset Keyboard (WSU Only)	Sound Alarm	Enable Function Keys	Enable Command Keys	Blink Cursor	Erase Input Fields	Override Fields	Suppress Input	Reserved	WSU Only: Enter Mode Sequence: Start	End	Entry Required	Repeat	Reserved	Priority	Preprocess	WSU Only: Review Mode Record Identifying Indicators: 1	2	3	WSU Only: Insert Mode Record Identifying Indicators: 1	2	3	Reserved	Key Mask	Reserved
	1 2 3 4 5	6	7 8 9 10 11 12 13 14	15 16	17 18	19 20	21	22	23 24	25 26	27	28	29 30	31 32	33 34	35 36	37 38 39 40	41	42	43	44	45	46	47	48 49	50 51	52 53	54 55	56 57	58 59	60 61 62 63	64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	80
S		S																														

Figure 4-6. Display Control Specification

The entries in the display control specification and their meanings are:

Columns 1-5–Sequence Number

Columns 1 through 5 can contain record sequence information. Information in these columns is listed when the specifications are listed but is not processed by the $SFGR utility program.

Column 6–Form Type

The character S identifies this record as the screen display control specification.

Columns 7-14–Format Name

This entry assigns a unique format name to the format being defined. Duplicate names assigned to formats used by the same user program are invalid. Duplicate format names can be used by different user programs. The format name can be any combination of characters except commas (,), single quotes ('), and blanks. The first character of the format name must be alphabetic, #, $, or @. The number of characters in the format name must not exceed 8.

Note: SSP-ICF uses format names that begin with $$; therefore, you should avoid using format names that begin with $$.

Columns 15-16–Format ID (WSU only)

Not used for $SFGR.

Columns 17-18–Start Line Number

The start line number is the number of the line at which the display begins. The actual line number on which a field begins can be determined from the following equation:

Actual line number	=	Start line number specified in columns 17-18	+	Line number specified in columns 19-20 of field definition specification	-1

For example, if 6 is specified as the start line number and 5 is specified as the line number in the field definition (columns 19 and 20 of the field definition specification), the field is actually displayed on line 10 (6 + 5 - 1). Valid entries in the start line number field are V (in column 17) or a line number that is less than or equal to the number of lines on the display screen. If V is specified, the start line number is identified in the file definition statement of the user program at execution time. If V is specified and no start line number is identified in the file definition statement of the user program, or if no value is entered in columns 17 and 18, 1 is assumed as the start line number.

Columns 19-20–Number of Lines to Clear

Specifies the number of lines to be cleared, including and following the starting line specified in columns 17 and 18, before the format is displayed. Valid values for the number of lines to clear are 00 through 24. The specified number of lines are cleared, beginning with the start line specified in columns 17 and 18. For example, if the start line number is 6, and if 12 lines are to be cleared, then display screen lines 6 through 17 are cleared before the format is displayed. The number of lines specified cannot be greater than the number of lines on the display screen. If a value is not specified (blank) in columns 19 and 20, all lines are cleared.

Note: If a format does not clear all the lines on the screen, (that is, a value other than 24 or blanks is specified in columns 19 and 20), be careful of overlaying currently displayed fields and attributes on lines not cleared by the format. For further information, see *Multiple Formats* in Chapter 6.

Column 21–Lowercase

If Y (yes) is specified, all alphabetic characters entered at the keyboard are displayed and sent to the user program in lowercase (if the Shift key is not pressed when the character is entered) or in uppercase (if the Shift key is pressed when the character is entered).

If N (no) is specified or if column 21 is left blank, all alphabetic characters entered at the keyboard are displayed and sent to the user program in uppercase, regardless of whether or not the Shift key is pressed.

Column 22–Return Input

If Y (yes) is specified or if column 22 is left blank, all input fields are returned to the user program even if the operator did not press any data keys. If mandatory enter (column 29 of the field definition specification) is specified for any field in the format, the operator *must* enter data in that field. If, for example, the operator presses a command key and does not enter data in the mandatory-enter field, the command key request is rejected.

If N (no) is specified, input data fields are returned to the user program only if the operator presses one or more data keys. If the operator does not enter *any* data on the screen, he does not have to enter information into mandatory-enter fields. If, for example, the operator presses a command key and does not enter any data into the display, the command key request is accepted, and no data is returned to the user program.

Note: When you are programming in RPG II and you specify N in column 22, and the operator presses a command key but enters no data, the return input indicator comes on. The reason for this is that by pressing the command key, a byte is returned to the program, which RPG II reads as data.

Columns 23-24–Reset Keyboard (WSU only)

Columns 25-26–Sound Alarm

If Y (yes) is specified in column 25, the audible alarm sounds when the format is displayed.

If N (no) is specified in column 25 or if columns 25 and 26 are left blank, the alarm does not sound.

If an indicator (01 through 99) is specified in columns 25 and 26, the audible alarm sounds if the specified indicator is on.

Column 27–Enable Function Keys

During normal program operation, the SSP sends a return code to the user program when an enabled function key is pressed.

If Y (yes) is specified in column 27, the function control keys identified by numbers in the key mask entry (columns 64 through 79) are enabled (allowed). If the key mask entry contains no numbers, all function control keys are disabled. The function control keys and the numbers that identify them are:

Function Control Key	**Number**
Print	1 (See *Print Key Exception*, which follows.)
Roll Up	2
Roll Down	3
Clear	4
Help	5 (See *Help Key Exception*, which follows.)
Home key when the cursor is in the home position (record backspace)	6

If N (no) is specified in column 27, the function control keys identified by numbers in the key mask entry are disabled (not allowed). If the key mask entry contains no numbers, all function control keys are enabled. If the operator presses a disabled function control key, an error message is displayed. The operator can then press the Error Reset key, followed by the correct function control key.

Print Key Exception: When the Print key is enabled, it performs the same function that the other enabled function control keys do. When the Print key is disabled, it causes the contents of the display screen to be printed.

Help Key Exception: Whether enabled or disabled, the Help key performs its normal SSP function when the display station is in error mode. It also performs its normal system function when it is disabled and the display station is in normal mode. When the Help key is enabled and the display station is in normal mode, it performs the same function that the other enabled function control keys perform.

If R (retain) is specified in column 27, the function control key mask that is active for the display station is retained when this format is displayed.

If column 27 is blank, all function control keys are enabled. In this case, the key mask entry must not contain any numbers.

For more information about function control key usage in RPG programs, see the section that describes work station file considerations in the *RPG Reference Manual*.

Column 28–Enable Command Keys

The SSP sends a return code to the user program when an enabled command key is pressed.

If Y (yes) is specified in column 28, the command keys identified by alphabetic characters in the key mask entry (columns 64 through 79) are enabled (allowed). If the key mask entry contains no alphabetic characters, all command keys are disabled. The command keys and the alphabetic characters that identify them are:

Command Key	**Alphabetic Character**
1 through 14	A through N
15 through 24	P through Y

If N (no) is specified in column 28, the command keys identified by alphabetic characters in the key mask entry are disabled (not allowed). If the key mask entry contains no alphabetic characters, all command keys are enabled. If the operator presses a disabled command key, an error message is displayed. The operator can then press the Error Reset key, followed by the correct command key.

Enabled function and command keys are returned to the COBOL program in the control area specified for a transaction file.

Enabled command keys are returned to the BASIC program in the intrinsic function CMDKEY.

If R (retain) is specified in column 28, the command key mask that is active for the display station is retained when this format is displayed.

If column 28 is blank, all command keys are enabled. In this case, the key mask entry must not contain any alphabetic characters.

Columns 29-30–Blink Cursor

If Y (yes) is specified in column 29, the cursor blinks when the format is displayed.

If N (no) is specified in column 29 or if columns 29 and 30 are left blank, the cursor does not blink.

If an indicator (01 through 99) is specified in columns 29 and 30, the cursor blinks if the specified indicator is on.

Columns 31-32–Erase Input Fields

The erase input fields entry allows the program to erase the input fields and output/input fields on a display and to reset the keyboard by setting an indicator and redisplaying the format. The format is not transmitted to the display station. The format that specifies erase input fields need not be the same format which defined the input fields. You might want to request erase input fields when an application requires an operator to enter information into the same display time after time.

If an indicator (01 through 99) is specified in columns 31 and 32, an erase input fields operation can be performed for this format. Input fields are erased and the keyboard is reset if the specified indicator is on when the format is displayed; all other fields are unchanged. A normal put operation is performed if the indicator is off when the format is displayed.

If Y (yes) is specified in column 31, an erase input fields operation occurs every time this format is displayed. Ordinarily, Y should not be specified since all entries on the following field definition specifications are ignored when the format is displayed. Instead, if an erase input fields operation may be performed for this format, an indicator should be specified.

If N (no) is specified in column 31 or if column 31 and 32 are left blank, an erase input fields operation is not allowed for this format.

Notes:

1. If the entry in columns 31 and 32 causes an erase input fields operation to occur and there are no input fields currently defined on the screen, a display station error occurs.
2. There is a modified data tag associated with each input field currently defined on the screen. When data is keyed into an input field, the modified data tag is set on for that field. The erase input fields command erases only those input fields that have the modified data tag set on. When a format (with input fields defined and return input specified on the S spec) is displayed, the modified data tag is set on for all input fields that are not mandatory enter fields. An erase input fields command turns off all modified data tags in order to maintain the mandatory enter characteristics of any mandatory enter fields that might exist on the screen. When a format is displayed via an override operation, no modified data tags will be set on even if the format had return input specified.

Therefore, if a format with return input specified is displayed, followed by an erase input fields command, followed by an override operation (which writes data to output/input fields), followed by another erase input fields command, the first erase input fields command will erase all input fields, but the second erase input fields command will erase only those input fields, that were keyed in by the display station operator following the first erase input fields command.

Columns 33-34–Override Fields

An override operation allows you to override fields in a format when you redisplay the same format. To perform an override operation, you must specify an indicator (01 through 99) in columns 33 and 34. An override operation is performed if the indicator is on when the format is displayed. A normal output operation is performed if the indicator is off when the format is displayed.

During an override operation (the indicator in columns 33 and 34 is on), the following occurs:

- If a field has an indicator specified in columns 23 and 24 of the D specification and that indicator is off, the data in the field is unchanged. If data was keyed into the field, that data is unchanged. Any field that had Y, N, or blank specified in columns 23 and 24 is also unchanged.

- If a field has an indicator specified in columns 23 and 24 of the D specification and that indicator is on, the field is displayed with data from the program. Any data that was keyed into the field by the operator is lost. Output information is displayed from the *same location in the output record area* as for a normal display.

- For all fields, the use of indicator-controlled attributes such as highlight or reverse image is determined by the state of that indicator. All field attributes that are not controlled by indicators are unchanged.

  Note: If an indicator is specified in the protect field entry (columns 37 and 38 of the D specification), that indicator is ignored during the override operation.

For example, you may want to override fields in a display if the operator keys incorrect data into a field. To do this, specify an indicator in columns 33 and 34 of the S specification, which allows the format to be overridden. If the operator keys incorrect data into a field, you can then set on the indicator in columns 33 and 34 and redisplay the format. If the indicator specified in columns 23 and 24 of the D specification is off for the field, the incorrect data is unchanged. If the indicator is on, data from the program is displayed. You can also use indicators for field attributes such as highlight and reverse images and set these indicators on when the override indicator is set on.

Note: Since only a portion of the format is sent to the work station, when an override operation is done, be careful about using an override operation on a format which is not currently displayed.

Figure 4-7 summarizes the effect of indicators on output data during an override operation.

Indicator in Columns 23 and 24 of the D Specification	Indicator in Columns 33 and 34 of the S specification: OFF	ON
OFF	Output data comes from D specification (columns 57 through 79).	No change occurs to data on the screen.
ON	Output data comes from the program.	Output data comes from the program.
	Normal output operation	Override Operation

Figure 4-7. Effect of Indicators on Output Data during an Override Operation

If Y (yes) is specified in column 33, an override operation is performed every time this format is displayed. Ordinarily, Y should not be specified. Instead, if an override operation may be performed for this format, an indicator should be specified.

If N (no) is specified in column 33 or if columns 33 and 34 are left blank, an override operation is not allowed for this format.

Columns 35-36–Suppress Input

If Y (yes) is specified in column 35 or the specified indicator is on, *and* the keyboard is unlocked when the program displays this format, then the keyboard will remain unlocked and the operator can enter information into the input fields. If the operator presses the Enter/Rec Adv key, the keyboard will only lock. The input fields will not be read and will not be returned to the user program until (1) a format is displayed with suppress input specified as N (no) or with the specified indicator off, and (2) the operator presses the Enter/Rec Adv key following the display of this format.

If Y (yes) is specified in column 35 or the specified indicator is on, *and* the keyboard is locked when the program displays this format, *and* the format defines input field(s), then the keyboard will unlock and the operator can enter information into the input fields. If the operator presses the Enter/Rec Adv key, the keyboard will only lock. The input fields will not be read and will not be returned to the user program until (1) a format is displayed with suppress input specified as N (no) or with the specified indicator off, and (2) the operator presses the Enter/Rec Adv key following the display of this format.

If Y (yes) is specified in column 35 or the specified indicator is on, *and* the keyboard is locked when the program displays this format, *and* the format does not define input field(s), then the keyboard will remain locked and the operator cannot enter information into the input fields. The input fields will not be read and will not be returned to the user program until (1) a format is displayed with suppress input specified as N (no) or with the specified indicator off, and (2) the operator presses the Enter/Rec Adv key following the display of this format.

If N (no) is specified in column 35, or if columns 35 and 36 are left blank, or if the specified indicator is off, then input fields are read by the SSP as soon as the Enter/Rec Adv key is pressed. The input fields will be returned to the user program when and if the user program requests the input data.

If an indicator (01 to 99) is specified in columns 35 and 36, input is suppressed if the specified indicator is on.

Notes:

1. When multiple formats are displayed before information is read from the display screen, suppress input should be specified on all but the last display even if the displays do not contain input fields. If an RPG II MRT program transmits multiple displays and does not suppress input on all but the last display and if the operator interrupts the program with the ATTN key before the last display is transmitted, *the program will be suspended.* No other requesters will be serviced during the inquiry request. No indication that the program is suspended is given to the operators at the other display stations.
2. When multiple formats are displayed with more than one format defining input fields, data can be entered into and will be returned from only the input fields defined by the last displayed format that defined input fields. See *Multiple Formats* in Chapter 6 of this manual for further information.

Columns 37-63

Columns 37 through 63 are not used by $SFGR and must be left blank.

Columns 64-79–Key Mask

The key mask is a string of numbers or alphabetic characters that identify keys to be enabled or disabled when this format is displayed. The key mask must begin in column 64 and cannot contain embedded blanks. The numbers and alphabetic characters can be specified in any order.

Numbers in the key mask identify function control keys:

Number	**Function Control Key**
1	Print
2	Roll Up
3	Roll Down
4	Clear
5	Help
6	Home key when the cursor is in the home position (record backspace)

If the enable function keys entry (column 27) is Y, the numbers in the key mask identify function control keys to be enabled. If the enable function keys entry is N, the numbers in the key mask identify function control keys to be disabled. If the key mask does not contain any numbers, none of the function control keys are enabled (if column 27 is Y) or disabled (if column 27 is N).

Alphabetic characters in the key mask identify command keys:

Alphabetic Characters	**Command Keys**
A through N	1 through 14
P through Y	15 through 24

If the enable command keys entry (column 28) is Y, the alphabetic characters in the key mask identify command keys to be enabled. If the enable command keys entry is N, the alphabetic characters in the key mask identify command keys to be disabled. If the key mask does not contain any alphabetic characters, none of the command keys are enabled (if column 28 is Y) or disabled (if column 28 is N).

For example, column 27 is Y, column 28 is N, and columns 64 through 68 are 236AP. In this example, the roll up, roll down, and record backspace function keys are enabled and all command keys, except 1 and 15, are enabled.

Field Definition Specifications

The records that follow the display control specification are called field definition specifications. Each field definition specification completely describes a field on the display. Each field on the display format must be specified by a field definition specification. Figure 4-8 shows the portion of the screen format specification sheet used for coding field definition specifications.

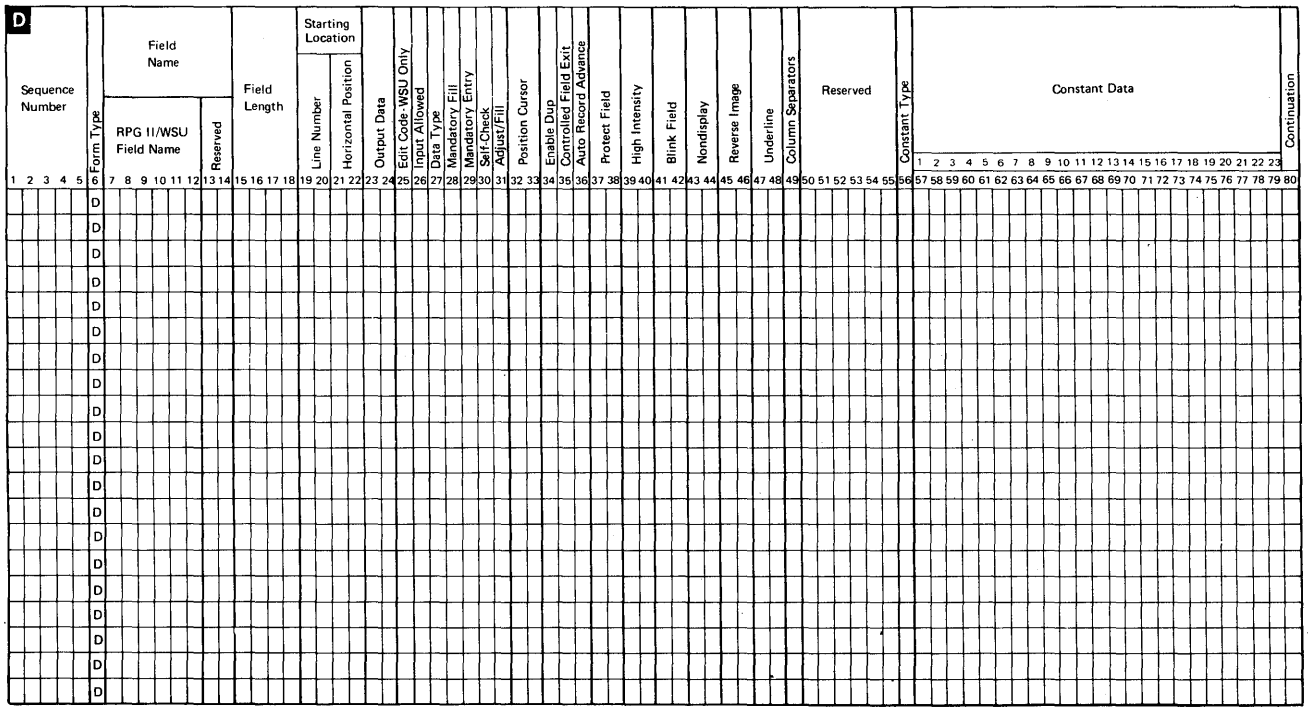


Figure 4-8. Field Definition Specifications

The entries in a field definition specification and their meanings are:

Columns 1-5–Sequence Number

Columns 1 through 5 can contain record sequence information. Information in these columns is listed when the specifications are listed but is not processed by the screen format generator program.

Column 6–Form Type

The form type identifies the record type. D indicates the record is a field definition.

Columns 7-14–Field Name

An * (asterisk) in column 7 identifies the record as a comment, unless the record is a continuation (X in column 80 of the preceding record).

If an * is not coded in column 7 and the record is not a continuation, columns 7 through 14 specify the name of the field being defined. The entry is used for reference only; it is not included in the generated format.

Columns 15-18–Field Length

Specifies the length of the field. The field length can be any number from 1 through 1919. Leading zeros are not required, but the value must be right-justified in the entry. (The first position on the screen is reserved for a screen attribute.)

Notes:

1. If an output field displays a message identified by a message identification code and a message member identifier in the user program output record, the recommended minimum field length is 6 bytes. If the length of the message text is greater than that of the output field, the message text will be truncated to the length of the output field. If the length of the output field is greater than that of the message text, the message text will be displayed, followed by null fills.
2. If the ideographic version of the SSP is being used and if the field is defined as data type X, E, or F (in column 27), the field length must be an even number that is equal to or greater than 4.

Columns 19-20–Line Number

Specifies the number of the line, relative to the start line number, on which the field begins. The number of the line on which the field actually begins can be determined from the following equation:

Actual line number	=	Line number specified in columns 19-20	+	Start line number specified on the display control specification	-1

For example, if 5 is specified as the line number and 6 is specified as the starting line number (columns 17 and 18 on the display control specification), the field actually begins on line 10 (5 + 6 - 1).

The actual line number cannot exceed 24. Leading zeros are not required, but the value must be right-justified in the entry.

Note: If the ideographic character version of the SSP is being used and if ideographic fields or output data are defined for this format, the line number cannot be greater than 12.

Columns 21-22–Horizontal Position

Specifies the column number of the first position in the field. The column number can be any number from 1 through 80. Leading zeros are not required, but the value must be right-justified in the entry.

Notes:

1. The first position on the screen (line 1, position 1) is reserved for a nondisplay screen attribute. Therefore, a field cannot begin in line 1, position 1. Also, each display field is preceded by a nondisplay control character that defines the characteristics of the field. At least one space must be allowed between fields for this control character. If a field begins in position 1 of a line, the control character occupies the last position of the preceding line.
2. The horizontal position must be an even number if the following conditions are true:
   - The ideographic version of the SSP is being used.
   - The horizontal position plus the field length (columns 15 through 18) is greater than 81 and the field type is X, E, or F.

Columns 23-24–Output Data

If Y (yes) is specified in column 23 and constant data or a message identification code and message member identifier are also specified in columns 57 through 79, the information specified in columns 57 through 79 is displayed in the field or the contents of the message identified by the message identification code are displayed.

If Y (yes) is specified in column 23 and if columns 57 through 79 are blank, the following statements are true:

- If M is not specified in column 56, data from the user program output data area is displayed.
- If M is specified in column 56, the message identified by the message identification code and message member identifier in the user program output record area is displayed.

If N (no) is specified in column 23 or if columns 23 and 24 are left blank, the field is not an output field.

If an indicator (01 through 99) is specified, the following statements are true:

- If the specified indicator is on when the format is displayed and M is not specified in column 56, then data from the user program output record area is displayed.
- If the specified indicator is on when the format is displayed and M is specified in column 56, then the message identified by the message identification code and message member identifier in the user program output record area is displayed.
- If the specified indicator is off when the format is displayed and M is not specified in column 56, then the data in columns 57 through 79 is displayed. If columns 57 through 79 are blank, blanks are displayed.
- If the specified indicator is off when the format is displayed and M is specified in column 56, then the message identified by the message identification code and message member identifier in columns 57 through 79 is displayed. If columns 57 through 79 are blank, blanks are displayed.
- If the user program performs an override operation and the specified indicator is on, data supplied by the user program or the message identified by the user program is displayed in the field. For information about override operations, see the description of columns 33 and 34 on the display control specification.
- If the user program performs an override operation and the specified indicator is off, the field is unchanged.

Note: If a field is specified as an output field but not as an input field (N or blank in column 26), then data in the field cannot be changed by the display station operator. If an output field is also defined as an input field (Y in column 26), then data in the field can be changed by the display station operator. If an indicator is specified, space *must* be reserved for the field in the user program output record area. (If M is specified in column 56, only 6 bytes need be reserved for the field.)

Column 25–Edit Code (WSU only)

Not used by $SFGR.

Column 26–Input Allowed

If Y (yes) is specified, the field is an input field.

If N (no) or blank is specified, the field is not an input field.

For information about determining the maximum number of input fields, see *Using Display Screen Formats* in Chapter 6.

Column 27–Data Type

Specifies the type of data that can be placed in the input field (data type cannot be specified for an output field):

Data Type	**Meaning**
A	The field can contain only the characters A through Z, commas, periods, hyphens, or blanks.
N	The field can contain only numeric data. Commas, periods, plus signs, or minus signs can also be entered in this field. *Note:* If special characters (commas, periods, plus signs, or minus signs) are entered in an N-type field, the data read by a program may not be as expected. The program uses only the digit portion of characters entered in an N-type field. The zone portion is forced to hex F, except for the sign position.
B or blank	The field can contain only alphameric (A/N) data.
S	The field can contain only signed numeric data; the last position of the field is reserved for a sign. Only decimal digits (0 through 9) can be entered into the field, and a control field exit key must be used to exit from the field. The Field+ and Field Exit keys can be used to enter a positive value. The Field- key can be used to enter a negative value. The field can be from 2 through 16 characters long and will occupy from 1 through 15 characters in the input or output buffer. If a negative number is sent as output to a signed numeric field, the minus sign will be displayed along with the number. If you specify blank fill (B or a blank in column 31), the leading zeros in a signed numeric field are replaced by blanks. When you specify signed numeric data only, adjust/fill and control field exit Y (yes) are assumed.

Data Type	Meaning
K	The field can contain Katakana characters.
R	The field contains data read from the magnetic stripe reader. The field can be up to 128 characters long. Nondisplay must be specified in columns 43 and 44 for the field.
X	The field can contain only ideographic characters.
E	The field can contain alphameric (A/N) and Katakana characters or ideographic characters, but not both. The field is initially filled with binary zeros, and the display station is set to enter alphameric and Katakana characters.
F	The field can contain alphameric (A/N) and Katakana characters or ideographic characters, but not both. The field is initially filled with ideographic nulls (*shift out*, followed by binary zeros, followed by *shift in*), and the display station is set to enter ideographic characters.

Notes:

1. X, E, and F are used for the ideographic version of the SSP.
2. If the X, E, or F field is an output/input field (Y in column 23 or columns 23 and 24 specify an indicator that is on when the field is displayed), the output data, whether from the format or from the program's buffer, will overlay the alphameric or ideographic nulls and override the described characteristics of X, E, and F fields.

Column 28–Mandatory Fill

Y (yes) specifies that if 1 character is entered in the field, all positions in the field must be filled with nonnull characters. (A nonnull character is any character that can be entered from the keyboard or a space produced by the operator pressing the space bar.)

You can use cursor keys to exit from a mandatory fill field. However, if you use the cursor keys, no adjusting will occur.

N (no) or blank specifies that the field need not be filled.

Note: Mandatory fill and adjust/fill (column 31) cannot be specified for the same field.

Column 29–Mandatory Entry

If Y (yes) is specified, the display station operator must enter at least 1 character in the field before input from the display can be returned to the application program.

Note: An operator can bypass a mandatory enter field if:

- All input fields on the display are mandatory enter fields, the Return Input entry on the S specification is Y, and the operator enters data in none of the input fields.
- The Return Input entry on the S specification is N and the operator enters data in none of the input fields.

You can use cursor keys to exit from a mandatory entry field. However, if you use the cursor keys, no adjusting will occur.

If N (no) or blank is specified, the display station operator does not have to enter information into the field.

Column 30–Self-Check

If modulus 10 or modulus 11 self-checking is specified, the information entered into the field is checked by the appropriate check algorithm after the field is entered. Self-check fields cannot be longer than 32 positions.

If T is specified, the field is a modulus 10 self-check field.

If E is specified, the field is a modulus 11 self-check field.

If blank is specified, the field is not a self-check field.

For information on computing modulus 10 and modulus 11 self-check digits, see *Computing Self-Check Digits* in Chapter 6.

Column 31–Adjust/Fill

If Z (right-adjust, zero fill) is specified, information entered into the field is right-adjusted and unused positions are filled with zeros before the field is transmitted to the user program.

If B (right-adjust, blank fill) is specified, information entered into the field is right-adjusted and unused positions are filled with blanks before the field is transmitted to the user program. For output to a signed-numeric field, leading zeros are replaced by blanks.

If column 31 is left blank, right-adjust, blank fill is assumed on input for signed-numeric; no adjust, no fill is assumed for all other fields.

When you specify adjust/fill, control field exit Y (yes) is assumed.

Notes:

1. Adjust/fill and mandatory fill (column 28) cannot be specified for the same field.
2. To enter adjust/fill fields, the operator must press the Field+ key for numeric or signed numeric fields, the Field- key for signed numeric fields, or the Field Exit key. The operator can use the Field- key to enter a numeric-only field only from a remote display station. For output to a signed-numeric field, leading zeros are replaced by blanks. Operators can press the Field Advance key to enter an adjust/fill field, but the adjust/fill will not occur.

Columns 32-33–Position Cursor

The position cursor entry is used to explicitly position the cursor at one of the input fields in this format. The position cursor entry controls the cursor position only if the keyboard is going from a disabled to an enabled state. (The keyboard is enabled whenever a program displays a format that does not suppress input or whenever an assembler program requests a put with invite operation.) If the keyboard is already enabled when this format is displayed, the home position of the cursor is changed, but the cursor itself is not repositioned. For example, if a program displays a format without suppressing input, the keyboard is enabled. If a second format that does not clear the entire screen is then displayed and if position cursor is specified, the cursor is not repositioned. For further information about displaying two or more formats before reading from the display screen, see *Multiple Formats* in Chapter 6.

If Y (yes) is specified in column 32, the cursor appears at the first position in this field, unless other fields have indicators specified in columns 32 and 33 and one or more of the indicators are on when the format is displayed. In that case, the cursor appears in the first position of the first field on the display with its specified indicator on. Y can be specified for only one field in the display.

If N (no) is specified in column 32 or if columns 32 and 33 are left blank, the cursor appears in the first position of this field only if all of the following conditions are true:

- This field is the first unprotected input field on the display.
- No field has a Y specified in column 32.
- No other field has specified (in columns 32 and 33) an indicator that is on.

If an indicator (01 through 99) is specified in columns 32 and 33 and that indicator is on when the format is displayed, the cursor appears in the first position of this field, unless a preceding field on the display has an indicator that is on specified in columns 32 and 33. In that case, the cursor appears in the first position of the first field on the display with its specified indicator on.

Note: Refer to the notes under columns *37-38–Protect Field* for special considerations regarding the position cursor when the field protected.

Column 34–Enable Dup

If Y (yes) is specified, the Dup (duplicate) key can be entered in the field. When the Dup key is pressed, the position of the cursor and the remainder of the field are filled with the duplicate character value (hex 1C), which is displayed as an overscored asterisk (*). The duplicate characters must be processed by the user program.

If N (no) or blank is specified, the Dup key is invalid and will cause an error if it is pressed while the cursor is in the field.

Column 35–Controlled Field Exit

If Y (yes) is specified, one of the field exit keys (Field Adv, Enter/Rec Adv, Field Exit, Field+, Field- [if the field is a signed-numeric field], Field Backspace, Home, Erase Input, or Dup) must be pressed before the cursor will leave the field.

You can use the cursor keys to exit from a controlled field. However, if you use the cursor keys, no adjusting will occur.

If N (no) or blank is specified, the cursor automatically exits from the field when the field is filled. If you specify adjust/fill (column 31), an N entry is ignored.

Column 36–Auto Record Advance

If Y (yes) is specified, the input fields on the screen are automatically returned to the user program (automatic record advance) when one of the following occurs:

- The last character in the field is entered and N or blank is coded in column 35.
- The cursor is in the field and the Enter/Rec Adv, Field Adv, Field Exit, Field+, Field- (if the field is a signed-numeric field), or Dup field exit key is pressed.

If N (no) or blank is specified, an automatic record advance is not performed for the field.

Columns 37-38–Protect Field

If Y (yes) is specified in column 37, the field is bypassed (skipped over) whenever the cursor would appear within the field.

If N (no) is specified in column 37 or if columns 37 and 38 are left blank, the field is not bypassed.

If an indicator (01 through 99) is specified, the field is bypassed if the specified indicator is on.

Notes:

1. If an override operation is used, the indicator is ignored.
2. The cursor may appear in a protected field if:
   a. The field is protected by an indicator that is on when the display appears.
   b. The field is the first field defined on the D specification.
   c. The cursor is not positioned via an indicator to any field that is not protected (columns 32-33 for all other fields are blank or contain indicators that are off).

   To avoid this cursor positioning, use the same indicator to protect this field and to position the cursor in another field on the display.
3. If a field is explicitly defined as either or both nondisplay and protected (Y in columns 43 and 37) and if column separators are requested (Y in column 49), the column separators will be displayed.
4. If an indicator is used to control both the nondisplay and protect attributes and if column separators are requested, the separators will not be displayed if both indicators are on when the field is displayed.

Columns 39-40–High Intensity

If Y (yes) is specified in column 39, the field is displayed with high intensity.

If N (no) is specified in column 39 or if columns 39 and 40 are left blank, the field is displayed with normal intensity.

If an indicator (01 through 99) is specified, the field is displayed with high intensity if the specified indicator is on.

Note: High intensity, reverse image (columns 45 and 46), and underline (columns 47 and 48) cannot all be specified for the same field at the same time. If Y (yes) is specified in all three entries, an error message is displayed, and $SFGR terminates. If one or more of the entries are indicator values and if an attempt is made to display the field with all three of the attributes requested, the field will be displayed as a nondisplay field.

Columns 41-42–Blink Field

If Y (yes) is specified in column 41, the field blinks.

If N (no) is specified in column 41 or if columns 41 and 42 are left blank, the field does not blink.

If an indicator (01 through 99) is specified, the field blinks if the specified indicator is on when the format is displayed.

Columns 43-44–Nondisplay

If Y (yes) is specified in column 43, the field is a nondisplay field. That is, information in the field when the format is displayed or information entered into the field by the operator is not visible on the screen.

If N (no) is specified in column 43 or if columns 43 and 44 are left blank, the information in the field is displayed.

If an indicator (01 through 99) is specified, the field is a nondisplay field if the specified indicator is on when the format is displayed.

Notes:

1. If nondisplay is specified and high intensity (columns 39 and 40), reverse image (columns 45 and 46), or underline (columns 47 and 48) is also specified, the field is defined as nondisplay only.
2. If a field is explicitly defined as either or both nondisplay and protected (Y in columns 43 and 37) and if column separators are requested (Y in column 49), the column separators are displayed.
3. If an indicator is used to control both the nondisplay and protect attributes and if column separators are requested, the separators are not displayed if both indicators are on when the field is displayed.

Columns 45-46–Reverse Image

If Y (yes) is specified in column 45, the characters in the field appear as dark characters on a light background.

If N (no) is specified in column 45 or if columns 45 and 46 are left blank, the characters in the field appear as light characters on a dark background.

If an indicator (01 through 99) is specified, the characters on the field appear as dark characters on a light background if the indicator is on when the format is displayed.

Notes:

1. Reverse image, high intensity (columns 39 and 40), and underline (columns 47 and 48) cannot all be specified for the same field at the same time. If Y (yes) is specified in all three entries, an error message is displayed and $SFGR terminates. If one or more of the entries are indicator values and if an attempt is made to display the field with all three of the attributes requested, the field will be displayed as a nondisplay field.
2. If you specify a number of large reverse-image fields, the screen may appear to flash when the format is transmitted to the display station. This is particularly true if the format is transmitted to a remote work station. Although this flash does not affect system or format performance, it may be irritating to the display station operator. To correct the flashing if it occurs, place the field definitions for all reverse image fields toward the end of the D specifications.

Columns 47-48–Underline

If Y (yes) is specified in column 47, the field is underlined.

If N (no) is specified in column 47 or if columns 47 and 48 are left blank, the field is not underlined.

If an indicator (01 through 99) is specified in columns 47 and 48, the field is underlined if the specified indicator is on.

Note: Underline, high intensity (columns 39 and 40), and reverse image (columns 45 and 46) cannot all be specified for the same field at the same time. If Y (yes) is specified in all three entries, an error message is displayed and $SFGR terminates. If one or more of the entries are indicator values and if an attempt is made to display the field with all three of the attributes requested, the field will be displayed as a nondisplay field.

Column 49–Column Separators

If Y (yes) is specified, each character position in the field is preceded and followed by column separators (vertical lines). The column separators do not require additional character positions.

If N (no) or blank is specified, the column separators are not used.

Notes:

1. If a field is explicitly defined as either or both nondisplay and protected (Y in columns 43 and 37) and if column separators are requested (Y in column 49), the column separators are displayed.
2. If an indicator is used to control both the nondisplay and protect attributes and if column separators are requested, the separators are not displayed if both indicators are on when the field is displayed.

Columns 50-55–Reserved

Column 56–Constant Type

C indicates that constant information in columns 57 through 79 is to be displayed in the output field. C is required only if the information in columns 57 through 79 is all blanks and you want to display all blanks in the field. C is invalid if an indicator is specified in columns 23 and 24.

If column 56 is blank and constant information is in columns 57 through 79, then the data is displayed. If column 56 is blank and columns 57 through 79 are also blank, then information from the program output record area is displayed.

M indicates that a message identification code and a message member identifier are in columns 57 through 79 or in the user program output record area. For further information, see the description of columns 23 and 24.

Columns 57-79–Constant Data

This field specifies the information to be placed in an output or output/input field when the format is generated. If information is to be placed in the field, columns 57 through 79 should contain one of the following:

- The actual information to be displayed in the field.

- A 4-digit message identification code in columns 57 through 60 and a 2-character message member identifier in columns 61 and 62. The MIC identifies the message containing the information to be displayed in the field. Depending upon the length of the output field, the message text is padded with nulls or truncated when the format is displayed. The message member identifier identifies the message member that contains the message. The message identifier can be one of the following:

Identifier	Message Member Containing the Message
U1 or blank	User-1 message member
U2	User-2 message member
P1	Program-1 message member
P2	Program-2 message member
M1	SSP level-1 message member (##MSG1)
M2	SSP level-2 message member (##MSG2)

If a message identification code is specified but a message member identifier is not, U1 is assumed.

For information about assigning user-1, user-2, program-1, and program-2 message members, see the description of the MEMBER OCL statement in Chapter 1.

Notes:

1. If columns 57 through 79 are blank and the field is an output field (Y in column 23), then information from the program output record is displayed.
2. If a message identification code is specified in columns 57 through 79, then only 6 bytes need to be reserved for the field in the program output record area.
3. M1 and M2 are valid only if SSP-YES is specified on the LOADMBR utility control statement described later.
4. For the ideographic version of the SSP, the constant can contain ideographic characters. The *shift out* and *shift in* characters are counted as part of the data. If the constant output consists of more than 10 ideographic characters, an alphameric X should be coded in column 80 and the ideographic data continued on the next line. If a *shift in* character is in column 78 or 79 and if the constant is continued with a *shift out* character in column 7 of the continuation record, the *shift in/shift out* pair and the intervening blank or character in column 79, if one exists, are deleted when the constant is concatenated. If a *shift out* character appears in column 78 and a *shift in* character is in column 79, $SFGR deletes the *shift out/shift in* pair when the constant is concatenated. If you are using extended ideographic characters, put a *shift out* character in column 57 of the record to cause the ideographic characters to be displayed correctly.

Column 80–Continuation (X)

If more than 23 characters of data are required, an X in column 80 indicates that the record is continued. Positions 7 through 79 of the following record contain the continued constant data.

Note: A comment cannot follow a record with X in column 80.

$SFGR is executed when the BLDMENU or FORMAT procedure is run.

Utility Control Statement Formats

To create one or more formats:

```
// LOADMBR NAME–load member name [,REPLACE–{NO|YES}] [,SSP–{YES|NO}]

// INOUT [INLIB–{library name|#LIBRARY}] [,OUTLIB–{library name|#LIBRARY}]
      [,PRINT–{YES|NO|PARTIAL}] [,HALT–{YES|NO}]

// CREATE SOURCE–source member name [,NUMBER–{1|nn}]
[// CREATE...
    •
    •
    •
// CREATE...]
// END
```

Note: A maximum of 32 CREATE statements can be specified in one run of $SFGR. Also, the total number of formats created by one $SFGR run cannot be greater than 32.

To update a format in an existing load member:

```
[// LOADMBR NAME–load member name
// INOUT [INLIB–{library name|#LIBRARY}] [,OUTLIB–{library name|#LIBRARY}]
      [,PRINT–{YES|NO|PARTIAL}] [,HALT–{YES|NO}]]

// UPDATE SOURCE–source member name
// END
```

To add a format to a load member:

```
// LOADMBR NAME—load member name
// INOUT [INLIB—{library name / #LIBRARY}] [,OUTLIB—{library name / #LIBRARY}]
      [,PRINT—{YES / NO / PARTIAL}] [,HALT—{YES / NO}]
// ADD SOURCE—source member name [,NUMBER—{1 / nn}]
// END
```

To delete a format from a load member:

```
// LOADMBR NAME—load member name
[// INOUT OUTLIB—{library name / #LIBRARY}]
// DELETE FORMAT—format name
// END
```

Parameters

For the LOADMBR statement:

NAME: The name of the library load member that contains or will contain the display format. If the load member is being created, it will be placed in the library specified by the OUTLIB parameter on the INOUT statement. If an add, update, or delete operation is being performed, the load member must already exist in the library specified by the OUTLIB parameter.

REPLACE: The REPLACE parameter has meaning only if CREATE statements are entered later. If REPLACE-YES is specified and a $SFGR load member with the same name as the load member being created already exists in the specified library, the existing member is replaced. No message is displayed to indicate that the member was deleted. If a load member with the same name exists but that load member is not a $SFGR load member, $SFGR displays a message. If the operator selects option 0, the member is replaced. If the operator selects option 3, the $SFGR run is terminated.

If REPLACE-NO is specified and a load member with the same name as the load member being created already exists in the specified library, $SFGR displays a message to the operator. The operator must then decide whether or not to delete the existing member.

If the REPLACE parameter is not specified, REPLACE-NO is assumed.

SSP: SSP-YES specifies that the format load member created is an SSP member. If SSP is not specified, SSP-NO is assumed.

For the INOUT statement:

INLIB: The name of the library that contains the source specifications. If INLIB is not specified, the system library (#LIBRARY) is assumed.

OUTLIB: The name of the library that contains or will contain the load member modified by or created by $SFGR. If OUTLIB is not specified, the system library (#LIBRARY) is assumed.

PRINT: PRINT-YES specifies that output from $SFGR is to be printed. The following information is printed:

- The screen source member name
- The screen format S- and D-specifications
- Any informational, warning, or terminal messages
- The input and output buffer descriptions
- A list of the screen format indicators used
- The input and output library names
- The screen format load member name
- The storage requirements for each format
- The data stream length for each format

PRINT-NO specifies that only terminal messages and the statement in error are to be printed.

PRINT-PARTIAL indicates that the following should be printed:

- The screen source member name
- Any warning or terminal errors together with the statement causing the error, or any informational messages
- The input and output library names
- The screen format load member name

If the PRINT parameter is not specified, PRINT-YES is assumed.

HALT: HALT-YES specifies that warning or terminal errors cause an error message to be displayed. For warning errors, the operator can either continue or cancel the screen format generation.

HALT-NO specifies that warning errors do not affect the screen format generation, but terminal errors cause the screen format generation process to be canceled, and the next job or job step is started.

Note: For terminal errors, the ?CD? OCL substitution expression is set to 1008; refer to *Substitution Expressions* in Chapter 5 for more information.

For the CREATE statement:

SOURCE: The name of the source library member that contains the screen format specifications for the formats being created.

NUMBER: The number of display formats defined in the screen format specifications source member. The number specified cannot exceed 32. If the NUMBER parameter is not specified, 1 is assumed.

For the UPDATE statement:

SOURCE: The name of the source library member that contains the new screen format specifications for the screen format being updated.

For the ADD statement:

SOURCE: The name of the source library member that contains the screen format specifications for the format(s) being added to the load member.

NUMBER: The number of display formats defined in the screen format specifications source member. The number of display formats in a load member cannot exceed 32. If the NUMBER parameter is not specified, 1 is assumed.

For the DELETE statement:

FORMAT: The name of the display format being deleted.

Control Statement Sequence

```
// LOAD $SFGR
// RUN
// LOADMBR . . .
// INOUT . . .

[// CREATE . . .
      •
      •
      •
// CREATE . . .]

[// ADD . . .
      •
      •
      •
// UPDATE . . .
      •
      •
      •
// DELETE . . .]

// END
```

Notes:

1. The LOADMBR statement must be the first statement following the RUN statement. Only one LOADMBR statement can be used during a $SFGR run.
2. If the INOUT statement is used, it must immediately follow the LOADMBR statement. Only one INOUT statement can be used during a $SFGR run.
3. Up to 32 CREATE statements can be used during a $SFGR run; but, if a CREATE statement is used, an ADD, UPDATE, or DELETE statement cannot be used.
4. Any combination of up to 32 ADD, UPDATE, or DELETE statements can be used during a $SFGR run.
5. After a display screen format is created for use with a program, only the following changes can be made to the format without requiring that the program also be modified and recompiled:
   a. Field attributes (except data type and field length) can be changed.
   b. The position of a field on the display screen can be changed.
   c. New constant output or output from a member can be added to the display.
6. An attempt to put duplicate format names into one load member will cause a terminal error condition.

   The $SFGR specifications must be kept in the original sequence since the screen field definitions and field locations in the program buffers must remain consistent.

Example

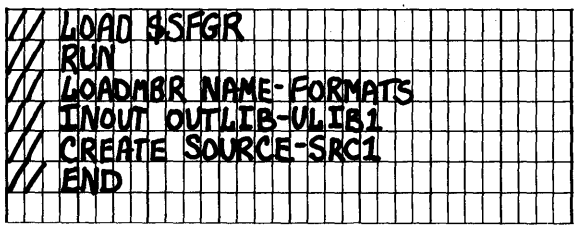
Generate a display screen format named FORMAT1. The format should be placed in a member named FORMATS in a user library called ULIB1.

Step 1: Lay out the fields on the display layout sheet. Figure 4-10 shows the layout sheet for this example.

Step 2: Fill out the display screen format specifications. Figure 4-11 shows the display screen format specifications for this example.

Step 3: Use the $MAINT utility or the source entry utility (SEU) to place the specifications in a system library source member called SRC1.

Step 4: Run $SFGR by entering the following statements:



Display Screen Layout Sheet

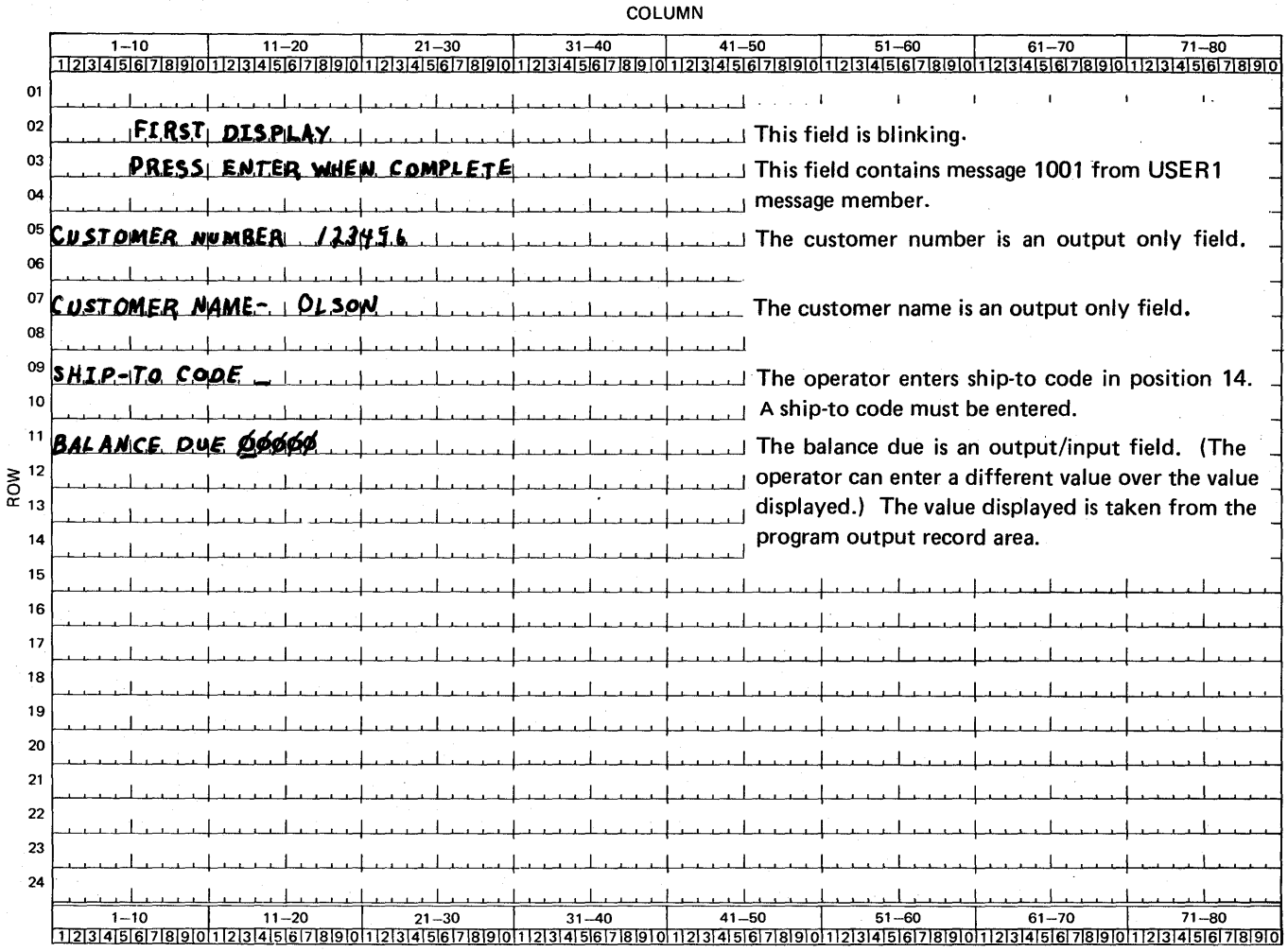
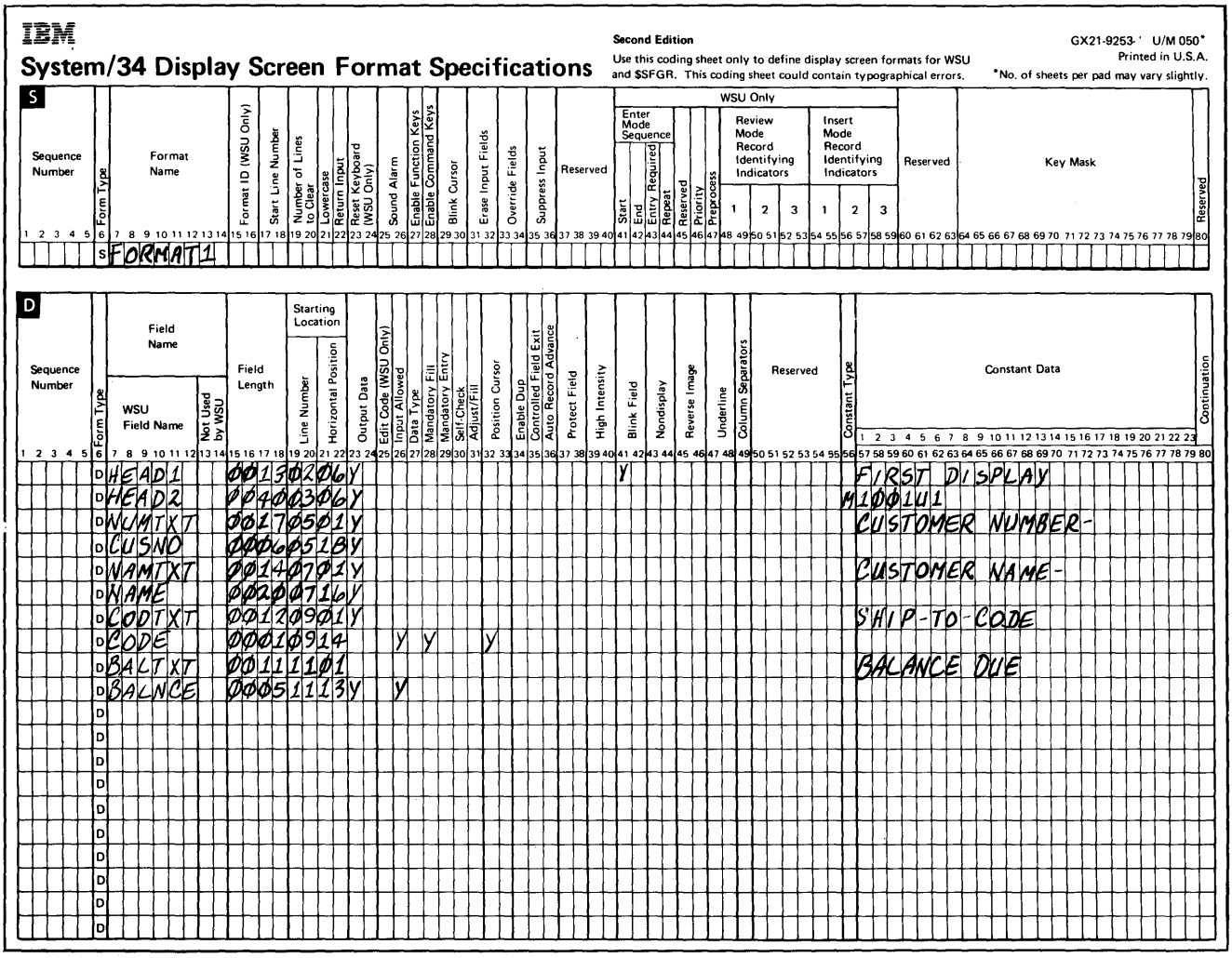


Figure 4-10. Filled-Out Layout Sheet



IBM

System/34 Display Screen Format Specifications

Second Edition

Use this coding sheet only to define display screen formats for WSU and $SFGR. This coding sheet could contain typographical errors.

GX21-9253-' U/M 050*
Printed in U.S.A.
*No. of sheets per pad may vary slightly.

S

Sequence Number	Form Type	Format Name	Format ID (WSU Only)	Start Line Number	Number of Lines to Clear	Lowercase	Return Input	Reset Keyboard (WSU Only)	Sound Alarm	Enable Function Keys	Enable Command Keys	Blink Cursor	Erase Input Fields	Override Fields	Suppress Input	Reserved	WSU Only: Enter Mode Sequence: Start	End	Entry Required	Repeat	Reserved	Priority	Preprocess	WSU Only: Review Mode Record Identifying Indicators: 1	2	3	WSU Only: Insert Mode Record Identifying Indicators: 1	2	3	Reserved	Key Mask	Reserved
1 2 3 4 5	6	7 8 9 10 11 12 13 14	15 16 17 18	19 20	21	22	23 24	25 26	27	28	29 30	31 32	33 34	35 36	37 38 39 40	41	42	43	44	45	46	47	48 49	50 51	52 53	54 55	56 57	58 59	60 61 62 63	64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	80
	S	FORMAT1																													

D

Sequence Number	Form Type	Field Name: WSU Field Name	Field Name: Not Used by WSU	Field Length	Starting Location: Line Number	Starting Location: Horizontal Position	Output Data	Edit Code (WSU Only)	Input Allowed	Data Type	Mandatory Fill	Mandatory Entry	Self-Check	Adjust/Fill	Position Cursor	Enable Dup	Controlled Field Exit	Auto Record Advance	Protect Field	High Intensity	Blink Field	Nondisplay	Reverse Image	Underline	Column Separators	Reserved	Constant Type	Constant Data	Continuation
1 2 3 4 5	6	7 8 9 10 11 12	13 14	15 16 17 18	19 20	21 22	23 24	25	26	27	28	29	30	31	32	33 34	35	36	37 38	39 40	41	42 43	44	45 46	47 48	49 50 51 52 53 54 55	56	57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	80
	D	HEAD1		0013	02	06	Y														Y							FIRST DISPLAY	
	D	HEAD2		0040	03	06	Y																				M10	01U1	
	D	WUMTXT		0017	05	01	Y																					CUSTOMER NUMBER-	
	D	CUSNO		0006	05	18	Y																						
	D	NAMTXT		0014	07	01	Y																					CUSTOMER NAME-	
	D	NAME		0020	07	16	Y																						
	D	CODTXT		0012	09	01	Y																					SHIP-TO-CODE	
	D	CODE		0001	09	14			Y	Y				Y															
	D	BALTXT		0011	11	01																						BALANCE DUE	
	D	BALNCE		0005	11	13	Y		Y																				
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												
	D																												

Figure 4-11. Filled-Out Display Screen Format Specifications

$SLFL–SETFILE Utility Program

Function

The $SLFL utility program sets a disk file characteristic that enables immediate access to added records when processing is done sequentially by key.

This operation does not involve copying the file to a new disk location.

Utility Control Statement Format

// SETFILE LABEL–file label ,DATE–{mmddyy / ddmmyy / yymmdd} ,IFILE–{YES / NO}

Parameters

LABEL: The label of the file that is to have its disk file characteristic set.

DATE: The creation date of the file to be set.

IFILE: Specifies that the file is assigned the IFILE characteristic. A file with the IFILE characteristic gives index sequential users immediate access to added records.

Control Statement Sequence

```
// LOAD $SLFL
// RUN
// SETFILE . . .
// END
```

Example

Set a file to an IFILE:

```
// LOAD $SLFL
// RUN
// SETFILE LABEL-file1,DATE-070480,IFILE-YES
// END
```

$UASC–CRT Display of Spool File Entries

Function	The $UASC utility program displays a spool file entry or entries that are in expanded format at a display station.
Utility Control Statement Format	None
Parameters	None

Control Statement Sequence	// LOAD $UASC
	// FILE NAME-name,DISP-SHR
	// RUN

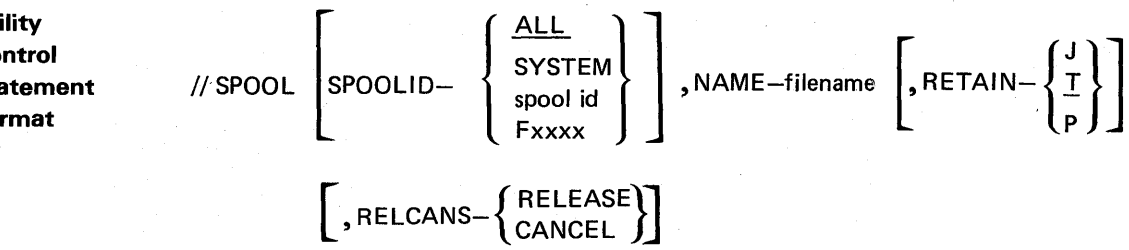
$UASF–User Access to Spool File Utility Program

Function

The $UASF utility program copies spool file entries from the spool file to a disk file.

$UASF is executed when the COPYPRT procedure is run.

Utility Control Statement Format



Parameters

SPOOLID: Specifies which spool entries are to be copied.

ALL: All entries are copied that are not being processed by spool and that have user IDs matching the user ID of the display station operator.

SYSTEM: If security is not active, copy all spool file entries not being processed by spool. If security is active, the operator must have a security classification of system operator or higher to copy all spool file entries.

spool-id: The 6-character spool file entry ID of the spool file entry to be copied. If security is not active, the display station operator can copy any spool file entry. If security is active, the user ID associated with the spool file entry must be the same as the user ID of the display station operator, unless the operator has a security classification of system operator or higher.

Fxxxx: xxxx is the 1- through 4-character forms ID of the entry or entries to be copied. (For more information about specifying forms numbers, see the CHANGE PRT command.) If security is not active, the display station operator can copy any entry or entries with the specified forms number. If security is active, the user ID associated with the spool file entry or entries must be the same as the user ID of the display station operator, unless the operator has a security classification of system operator or higher. Only entries not being processed by spool can be copied.

NAME: Specifies the label of the file that will contain the copied spool file entries. NAME is a required parameter when OCL statements are used to invoke the $UASF utility.

RETAIN: Classifies the file as a job file (RETAIN-J), a temporary file (RETAIN-T), or a permanent file (RETAIN-P). If the RETAIN parameter is not specified, RETAIN-T is assumed.

RELCANS: Specifies RELEASE to release spool entries after copy or CANCEL to cancel after copy.

No more than 255 spool file entries can be copied. For each entry in the spool file copied into the data file, one header record of 150 bytes is built (see note). The header record contains the following fields:

Beginning Column	Field Length	Content
1	1	H
4	6	spool-id
12	8	procedure name
22	8	job name
32	8	user-id
42	8	printer file name
52	2	logical printer ID
56	4	forms number
61	2	number of copies to be printed (binary)
65	2	number of pages (binary)
69	4	record count (binary)
74	2	lines per page (binary)
78	1	contains I if this spool entry contains IGC characters
81	1	contains M if this spool entry contains print records with a length greater than 132

Following the header record are 150-character data records with the following fields:

Beginning Column	Field Length	Content
1	2	Page number (binary)
3	2	Line number (binary)
5	4	Record number (binary)
9	1	contains I if this record contains IGC characters
10	1	contains hex 0E if first character in print field is IGC
11	132	Expanded data record (see note)

All unused fields are set to blanks.

Note: If print line length is greater than 132, the record length is 215 and the expanded data record length is greater than 132.

When the $UASF utility is initiated, the entry or entries to be copied are marked so that they cannot be printed by the spool writer until copying is complete.

Control Statement Seqeunce

```
// LOAD $UASF
// RUN
// SPOOL . . .
// END
```

Example

Copy all entries into a file labeled COPYALL:

$XNLM—X.21 Phone List Utility

Function

The $XNLM utility program provides a way to interactively build or change a list of phone numbers for an X.21 public data network. Each list can contain up to 84 numbers for the public data network. Each list is stored as a library load member. For information about how System/34 uses a list of phone numbers for an X.21 public data network, see the *Interactive Communications Feature Reference Manual* and the *Data Communications Reference Manual.*

The $XNLM utility program is run when the DEFINX21 procedure is entered.

When the $XNLM utility program is run, you can choose either to build a new list of phone numbers or to change an existing list. You are then prompted for the name of the list and name of library in which the list is to be stored. The default library is #LIBRARY. After the list name and the library name are entered, a screen is displayed allowing you to build a list or to change numbers within an existing list:

```
                        3.0 DEFINX21 PHONE LIST

                             Build a phone list

        Connection number      Retry value       Delay value
        1-18 digits,DC,I,D        1-255             0-16




Enter-Update   CMD7-EOJ   Help-More information
```

The following information can be specified for each number in the list:

- A number consisting of 1 through 18 characters. Valid characters are the numeric digits 0 through 9, slashes (/), hyphens (-), periods (.), and commas (,). The phone number must be left-justified.

  The characters DC in positions 1 and 2, with no characters following, indicate a direct call. The direct call capability is offered by public data networks to simplify the call procedure for a subscriber who is always connected to the same remote system. On calling, System/34 presents to the network the direct call sequence instead of the remote system's number. The network connects the calling System/34 to the remote system.

- A retry value specifying the total number of times an attempt to call the number should be made. Allowable values are 001 through 255; the default value is 001. The retry value must be right-justified and zerofilled.

- A delay value in seconds indicating the time to wait before attempting a retry to a number that could not be reached. Allowable values are 00 through 16; the default value is 00 seconds. The delay value must be right-justified and zerofilled.

  Note: The delay function is invoked only if the call failed to connect on the last attempt because of a CCITT Group Code 2 (no connection, number busy) error or a CCITT Group Code 6 (network congestion) error.

The $XNLM utility program verifies that allowable values have been entered into the above fields. The load member is built or updated using the specified list of numbers and is saved in the specified library.

When building or changing a list of phone numbers for an X.21 public data network, you can delete or insert numbers. To delete a number within the list, move the cursor to the first position of the number to be deleted. Place a D in the first position of the number, and press the Enter/Rec Adv key. The number is deleted.

To insert a number within the list, move the cursor to the first position of the first number that precedes the number to be inserted, and press the Enter/Rec Adv key. A number with the characters DC, and a retry value of 001 and a delay value of 00 is inserted. Replace the direct call characters with the proper number and the retry and delay values with the proper values.

Note: The $XNLM utility program will not accept an attempt to insert a number after the last number defined in the list.

The following command keys can be used:

Key	Function
Cmd key 3	Returns the screen to the previous display. If a build or change operation was being performed, and Cmd key 3 is used, the newly specified numbers are lost or the list is not updated. This command key cannot be used on the operation selection screen.
Cmd key 7	Ends the $XNLM utility program. If used during a build operation, the list of phone numbers is created. If used during a change operation, the list is updated.
Cmd key 19	This command key can be used during a build or change operation. If this command key is used, the $XNLM utility program is ended, and any changes made to the list are ignored. The list is not created or updated.
Roll Up	Displays the next number, if one exists, in the list of phone numbers.
Roll Down	Displays the previous number, if one exists, in the list of phone numbers.

Utility Control Statement Format

Utility control statements are not used.

Control Statement Sequence

```
// LOAD $XNLM
// RUN
```

Example

To build a list of phone numbers, named PHONEX21 and stored in library COMMLIB, for an X.21 public data network, enter:

```
// LOAD $XNLM
// RUN
```

Then do the following:

- Select option 1 to build the list.
- Enter the name of the list, PHONEX21, and the name of the library in which the list is to be stored, COMMLIB.
- Enter the numbers, retry values, and delay values for the list as shown in the following screen:

```
                    3.0 DEFINX21 PHONE LIST

                         Build a phone list

  Connection number      Retry value       Delay value
  1-18 digits,DC,I,D       1-255             0-16

  15075553131                001               00

  16125554745                003               05

  DC                         001               00



Enter-Update   CMD7-EOJ   Help-More information
```

- Press Cmd key 7 to end the build operation and the $XNLM utility program.

$XREST–Extended Character File Restore Utility (for the Ideographic Version of the SSP)

Function

The $XREST utility program restores from diskette the extended character file. The restored file is placed in a disk file labeled #EXTN. The diskette contains the IBM-supplied extended characters and any user-generated extended characters.

The $XREST utility program can only be used for files created by $XSAVE.

The $XREST utility program can be run only from the system console and cannot be run while any other jobs are being run.

You can use the Character Generator Utility to enter ideographic characters into the extended character file. For information about how to use the Character Generator Utility, see the *Character Generator Utility User's Guide and Reference Manual.*

$XREST is executed when the XREST procedure is run.

Utility Control Statement Format

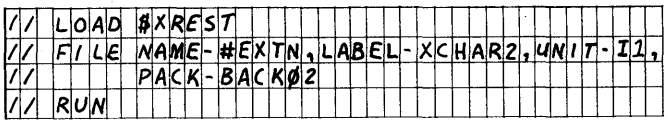
Utility control statements are not used.

Control Statement Sequence

```
// LOAD $XREST
// FILE NAME-#EXTN,UNIT-I1, . . .
// RUN
```

Example

Restore the extended character file from a diskette labeled BACK02. The label of the file on diskette is XCHAR2:



$XSAVE–Extended Character File Save Utility (for the Ideographic Version of the SSP)

Function

The $XSAVE utility program saves the contents of the extended character file on diskette. For information about how the extended character file is generated, see the RELOAD procedure.

$XSAVE is executed when the XSAVE procedure is run.

Utility Control Statement Format

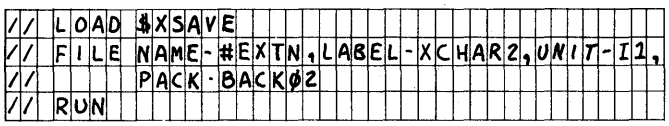
Utility control statements are not used.

Control Statement Sequence

```
// LOAD $XSAVE
// FILE NAME-#EXTN,UNIT-I1, . . .
// RUN
```

Example

Save the extended character file on a diskette labeled BACK02. The label of the diskette file should be XCHAR2:



Chapter 5. Writing and Using Procedures

A procedure is a set of OCL statements and (possibly) utility control statements that are inserted into the system input job stream via an INCLUDE OCL statement or a special form of the INCLUDE OCL statement called a procedure command. Procedures make it possible to avoid recoding and rekeying frequently used statements each time they are required. To be called by an INCLUDE statement or by a procedure command statement, a procedure must reside in a library as a procedure member.

This chapter describes how you can write and use your own procedures. For information about using SSP procedures, see Chapter 2 and Appendix D.

CREATING A PROCEDURE

The operator can use the $MAINT utility program or some other program, such as the source entry utility (described in the *Source Entry Utility Reference Manual*), to enter the statements for a procedure member directly from the keyboard. For example, a simple procedure called PROCD is to contain three statements. You can use the $MAINT utility program to create the procedure member by entering the following statements:

```
// LOAD $MAINT
// RUN
// COPY FROM-READER,LIBRARY-P,NAME-PROCD,TO-F1
// LOAD PROG2
// FILE NAME-FILEIN,LABEL-WEEKLY,DISP-SHR
// RUN
// CEND
// END
```

For information about reader-to-library operation, see the description of the $MAINT utility program in Chapter 4.

If the statements for a procedure member exist in a diskette file, the operator can use the $MAINT utility program to copy them, in record mode, into a procedure member. Remember, that when the diskette file is copied in record mode, the first statement in the file must be a COPY statement in the following format:

```
// COPY NAME-name,LIBRARY-P
```

where NAME specifies the procedure member name and LIBRARY-P specifies a procedure member. The last statement in the file must be:

```
// CEND
```

Note: For the formats of the COPY and CEND statements, see *Storing Library Members in Disk or Diskette Files* in Chapter 6 of this manual.

CALLING A PROCEDURE

Procedures can be called in three ways:

- By keying in an INCLUDE OCL statement (or a special form of the INCLUDE OCL statement called a procedure command)
- By selecting an item from a list of menu options
- By calling a procedure from another procedure

Keyboard Entry of the INCLUDE Statement

A procedure is usually called by a special form of the INCLUDE OCL statement known as a *procedure command*. A procedure command is formed by deleting the //, blanks, and INCLUDE from the format of an INCLUDE statement. The general format of a procedure command is:

```
procname [parameter-1,parameter-2, . . .parameter-n]
         [data                                     ]
```

A procedure command statement can begin in any position and can be followed by parameters or data to be passed to the procedure.

For example, keying

```
PAYROLL
```

and pressing the Enter/Rec Adv key is sufficient to call a procedure named PAYROLL, provided no parameters must be passed to the procedure.

For information about the other two formats permitted for an INCLUDE OCL statement, see the description of the INCLUDE OCL statement in Chapter 1.

Note: The //, blanks, and INCLUDE cannot be omitted from the INCLUDE statement if the procedure name is IF, IFT, IFF, ELSE, CANCEL, RETURN, RESET, GOTO, or TAG, or if the procedure name is the same as an OCL statement identifier or the same as a control command.

Calling a Procedure by Selecting an Item from a Menu

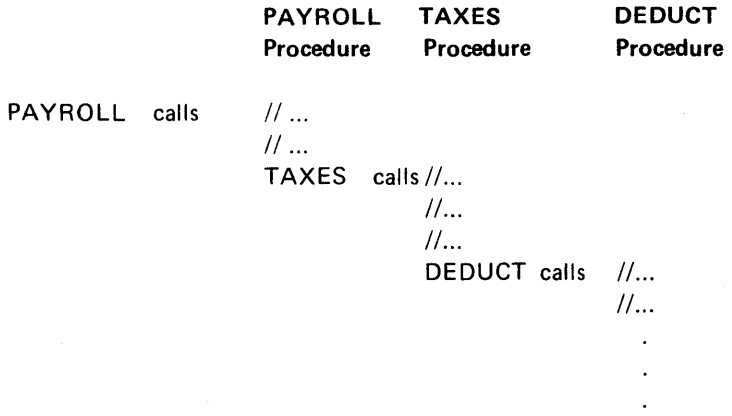
The operator can run a procedure by selecting an item from a menu. When the operator selects an item number, the system executes the statement that corresponds to that item number.

The menu items and the statements corresponding to them are defined by the programmer when the menu is created.

For information about creating a menu, see the description of the BLDMENU procedure in Chapter 2.

Calling a Procedure from Another Procedure

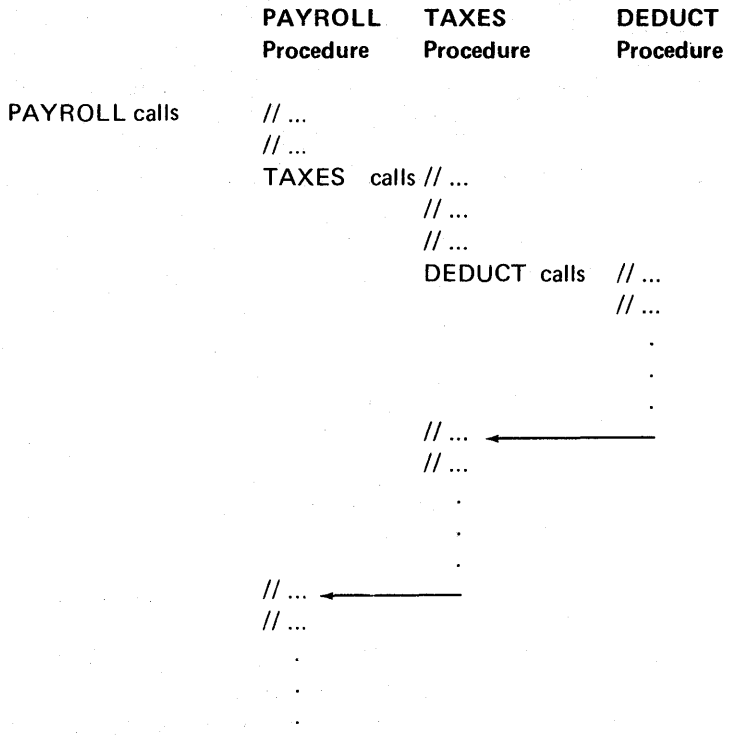
One procedure can call another procedure. Suppose, for example, that a procedure named PAYROLL contains (in addition to other OCL statements) a TAXES procedure command, and that another procedure named TAXES contains a DEDUCT procedure command. Both the TAXES procedure and the DEDUCT procedure are called and executed when the operator enters the PAYROLL procedure command.



A procedure called by another procedure is called a *nested procedure*. In the preceding example, TAXES and DEDUCT are nested procedures. A nested procedure always returns to the next OCL statement in the procedure that called it unless a RESET statement or CANCEL keyword is used within the calling procedure.

For a description of the RESET statement, see *RESET Statement* later in this chapter.

For a description of the CANCEL keyword, see *Conditional Expressions (IF and ELSE)* later in this chapter.



The preceding example contains three levels of procedures: the first level contains PAYROLL, the second level contains TAXES, and the third level contains DEDUCT. One level can contain more than one procedure command, but a maximum of 16 levels of procedures without a RESET statement are allowed in one job stream.

EXECUTING A PROCEDURE

When a non-MRT procedure name is recognized by the SSP, the following actions occur:

1. The procedure member corresponding to the procedure name is found in the active user library or the system library.

2. Then, the OCL statements, utility control statements, or nested procedures are read, one statement at a time, by the SSP. Parameters are substituted for substitution variables, conditional expressions are processed, and the resultant OCL and utility control statements from the original procedure and any nested procedures are executed as a normal job stream.

When an MRT procedure name is recognized by the SSP, the following actions occur:

1. The SSP checks if the requested procedure is active.

2. If the procedure is already active, the SSP attaches the requesting display station to the copy of the MRT program that is already loaded for that procedure (if the MRT max count is not exceeded).

3. If the procedure is not active, the SSP reads the OCL statements and utility control statements one at a time. The SSP processes substitution expressions and conditional expressions. The resulting OCL and utility control statements from the procedure are executed as a normal job stream.

If your program is run from a procedure, you can include substitution expressions, conditional expressions, and GOTO and TAG statements in the data read by RPG SUBR01. These expressions and statements can also be used in sort sequence specifications.

Substitution expressions and conditional expressions are described later in this chapter.

For information about MRT procedures and programs, see *MRT (Multiple Requestor Terminal) Programs* in the *Concepts and Design Guide*.

PROCEDURE PARAMETERS

Some procedures require parameters when the procedures are requested. A maximum of 11 parameters can be passed when calling a procedure. Parameters passed to procedures are *positional parameters*. Whenever a positional parameter appears in a statement, it must appear in the same position in relation to other positional parameters in the statement. If a valid positional parameter is omitted from a statement requesting a procedure but a following parameter is used, a comma must indicate the position reserved for the omitted parameter.

For example:

```
// INCLUDE PROCEDUR FILEA,,NO
```

The first parameter is FILEA, the second parameter is omitted, and the third parameter is NO. A fourth parameter, XYZ, is omitted but is not indicated by a comma because it is not followed by another parameter.

Some parameters have *defaults*. A default is a parameter that is substituted for an omitted parameter. You can specify defaults in your procedures.

For information about specifying defaults, see *Substitution Expressions* later in this chapter.

The blank (b), comma (,), single quote ('), question mark (?), slash (/), and hyphen (-) have special meanings in procedures, in OCL statements, and in utility control statements and should not be used in parameters for a procedure. The ?, /, and - should not be used in any control statement unless the format of the statement given in this manual indicates that one or more of the symbols are required. (For example, for OCL and utility control statements beginning with //, a hyphen is required to separate keywords and values in keyword parameters.)

Notes:

1. Parameters cannot be passed to MRT (multiple requestor terminal) procedures. However, data can be passed to an MRT program or an SRT (single requestor terminal) program on its first input operation from the requesting display station. Data passed to the program starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. For a description of SRT programs and MRT programs and procedures, see the *Concepts and Design Guide*.
2. If sequence numbers are used on INCLUDE OCL statements, the sequence numbers are treated as comments and the rules for coding comments apply (see *General OCL Coding Rules* in Chapter 1).
3. For the ideographic version of the SSP, procedure parameters cannot contain ideographic characters.

PROCEDURE ATTRIBUTES

When a procedure is created by the $MAINT utility program or by SEU (source entry utility), the procedure can be assigned the following attributes:

- MRT procedure. For information about MRT programs and procedures, see *MRT (Multiple Requestor Terminal) Programs* in the *Concepts and Design Guide*.

- Do not log OCL to history file. OCL statements from the procedure are not logged to the history file when the procedure is run. The INCLUDE OCL statement for the procedure is logged to the history file.

- Data is passed to the program. Parameters are not passed to procedures with this attribute. Instead, data on the INCLUDE OCL statement is passed to the first program run by the procedure. The data is passed on the first input request from the requesting display station. The data starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. Every MRT procedure has this attribute whether or not the attribute is selected when the procedure is created.

SUBSTITUTION EXPRESSIONS

Substitution expressions allow the programmer to substitute information into the statements generated when the procedure is run. Examples of information that can be substituted are:

- Positional parameters on the procedure command that called the procedure
- Information supplied by the operator in response to prompts issued from within the procedure
- Specified bytes in the display station local data area
- Specified bytes in a message in the USER1 message member (For an explanation of how to assign a USER1 message member, see the description of the MEMBER OCL statement in Chapter 1.)

If a procedure that displays prompts is placed on the input job queue, the prompts are displayed at the system console. In that case, the system operator must know the correct responses to the prompts.

Substitution expressions can be used while entering statements from the keyboard or in the command statement in a menu. However, a substitution expression for a positional parameter entered from the keyboard or contained in a menu results in a null substitution. A substitution expression for a nonpositional parameter results in the proper substitution. For example, if you enter the following statement from the keyboard:

```
// FILE NAME-?2'FILEA'?,DATE-?DATE?
```

and the current date is 021480, the following substitutions are generated:

```
// FILE NAME-,DATE-021480
```

Substitution is not performed for OCL comment statements, which are indicated by an asterisk (*) in position 1 of the statement.

Substitution Expression Formats

Substitution expressions always begin and end with a question mark. A substitution expression begins any time a question mark is immediately followed by a number or by one of the following characters: C, D, F, L, M, R, S, U, or W. A question mark followed by any other character is not treated as a substitution expression. Substitution or prompting occurs whenever a valid expression is encountered, even if the expression is in the comment portion of an OCL statement or in a message statement.

Following are descriptions of the substitution expression formats that can be used in a procedure member. Examples of how to use the formats are included.

Substitution Format	Meaning
?n?	Substitutes the value of the nth positional parameter. If the nth parameter is not defined, no value is substituted. For example, a procedure member contains the following statement:

```
// * '?3? WAS DELETED'
```

If the third parameter is not defined (that is, it was not specified on the procedure command and was not assigned a value by a previous statement within this procedure), the following statement is generated:

```
// * ' WAS DELETED'
```

If the value of the third parameter is FILEX, the following statement is generated:

```
// * 'FILEX WAS DELETED'
```

?nF'value'?

Forces a new value to be assigned to the nth positional parameter, whether or not the nth positional parameter was previously defined. The specified value is then substituted into the statement containing the expression. For example, a procedure member contains the following statements:

```
// PROMPT MEMBER-MBRL,FORMAT-?3F'FMT3'?
// * 'THE FORMAT DISPLAYED WAS ?3?'
```

The following statements are generated:

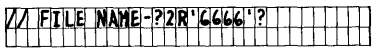
```
// PROMPT MEMBER-MBRL,FORMAT-FMT3
// * 'THE FORMAT DISPLAYED WAS FMT3'
```

To change a parameter so that it is no longer defined, the following format can be used:

?nF''?

Substitution Format	Meaning
?n'default'?	Substitutes the value of the nth positional parameter; if the nth parameter is not defined, permanently assigns a default value to the parameter and then substitutes that value. 'default' specifies the permanent default value. The length of the 'default' value can be no more than 8 characters. If the default value is assigned, subsequent references to the nth parameter within the procedure member use the default value. For example, a procedure member contains the following statement: If the second parameter was not defined, the following statement is generated: // FILE NAME-FILEIN Subsequent references to the second parameter use FILEIN.
?nT'default'?	Substitutes the value of the nth positional parameter; or, if the nth parameter is not defined, temporarily assigns a default value to the parameter and then substitutes the value. 'default' specifies the temporary default value. *A temporary default value is used only for the current substitution expression.* The length of the 'default' value can be no more than 8 characters. For subsequent references to the nth parameter within the procedure, the parameter is undefined. For example, a procedure member contains the following statements: If the second parameter was not defined, the following statements are generated: // FILE NAME-WEEKLY // * '2ND PARAMETER WAS NULL'

Substitution Format	Meaning
?nR'mic'?	Substitutes the value of the nth positional parameter; if the nth parameter is not defined, displays a message from the USER1 message member and waits for the operator to enter from the keyboard the value to be substituted. Subsequent references to the nth parameter within the procedure member use the value entered by the operator. R indicates that an operator reply is required if the parameter is not defined. 'mic' identifies the message identification code of the message to be displayed if the nth parameter is not defined. For example, a procedure member contains the following statement:



If the second parameter is not defined, the following message (message 6666 from the USER1 message member) is displayed:

ENTER THE NAME OF THE FILE

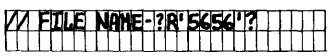
The operator then enters the word PAYROLL from the keyboard, and the following statement is generated:

// FILE NAME-PAYROLL

Subsequent references to the second parameter use PAYROLL.

?R'mic'?

Displays a message from the USER1 message member and waits for the operator to enter from the keyboard the value to be substituted. R indicates that an operator reply is required. 'mic' is the message identification code of the message to be displayed. For example, a procedure member contains the following statement:



When this statement is processed during execution of the procedure, the SSP displays message 5656 from the USER1 message member. The operator then enters the word INFILE from the keyboard and the following statement is executed:

// FILE NAME-INFILE

Substitution Format	Meaning
?nR?	Substitutes the value of the nth positional parameter; if the nth parameter is not defined, displays a system message (ENTER MISSING PARAMETER) and waits for the operator to enter the value to be substituted. Subsequent references to the nth parameter within the procedure member use the value entered by the operator. R indicates that an operator reply is required if the parameter is not defined. For example, a procedure member contains the following statement:

```
PAYROLL WEEKLY,?1R?
```

If the first parameter is not defined, ENTER MISSING PARAMETER is displayed. The operator then enters the word SALARY from the keyboard, and the following statement is generated:

```
PAYROLL WEEKLY,SALARY
```

Note: Within the procedure member named PAYROLL, references to the second positional parameter use the word SALARY.

?R? — Displays a system message (ENTER REQUIRED PARAMETER) and waits for the operator to enter from the keyboard the value to be substituted. R indicates that an operator reply is required. For example, a procedure member contains the following statement:

```
// FILE NAME-?R?
```

When the statement is being generated, ENTER REQUIRED PARAMETER is displayed. The operator then enters INFILE, and the following statement is generated:

```
// FILE NAME-INFILE
```

Substitution Format	Meaning
?WS?	Substitutes the 2-character work station ID of the display station that called the procedure. For example, a procedure statement contains the following statement:

```
// FILE NAME-JFILE,LABEL-?WS?FILE
```

If the procedure is called from display station W3, the following statement is generated:

```
// FILE NAME-JFILE,LABEL-W3FILE
```

If the procedure is run from the input job queue, the work station ID of the display station from which the job was submitted is substituted.

If the procedure is evoked using // EVOKE, the work station ID is the original work station ID.

If the procedure is evoked using ICF EVOKE, the work station ID is the session ID (00-99).

?USER? Substitutes the 1- through 8-character user ID assigned to the operator that submitted the job. For example, a procedure contains the following statement:

```
// FILE NAME-?USER?FILE
```

If the user ID of the operator is DMM, the following statement is generated:

```
// FILE NAME-DMMFILE
```

Substitution Format	Meaning
?L'offset,lng'?	Substitutes a value from the 256-byte display station local data area. (For information on modifying the display station local data area, see *LOCAL Statement* in Chapter 1.) L indicates that a value is to be substituted from the local data area; offset specifies the displacement (a decimal number from 1 through 256, where 1 specifies the first byte in the local data area) into the local data area of the first byte to be substituted; and lng specifies the decimal number of bytes in the value to be substituted. Beginning blanks and embedded blanks are allowed in the substituted value, but trailing blanks are truncated. For example, a procedure member contains the following statement:

If positions 36 through 43 of the local data area contain FILEAbbb, the following statement is generated:

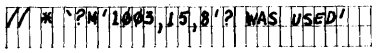
```
// FILE NAME-FILEA,LABEL-LCZ
```

CAUTION
The RPG II procedures (AUTO, RPG, and RPGR) use bytes 201 through 256 of the display station local data area, the work station utility generation procedure (WSU) uses bytes 238 through 256 of the local data area, the screen design aid procedure (SDA) uses bytes 1 through 104 of the local data area, the HELP procedure uses bytes 249 through 256 of the local data area, and the system measurement facility procedures (SMFSTART and SMFPRINT) use bytes 220 through 256 of the local data area. Any user data in those bytes is destroyed when one of those procedures is run.

Note: A local substitution expression can appear anywhere within a procedure and can be specified with any allowable length. However, if the substituted data is then used in a procedure control expression that has a length restriction, the data must adhere to that restriction.

CAUTION
If the local data area contains IGC data, care should be taken to ensure that an equal number of shift outs and shift ins are substituted. Results will be unpredictable, otherwise.

Substitution Format	Meaning
?M'xxxx,offset,lng'? or ?Mxxxx?	Substitutes a value from a record in the USER1 message member. (For information about assigning a USER1 message member, see the description of the MEMBER OCL statement in Chapter 1.) M indicates a value is to be substituted from a record in the message member; xxxx specifies the message identification code of the record containing the value; offset specifies the displacement (a decimal number from 1 through 75, where 1 specifies the first byte in the record) into the record of the first byte to be substituted; and lng specifies the decimal number of bytes in the value to be substituted. If offset and lng are not specified, the entire record text is substituted. Beginning blanks and embedded blanks are allowed in the substituted value, but trailing blanks are truncated. For example, a procedure member contains the following statement:



If positions 15 through 22 of record 1003 contain FILEAbbb, the following statement is generated:

```
// * 'FILEA WAS USED'
```

Note: A message member substitution expression can appear anywhere within a procedure and can be specified with any allowable length. However, if the substituted data is then used in a procedure control expression that has a length restriction, the data must adhere to that restriction.

CAUTION
If the message contains IGC data, care should be taken to ensure that an equal number of shift outs and shift ins are substituted. Results will be unpredictable, otherwise.

Note: For information about retrieving ideographic messages from the message member, see *Considerations for the Ideographic Version of the SSP* described in the $MGBLD utility program in Chapter 4.

Substitution Format	Meaning
?CD?	Substitutes a 4-character return code set by the SSP or program product. The possible return codes are:

Return Code	Meaning
0000	The previous job step terminated normally or this step is the first step in the job.
1002	Warning errors were encountered in COBOL compile.
1004	Conditional errors were encountered in COBOL compile.
1008	Terminal errors were encountered in the previous job step (ASM, COBOL, FORMAT, FORTRAN, OLE, RPG, or WSU).
1312	The previous job step was an MRT program that terminated without releasing the display station.
1990	The $HELP utility was requested to run an MRT procedure.

Substitution Format	Meaning
?CD? (continued)	Substitutes a 4-character return code set by the SSP or program product. The possible return codes are:

Return Code	Meaning
1991	Operator canceled $HELP using Cmd key 7.
2001-2024	Command keys 1 through 24 returned from the PROMPT screen.
2090	Roll Up key.
2091	Roll Down key.
2092	Help key.
2093	Record Backspace key.
3721	The operator canceled the previous job step by selecting the 2 option in response to a message, or the previous step was an MRT program and the operator released the display station by interrupting the MRT program (by pressing the Attn key) and then selecting the 2 option.
8158	The SSP-ICF session abnormally ended.

In the following example, a procedure called OPTERM is run if the return code is 3721. Otherwise, a procedure called PROC2 is run.

```
// IF ?CD?/3721 OPTERM
// ELSE PROC2
```

Note: The SSP resets the return code to 0000 whenever it executes a RUN OCL statement.

Substitution Format	Meaning

?SLIB?	Substitutes the name of the active library for the session. For example, the active library for a session is #LIBRARY, and a procedure member contains the following statement:
	
	`// JOBQ ?SLIB?,PROCA`
	
	The following statement is generated:
	
	`// JOBQ #LIBRARY,PROCA`
	
	Note: Even if the procedure member contains a LIBRARY OCL statement that names a different library, #LIBRARY would still be substituted because a LIBRARY OCL statement within a procedure does not change the active library for the session. A LIBRARY OCL statement within a procedure changes only the current active library (the active library for the procedure).

?CLIB?	Substitutes the name of the current active library. For example, a procedure member contains the following statements:
	
	`// LIBRARY NAME-USERLIB`
	`// JOBQ ?CLIB?,PROCB`
	
	The following JOBQ statement is generated:
	
	`// JOBQ USERLIB,PROCB`

?DATE?	Substitutes the current program date. For example, a procedure member contains the following statement:
	
	`// FILE NAME-FL?DATE?`
	
	If the current program date is 021479, the following statement is generated. Note that the format is the current session date format:
	
	`// FILE NAME-FL021479`

Substitution Format	Meaning
?F'S,label,date'? or ?F'S,label'?	Substitutes the number of blocks or records allocated for a P or T disk file. The value substituted is given in the units (blocks or records) specified when the file was created. The result of the substitution is an 8-digit number with leading zeros. If the files do not exist on disk, or the file is a J or S file, 00000000 is substituted. If 2 or more files exist on disk with the same name, the size of the file with the most recent creation date is substituted. F indicates that file information is being substituted; S indicates that the specified size will be substituted; label is the label of the file; and date is the creation date of the file. For example, a procedure contains the following statement:

```
// FILE NAME-FILE2,BLOCKS-?F'S,FILE1'?,DISP-OLD
```

If FILE1 was allocated with a specified size of 50 blocks, the following statement will be generated for FILE2:

```
// FILE NAME-FILE2,BLOCKS-00000050,DISP-OLD
```

Substitution Format	Meaning
?F'A,label,date'? or ?F'A,label'?	Substitutes the actual number of data records in a P or T disk file, as recorded in the disk VTOC. The result of the substitution is an 8-digit number with leading zeros. If the files do not exist on disk, or the file is a J or S file, 00000000 is substituted. If 2 or more files exist on disk with the same name, the size of the file with the most recent creation date is substituted. F indicates that file information is being substituted; A indicates that the actual number of records will be substituted; label is the label of the file; and date is the creation date of the file. For example, a procedure contains the following statements:

```
// FILE NAME-COPYO,UNIT-F1,LABEL-FILE2,
//      RECORDS-?F'A,FILE1'?
```

If FILE1 actually contains 150 date records, the following statements are generated for FILE2:

```
// FILE NAME-COPYO,UNIT-F1,LABEL-FILE2,
//      RECORDS-00000150
```

Notes:

1. These types of substitution, when used with a FILE OCL statement, may result in an error condition if the file did not exist or if the file is empty.
2. If records are being added to a file that is currently in use, the disk VTOC does not reflect the added records until the file is closed; therefore, the substituted number of records will not match the actual number of records in the file.

Nested Substitution Expressions

System/34 allows nesting of substitution expressions. For example, a procedure contains the following statement:

```
?M 1234,1,29'?
```

This expression specifies that the first 29 characters of message number 1234 should be substituted. In this example, the first 29 characters of message 1234 are:

```
// FILE NAME-?WS?FILE,UNIT-I1
```

Therefore, the statement that results from the first substitution expression contains another substitution expression. The SSP then processes the latter expression. If the work station ID of the work station that submitted the job is W1, the resulting statement is:

```
// FILE NAME-W1FILE,UNIT-I1
```

By using nested expressions, you can use a substitution expression as a default value within another expression. For example,

```
?1'?2?'?
```

instructs the SSP to substitute the value of the first parameter. If the first parameter is not defined, the SSP substitutes the value of the second parameter.

When using nested substitution expressions, take care to avoid expressions that might cause an infinite loop. For example, a procedure contains the following statement:

```
// FILE NAME-?1R?,UNIT-I1
```

If the operator did not specify the first parameter, he is prompted to enter the first parameter value. If he responds by entering ?1?, the SSP will continually substitute ?1? into the statement. The operator will have to interrupt and then cancel the procedure.

As another example, a procedure contains the following statement:

```
// LOCAL OFFSET-1,DATA-?M'1234,1,14'?
```

This expression specifies that the first 14 characters of message member 1234 should be substituted. In this example, the first 14 characters of message member 1234 are:

?M'1234,1,14'?

This statement causes the SSP to continually substitute ?M'1234,1,14'? into the statement. Again, the operator will have to interrupt and cancel the procedure.

Note: A message member substitution expression can appear anywhere within a procedure and can be specified with any allowable length. However, if the substituted data is then used in a procedure control expression that has a length restriction, the data must adhere to that restriction.

CONDITIONAL EXPRESSIONS (IF AND ELSE)

Conditional expressions within a procedure member allow the programmer to conditionally generate certain OCL and utility control statements when the procedure is run. Two types of conditional expressions can be used: IF expressions and ELSE expressions.

IF Expressions

The IF expression can be used only within a procedure member. An IF expression tests for a specified condition; if the condition is met, a specified statement is generated.

If you locate an IF statement between a LOAD and a RUN statement and the IF condition is met, it may cause the SSP to perform a function which is not allowed between a LOAD and a RUN statement (for example, loading a system utility to delete a file).

An IF expression can have any of the following formats:

```
// IF condition-parameter statement-parameter

// IFT condition-parameter statement-parameter

// IFF condition-parameter statement-parameter
```

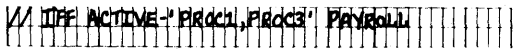
IF and IFT test if the condition specified by condition-parameter is true. If the specified condition is true, the statement specified by statement-parameter is generated.

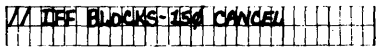
IFF tests if the condition specified by condition-parameter is false. If the specified condition is false, the statement specified by the statement-parameter is generated.

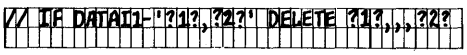
CAUTION
Continuation of a statement following an IF expression onto two or more lines will invalidate the continuation expressions and any subsequent ELSE expressions.

Condition-Parameter:

Following is a list of the possible formats for the condition-parameter. With each format is an explanation of the true condition and, where necessary, an example of how the format is used. A condition-parameter must be a continuous string of nonblank characters. A blank indicates the end of the condition-parameter.

Condition-Parameter Format	**Explanation**
ACTIVE-procedure name	True if the specified procedure is currently running on the system. For example, specifies that if a procedure called PROC1 is active, the system should return to the calling procedure for the next statement. See the description of the RETURN keyword, which follows this list of conditional parameters.
ACTIVE-'proc1,proc2 . . . procn'	True if any of the specified procedures are currently running on the system. For example,  specifies that if neither of the specified procedures (PROC1 or PROC3) is active, the procedure called PAYROLL should be run. *Notes:* 1. If the condition is true, the specified procedures were active at the time the ACTIVE test was evaluated. 2. Processing based on the ACTIVE test should be done with caution because the condition being tested for may have changed. 3. If you attempt to test the procedure that contained the ACTIVE test, the test will always be false. 4. The ACTIVE test cannot be used to determine if you are the last user of an MRT. 5. If a procedure is evoked using the EVOKE OCL statement, and an ACTIVE test of that procedure immediately follows, the results of the test cannot be predicted. The system may or may not have had a sufficient amount of time to begin executing the tested procedure. Processing based upon this type of ACTIVE test is not recommended.

Condition-Parameter Format	Explanation
BLOCKS-nnnnn	True if the specified number of contiguous blocks is available on the disk. nnnnn can be a 1- to 5-digit decimal number. For example,  specifies that if 150 contiguous disk blocks are not available, the job should be canceled. See the description of the CANCEL keyword, which follows this list of conditional parameters. *Note:* If the condition is true, the specified number of blocks is available *at the time the BLOCKS expression is evaluated*. If other programs are running, another program might use that available space before your program tries to use it.
DATAI1-'label,date,location' or DATAI1-label	True if a file exists on the diskette with the name and creation date (optional) specified. If your system has a diskette magazine drive, the location parameter specifies the location of the diskette or diskettes to be searched. If you specify that multiple magazines or diskette slots are to be searched, the system will search until it finds the first diskette that contains a file with the specified label and the creation date. The location parameter can be: • S1, S2, or S3, which identifies an individual diskette slot. If a location is not specified, S1 is assumed. • ALLS, which specifies that S1, S2, and S3 will be searched. • M1.01 through M1.10, which identifies a location within magazine M1. • M2.01 through M2.10, which identifies a location within magazine M2. • ALL1, which specifies that all diskettes in magazine M1 should be searched. • ALL2, which specifies that all diskettes in magazine M2 should be searched. • ALL, which specifies that all diskettes in both magazines should be searched.

Condition-Parameter Format	Explanation
DATAI1-'label,date,location' or DATAI1-label (continued)	For example,  specifies that if a diskette file with the label specified by the first parameter and the date specified by the second parameter exists on the diskette, the DELETE SSP procedure is run to delete that file. *Note:* After the test is made for a magazine drive, the diskette remains in the diskette drive until the magazine drive door is opened, a procedure is run which accesses the diskette drive, or the job step or job ends.
DATAF1-'label,date' or DATAF1-label	True if a permanent or temporary file exists on the disk with the name and creation date (optional) specified. For example, specifies that if a disk file with the label specified by the second positional parameter does not exist, the procedure should be canceled. See the description of the CANCEL keyword, which follows this list of conditional parameters. *Note:* This test will *not* detect a scratch file or a job file that has the specified name.
DSPLY-960	True if the procedure containing the expression is run from a 960-character display station. For example, specifies that the PROMPT OCL statement should display one of two formats: FMT960 for a 960-character display station or FMT1920 for a 1920-character display station. Both formats are in a load member called FORMATS. If this expression is processed by a procedure run from the input job queue, an error message is displayed, and the operator must cancel the job.

Condition-Parameter Format	Explanation
DSPLY-1920	True if the procedure containing the expression is run from a 1920-character display station. For example,

```
// IF DSPLY-1920 PROMPT MEMBER-FORMATS,FORMAT-FMT1920
// ELSE PROMPT MEMBER-FORMATS,FORMAT-FMT960
```

specifies that the PROMPT OCL statement should display one of two formats: FMT960 for a 960-character display station or FMT1920 for a 1920-character display station. Both formats are in a load member called FORMATS.

If this expression is processed by a procedure run from the input job queue, an error message is displayed, and the operator must cancel the job.

DSPLY-IGC

This expression is valid for the ideographic version of the SSP and is true if the procedure containing the expression is run from a display station that can display ideographic characters and that is in ideographic character mode. For example,

```
// IF DSPLY-IGC PROMPT MEMBER-FORMATS,FORMAT-FMT1
// ELSE PROMPT MEMBER-FORMATS,FORMAT-FMT2
```

specifies that the PROMPT OCL statement should display one of two formats: FMT1 for an ideographic display station in ideographic mode and FMT2 for a display station that cannot display ideographic characters or that is not in ideographic mode. Both formats are in a load member called FORMATS.

If this expression is processed by a procedure from the input job queue, an error message is displayed, and the operator must cancel the job.

Condition-Parameter Format	Explanation
EVOKED-YES	True if the procedure was evoked through SSP-ICF or by the EVOKE OCL statement. For example,

```
// IF EVOKED-YES * 'LISTLIBR PROCEDURE EXECUTING'
```

specifies that if the procedure containing this statement was evoked through SSP-ICF or by the EVOKE OCL statement, the message LISTLIBR PROCEDURE EXECUTING should be displayed.

EVOKED-NO

True if the procedure was not evoked through SSP-ICF or by the EVOKE OCL statement. For example,

```
// IF EVOKED-NO PAYPROC
```

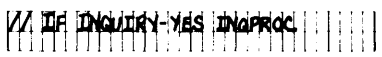
specifies that if the procedure containing this statement was not evoked through SSP-ICF or by the EVOKE OCL statement, the user procedure PAYPROC should be run.

LISTDONE-membername

This conditional expression is true if every number in the specified list has been called successfully or, for those numbers not called successfully, the retry count has been exhausted.

```
// TAG LABEL1
// LOAD PROG
// COMM LINE-1,PHONE-LIST1
// RUN
// IF LISTDONE-LIST1 RETURN
// ELSE GOTO LABEL1
```

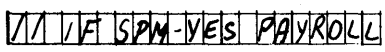
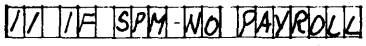
The program, PROG, will be executed in a loop, each time calling the next number in the list. When the last number has been called, or if all error retry counts have been exhausted, the IF expression will be true and control will be returned to the calling procedure.

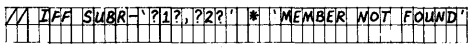
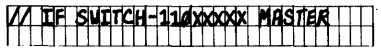
Condition-Parameter Format	Explanation
INQUIRY-YES	True if the procedure containing the expression is running under inquiry. That is, the operator used the Attn (attention) key to interrupt a program in order to run this procedure. For example,  specifies that if the procedure containing this statement is running under inquiry, the user procedure called INQPROC should be run.
INQUIRY-NO	True if the procedure is not running under inquiry. For example, specifies that if the procedure containing this statement is not running under inquiry, the user procedure called PROCA should be run.
JOBQ-YES	True if the procedure is run from the input job queue. For example, specifies that if the procedure containing the statement is run from the input job queue, the user procedure PAYPROC should be run, and the first parameter passed to PAYPROC should be JQ.
JOBQ-NO	True if the procedure is not run from the input job queue. For example, specifies that if the procedure containing this statement is not run from the input job queue, the user procedure PAYPROC should be run. No parameters are passed to the PAYPROC procedure.

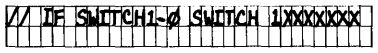
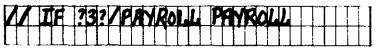
Condition-Parameter Format	Explanation
LOAD-'member,library' or LOAD-member	True if a load member (O member) with the specified name exists in the specified library (optional). If library is not specified, the condition is true if the specified load member exists in the system library (#LIBRARY). For example, `// IF LOAD-JOB2 JOBPROC` specifies that if a load member named JOB2 exists in the system library (#LIBRARY), a user procedure called JOBPROC should be run.

MRTMAX-procedure name	True if the specified MRT procedure has the maximum number of requestors attached. For example,
	
	`// IF MRTMAX-PROCB * 1001`
	`PROCB`
	
	specifies that if PROCB has the maximum number of requestors attached, message 1001 (a message indicating a possible delay) is displayed before PROCB is run.

PROC-'member,library' or PROC-member	True if a procedure member (P member) with the specified name exists in the specified library (optional). If library is not specified, the condition is true if the specified procedure member exists in the system library (#LIBRARY). For example, `// IF PROC-PROCA * 'PROCA NOT IN LIBRARY'` specifies that if a procedure member named PROCA is not in the system library (#LIBRARY), the message PROCA NOT IN LIBRARY should be displayed.

Condition-Parameter Format	Explanation
SOURCE-'member,library' or SOURCE-member	True if a source member (S member) with the specified name exists in the specified library (optional). If library is not specified, the condition is true if the specified source member exists in the system library (#LIBRARY). For example, specifies that if a source member named SOURCE1 does not exist in a library named NEWLIB, a user procedure called BLDSRC should be run.
SPM-YES	True if the procedure containing the expression is run in single program mode. For example:  specifies that if the procedure containing the statement is run in single program mode, the user procedure PAYROLL should be run. No parameters are passed to the PAYROLL procedure.
SPM-NO	True if the procedure containing the expression is not run in single program mode. For example:  specifies that if the procedure containing the statement is not run in single program mode, the user procedure PAYROLL should be run. No parameters are passed to the PAYROLL procedure.

Condition-Parameter Format	Explanation
SUBR-'member,library' or SUBR-member	True if a subroutine member (R member) with the specified name exists in the specified library (optional). If library is not specified, the condition is true if the specified subroutine member exists in the system library (#LIBRARY). For example,  specifies that if the subroutine member specified by the first positional parameter is not in the library specified by the second positional parameter, the message MEMBER NOT FOUND is displayed.
SWITCH-indicator-settings	True if all eight external indicators for the display station are in the state indicated by the indicator-settings value. The indicator-settings value consists of 8 characters, one for each of the eight external indicators (U1 through U8). For each of the 8 positions in the indicator-settings value, one of the following characters must be used: **Character Meaning** 0 The corresponding indicator must be off for the condition parameter to be true. 1 The corresponding indicator must be on for the condition parameter to be true. X The corresponding indicator is not checked. For example,  specifies that the procedure called MASTER should be run if indicators U1 and U2 are on and indicator U3 is off (the other indicators are not checked).

Condition-Parameter Format	Explanation
SWITCH1-0	True if external indicator U1 is off. For example,  specifies that if indicator U1 is off, it should be turned on. *Note:* The first 3 characters (//b) of an OCL or a utility control statement are not repeated when the statement is used as the statement parameter in an IF expression. For information about the statement parameter, see *Statement Parameter* later in this discussion.
SWITCH1-1	True if external indicator U1 is on.
•	
•	
•	
SWITCH8-0	True if external indicator U8 is off.
SWITCH8-1	True if external indicator U8 is on.
string1/string2	True if string1 is equal to string2. Each character string can be up to 8 nonblank characters long. For example,  specifies that if the third positional parameter is PAYROLL, the procedure called PAYROLL should be run. The two strings being compared are padded on the right with blanks to a length of 8, and the comparison is performed on a character-by-character basis, starting with the leftmost character. The comparison is performed this way for both alphabetic and numeric strings.

Condition-Parameter Format	Explanation
string1/string2 (continued)	The condition shown in the following example will result in a diagnostic message because of the imbedded blank in the character string:

```
// IF 1235b123/ RETURN
```

This situation may occur as the result of a substitution from a message member or a local data area.

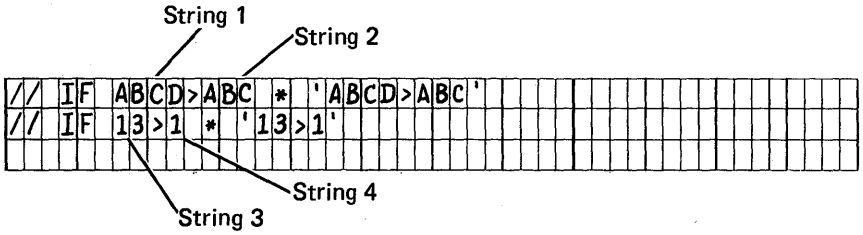
string1>string2

True if string 1 is greater than string 2. Each character string can be up to 8 nonblank characters long. For example:

```
// IF ?4?>1ØØ MASTER
```

specifies that if the value of the fourth positional parameter is greater than 100, the procedure called MASTER should be run.

If the either of the two values being compared has any alphabetic characters, the two strings are padded on the right with blanks to a length of 8, and the comparison is performed on a character-by-character basis, starting with the leftmost character. If the both of the values being compared have all numeric characters, the two strings are padded on the left with decimal zeros to a length of 8, and the comparison is performed on a character-by-character basis, starting with the leftmost character. For example,



would be evaluated as:

string 1: ABCDbbbb
string 2: ABCbbbbb
string 3: 00000013
string 4: 00000001

Condition-Parameter Format	Explanation

VOLID-'vol-id,location'
or
VOLID-vol-id
	True if the volume ID of the inserted diskette is the same as the volume ID specified. If your system has a diskette magazine drive, the location specifies the location of the diskette to be checked. If you specify that multiple magazines or diskette slots are to be searched, the system will search until it finds the first diskette with the specified volume ID. The location parameter can be:
	
	• S1, S2, or S3, which identifies an individual diskette slot. If no location is specified, S1 is assumed.
	
	• ALLS, which identifies that S1, S2, and S3 will be searched.
	
	• M1.01 through M1.10, which identifies a location within magazine M1.
	
	• M2.01 through M2.10, which identifies a location within magazine M2.
	
	• ALL1, which specifies that all diskettes in magazine M1 will be searched.
	
	• ALL2, which specifies that all diskettes in magazine M2 will be searched.
	
	• ALL, which specifies that all diskettes in both magazines will be searched.
	
	If the diskette drive or specified location is empty, a message will be displayed. The operator can then insert a diskette into the specified location or cancel the job.
	
	For example,
	
	`// IFF VOLID-VOL001 * 'INSERT CORRECT DISKETTE'`
	`// PAUSE`
	
	specifies that if the volume ID of the inserted diskette is not VOL001, the message INSERT CORRECT DISKETTE should be displayed.
	
	Note: After the test is made for a magazine drive, the diskette remains in the diskette drive until the magazine drive door is opened, a procedure is run which accesses the diskette drive, or the job step or job ends.

Statement Parameter: The statement parameter in an IF expression can be any of the following:

- An OCL statement without the 3 characters (//b) that normally introduce an OCL statement.
- A procedure command that calls another procedure.
- A utility control statement without the 3 characters (//b) that normally introduce a utility control statement.
- Another IF expression. For example,

```
// IF PROC-PAYROLL IF SWITCH3-1 PAYROLL
```

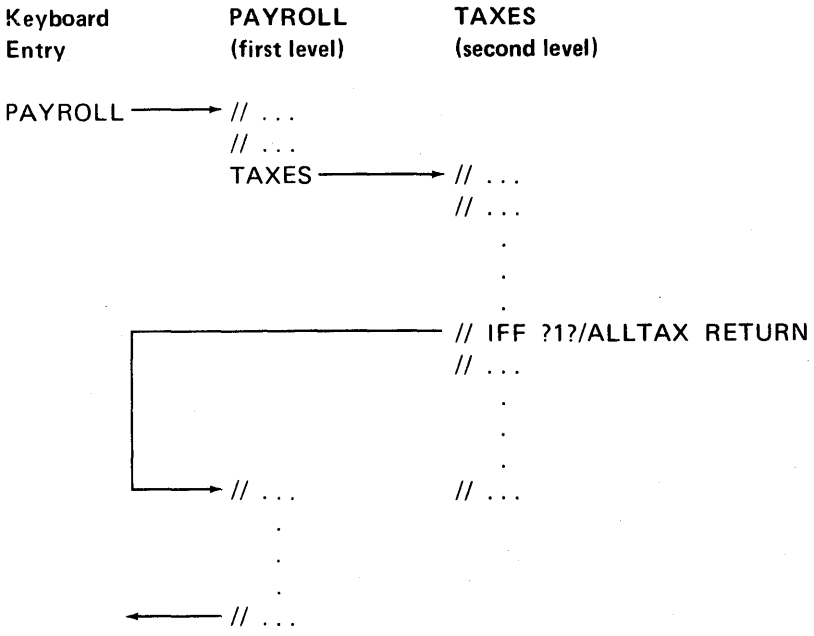
  specifies that if a procedure member named PAYROLL is in the system library and if indicator U3 is on, the PAYROLL procedure should be run.

- A RESET statement or GOTO statement (described later in this chapter) without the 3 characters (//b) that normally introduce those statements.
- A CANCEL or RETURN keyword. The CANCEL keyword cancels the entire job. If a CANCEL keyword is used to cancel an MRT procedure, an error message is issued to all active requestors of that procedure. The RETURN keyword, if encountered in a first-level procedure, causes an immediate return to the keyboard for the next control statement. The RETURN keyword, if encountered in a nested procedure, causes a return to the calling procedure for the next control statement.

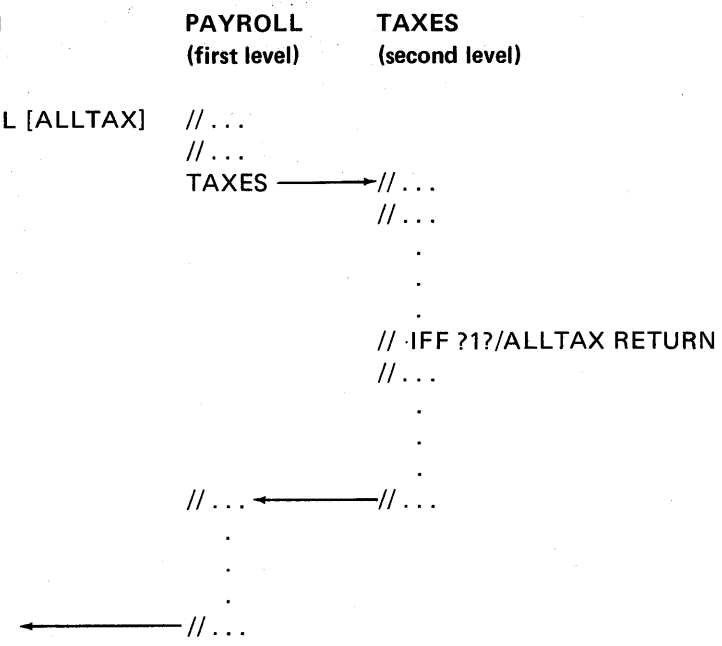
Note: The CANCEL and RETURN keywords are not restricted to use in an IF expression; refer to the *CANCEL* and *RETURN* statements later in this chapter.

The following example shows two levels of procedures: PAYROLL (first level) and TAXES (second level). The TAXES procedure represents a nested procedure that contains the RETURN keyword in a conditional expression.

The keyboard entry is PAYROLL. PAYROLL contains an optional parameter ALLTAX, which is not specified so that the conditional expression is executed and the RETURN keyword is encountered. RETURN encountered in a nested procedure causes a return to the calling procedure (PAYROLL). The remaining OCL statements in the TAXES procedure are not executed. For example:



If the ALLTAX parameter was specified in the keyboard entry, the RETURN keyword in the conditional expression would not be encountered, all OCL statements in the TAXES procedure would be executed, and processing would return to the PAYROLL procedure. For example:



ELSE Expressions

The ELSE expression can be used only in conjunction with the IF expression. The ELSE expression is executed if the IF expression is not satisfied. The following restrictions apply:

- If the ELSE expression does not immediately follow the IF expression, the ELSE expression is ignored.
- Continuation of a statement following an ELSE expression onto two or more lines will invalidate the continuation expression.

For example:

```
// IF ?1?/ RETURN
// ELSE DELETE ?1?
```

The example tests whether the first positional parameter in the procedure command that called the procedure is a *null entry*. A null entry is an entry that contains no value. In the statement:

```
NAME PARM1,,PARM3
```

the second parameter is a null entry.

In the preceding example, the IF and ELSE statements specify; (1) if the first parameter in the procedure command that called the procedure is a null entry, RETURN to the calling procedure if this is a nested procedure or to the keyboard if this procedure is not nested; (2) if the first parameter in the statement is not a null entry, call the DELETE procedure to delete the file specified by the first parameter.

The ELSE expression can be used with all forms of the IF expression (IF, IFT, and IFF).

There can be only one ELSE expression per line and it must be the first expression in that line. An IF expression can follow an ELSE expression in a conditional statement. For example:

```
// IF ?1?/ PROCA
// ELSE DELETE ?1?
// IF ?2?/ PAYROLL DAILY
// ELSE IF ?2?/YEAREND PAYROLL YEAREND
// ELSE PAYROLL WEEKLY
```

In this example, if the first parameter is passed by the procedure command that called the procedure, the file specified by the parameter is deleted by the DELETE procedure. If the first parameter is not specified, the PROCA procedure is run. If the second parameter is a blank, the PAYROLL procedure with the parameter DAILY is called. If the second parameter is YEAREND, YEAREND is passed to PAYROLL and the PAYROLL procedure is run. If the second parameter is neither a blank nor YEAREND, the parameter WEEKLY is passed to PAYROLL and PAYROLL is run.

You can have conditional expressions in which more than one IF expression precedes an ELSE expression, as in the following:

```
// IF SWITCH1-1 IF SWITCH2-1 PROCA
// ELSE PROCB
```

In this case, procedure PROCA will only be executed when both switch 1 and switch 2 are on. If either switch 1 or switch 2 is off, procedure PROCB is executed.

RESET STATEMENT

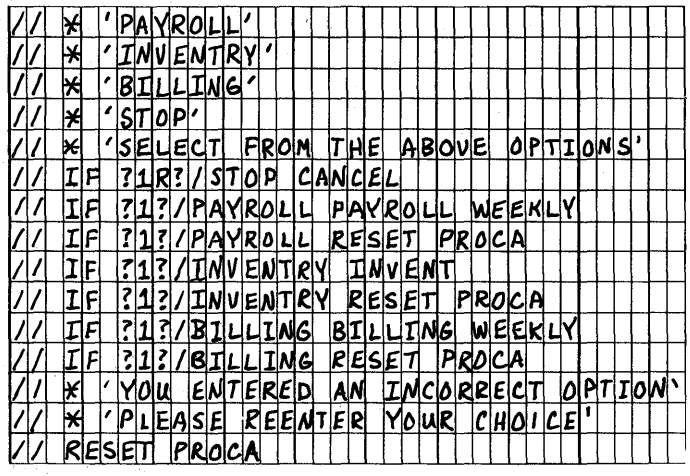
On System/34, 16 levels of nested procedures are allowed. However, for some applications, many more levels of procedure calls are desired. For example, a procedure might call itself an undetermined number of times. The RESET statement calls a procedure and specifies that it should be treated as a first-level procedure; that is, when the procedure called by the RESET statement completes, control is not returned to the procedure level containing the RESET statement.

The RESET statement has the following format:

```
// RESET procname [parameter-1,parameter-2, . . .parameter-n]
```

where procname specifies the name of the procedure being called, and parameter-1 through parameter-n are the optional procedure command parameters. The procedure specified by procname becomes a first-level procedure. Control is not returned to the procedure containing the RESET statement.

RESET Statement Example: A procedure called PROCA displays a list of options to an operator and prompts the operator to enter one of the options. PROCA then calls the procedure specified by the selected option. When that procedure completes, PROCA calls itself, and the process is repeated. In this application, PROCA must be able to call itself any number of times. Following are the statements in the procedure member called PROCA.



CANCEL STATEMENT

The CANCEL statement cancels an entire job. If a CANCEL statement is used to cancel an MRT procedure, an error message is issued to all active requestors of that procedure.

The CANCEL statement has the following format:

```
// CANCEL
```

RETURN STATEMENT

If the RETURN statement is encountered in a first-level procedure, it causes an immediate return to the keyboard for the next control statement. If encountered in a nested procedure, the RETURN statement causes a return to the calling procedure for the next control statement.

The RETURN statement has the following format:

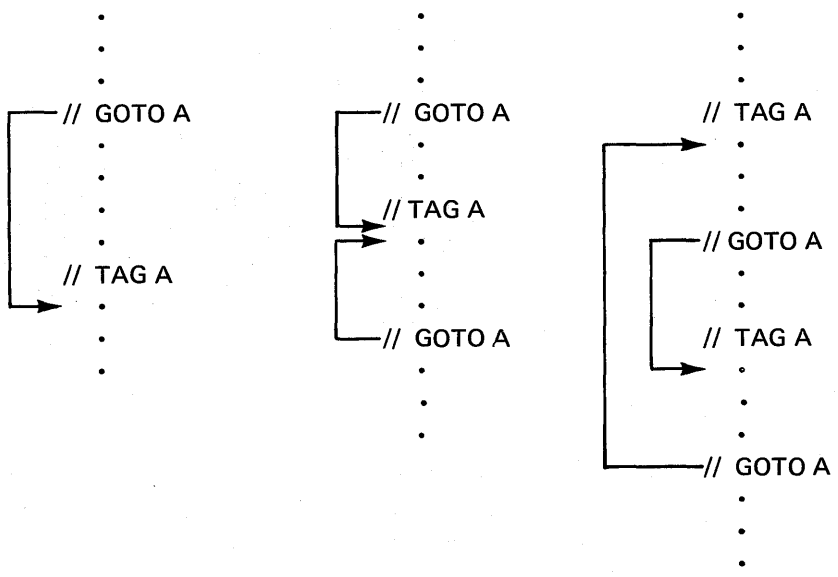
```
// RETURN
```

GOTO AND TAG STATEMENTS

The GOTO and TAG statements allow the programmer to branch around groups of statements within a procedure. GOTO and TAG statements can reduce the number of expressions that the SSP must process during procedure generation. The formats of the statements are:

```
// GOTO label
     and
// TAG label
```

A GOTO statement causes the SSP to branch to the statement following a TAG statement with the same label. (More than one TAG statement in a procedure can have the same label. The system branches to the next TAG statement with the required label.) The SSP searches for the target TAG statement within the procedure. The search begins with the statement following the GOTO statement. If the SSP does not find the target statement before it reaches the end of the procedure, the SSP goes to the first statement in the procedure and resumes the search.



GOTO/TAG Statement Example: PROCB runs four procedures: PROC1, PROC2, PROC3, and PROC4. Normally, to initiate PROCB, the operator enters PROCB. If the operator cancels PROCB at some point, he can later restart the procedure at the canceled step by entering:

```
PROCB RESTART,stepname
```

where stepname is PROC1, PROC2, PROC3, or PROC4. Following are the statements in PROCB:

```
// IF ?1?/ GOTO PROC1
// IFF ?1?/RESTART * 'INVALID PARAMETER'
// IFF ?1?/RESTART CANCEL
// IF ?2?/PROC2 GOTO PROC2
// IF ?2?/PROC3 GOTO PROC3
// IF ?2?/PROC4 GOTO PROC4
// * 'INVALID PARAMETER'
// IFF ?2?/ CANCEL
// TAG PROC1
PROC1
// TAG PROC2
PROC2
// TAG PROC3
PROC3
// TAG PROC4
PROC4
```

SAMPLE PROCEDURE

Following is a description of a user-coded procedure, FILEBKUP, which is not part of the SSP.

The FILEBKUP procedure copies a disk file to a diskette. The command statement format for FILEBKUP is:

FILEBKUP label1,vol-id, [label2 / label1]

Parameter	Meaning
label1	Required parameter that specifies the label of the disk file to be copied.
vol-id	Required parameter that specifies the volume ID of the diskette that will contain the copied file.
label2	Optional parameter that specifies the label of the new diskette file. If label2 is not specified, the label of the diskette file is label1.

If either or both of the required parameters (label1 and vol-id) are specified on the procedure command, the procedure assumes that label2 was intentionally omitted and uses label1 as the label of the diskette file.

If no parameters are specified on the procedure command, the procedure assumes that label2 might be desired and prompts the operator to enter label2.

Figure 5-1 shows the statements coded for FILEBKUP (Note that the statement numbers at the left in this figure are for reference only. They are not part of the statements.):

```
1  // * ' FILEBKUP IS EXECUTING'
2  // IF ?1?/ IF ?2?/ IF ?3?/ * 'ENTER YES TO ALTER DISKETTE FILE LABEL'
3  // IF ?1?/ IF ?2?/ IF ?3?/ * 'PRESS ENTER FOR SAME AS DISK FILE LABEL'
4  // IF ?1?/ IF ?2?/ IF ?3?/ IF ?11R?/YES * 'ENTER DISKETTE FILE LABEL'
5  // IF ?11?/YES IF ?3R?/ * 'PARAMETER 3 OMITTED-PROC CANCELED'
6  // IF ?11?/YES IF ?3?/ CANCEL
7  // IF ?1?/ * 'ENTER THE DISK FILE LABEL'
8  // IF ?1R?/ * 'PARAMETER 1 OMITTED-PROC CANCELED'
9  // IF ?1?/ CANCEL
10 // IF ?2?/ * 'ENTER THE DISKETTE VOLUME ID'
11 // IF ?2R?/ * 'PARAMETER 2 OMITTED-PROC CANCELED'
12 // IF ?2?/ CANCEL
13 // LOAD *COPY
14 // FILE NAME-COPYIN,LABEL-?1?
15 // FILE NAME-COPYO,UNIT-I1,PACK-?2?,RETAIN-999,
16 // IF ?3?/ LABEL-?1?
17 // IFF ?3?/ LABEL-?3?
18 // RUN
19 // COPYFILE OUTPUT-DISK
20 // END
```

Figure 5-1. Contents of the FILEBKUP Procedure

The statements in Figure 5-1 perform the following functions:

Statement Number	Function
1	Statement 1 displays a message informing the operator that FILEBKUP is executing.
2 and 3	If no parameters were passed, statements 2 and 3 prompt the operator to enter YES if the diskette file label is to be different from the disk file label or to press the Enter/Rec Adv key if the labels are to be the same.
4	Statement 4 displays the prompt ENTER MISSING PARAMETER (see the explanation of the ?nR? substitution format in *Substitution Expressions* earlier in this chapter) and waits for the operator to enter YES or a null entry. The entry becomes the value of the 11th positional parameter. If the operator enters YES, statement 4 causes the prompt ENTER DISKETTE FILE LABEL to be displayed.
5 and 6	If the operator entered YES but did not enter a diskette file label, statements 5 and 6 display a diagnostic message and cancel the job.
7	If the first parameter was not entered on the procedure command, statement 7 prompts the operator for the label of the disk file.
8	Statement 8 displays the prompt ENTER MISSING PARAMETER and waits for the operator to enter the label of the disk file. If the operator responds with a null entry, statement 8 causes a diagnostic message to be displayed.
9	If a null entry was specified as the disk file label, statement 9 cancels the job.
10	If the second parameter was not entered on the procedure command, statement 10 prompts the operator for the volume ID.
11	Statement 11 displays the prompt ENTER MISSING PARAMETER and waits for the operator to enter the volume ID. If the operator responds with a null entry, statement 11 causes a diagnostic message to be displayed.
12	If a null entry was specified as the diskette volume ID, statement 12 cancels the job.

Statement Number	Function
13	Statement 13 generates the required LOAD statement.
14	Statement 14 generates the COPYIN FILE statement, substituting the label specified by the first parameter.
15	Statement 15 generates the COPYO FILE statement, substituting the specified volume ID into the PACK parameter. The comma following the RETAIN parameter indicates that the FILE OCL statement is continued. For information about continuing OCL statements, see *General OCL Coding Rules* in Chapter 1.
16	If a label for the diskette file was not specified, statement 16 generates the LABEL parameter, using the label of the disk file.
17	If a label for the diskette file was specified, statement 17 uses that label to generate the LABEL parameter.
18	Statement 18 generates the RUN OCL statement.
19	Statement 19 generates the COPYFILE utility control statement.
20	Statement 20 generates the END utility control statement.

Chapter 6. Programming Considerations

SYSTEM/34 CONCEPTS

In previous versions of this manual, this chapter contained conceptual information about the following topics:

- Display station interaction
  - SRTs
  - MRTs
  - NEPs
- Interrupting a program (Inquiry)
- Checkpoint and restart facilities
- Disk files
  - File identification
  - File groups
  - File creation
  - File organization
  - File processing
  - File sharing
  - Key sorting
  - Multiple logical files
  - Offline multivolume files
- Libraries
- Print spooling

That information, along with conceptual information about other topics, can now be found in the *System/34 Concepts and Design Guide*.

This chapter now contains information about the following topics:

- Specifying region sizes for a job or a job step
- Library members
- Storing library members in disk or diskette files
- Using display screen formats
- Changing system parameters at IPL time
- Ensuring that halts are issued when printer alignment is required

SPECIFYING REGION SIZE

Specifying Region Size for a Job

To ensure that enough storage is available for every job step in a job, you can specify a job region size. When you specify a job region size, the SSP ensures that that much storage is available for each step in the job before initiating the job.

Notes:

1. The SSP considers all user storage not occupied by non-swappable programs as available storage. (IBM-supplied non-swappable programs include portions of MRJE, BSC support for RPG II data communications programs, and non-swappable spool writer.) If no non-swappable programs are in storage, available storage is equal to user storage.
2. Programs that require more storage than that specified on the REGION statement are not allowed to execute unless storage is available when the job step is executed.
3. If a job step releases the requesting display station or if the job step runs an MRT procedure, the job is, in effect, a new job. The SSP does not ensure that enough storage will be available for such job steps.
4. If a main storage error is detected during the job, enough main storage might not exist for a subsequent job step.

You can specify a default region size for all jobs submitted from a display station by using the SET procedure (described in Chapter 2) or by using the $SETCF utility program (described in Chapter 4). For any job submitted from a display station, you can specify a region size or override the default region size by using a REGION OCL statement before the first LOAD OCL statement in the job. (OCL statements are described in Chapter 1.) If neither the REGION OCL statement nor the SET procedure default region size is specified, the SSP checks that available storage is equal to, or greater than, load module size before starting a job step.

Specifying Region Size for a Job Step

Normally, the size of the load member determines the amount of main storage a program requires and no REGION statement is necessary. (Storage is allocated to a program in 2 K increments: a program that is 3 K bytes long requires 4 K bytes of main storage.) However, certain programs (such as the sort and the work station utility portions of the Utilities Program Product) can use additional main storage up to the region size for the job step. You can specify a region size for a job step by using a REGION OCL statement before the LOAD OCL statement for the job step. If you do not use the REGION OCL statement to set the job step region size, the region size for the job step is the same as the region size for the job. If no region size is specified for the job or the job step, the default region size is used.

Note: If you use two REGION OCL statements before the first LOAD OCL statement in a job, the first REGION statement sets the job region size, and the second REGION statement sets the first job step region size.

LIBRARIES

A library is a file on disk that contains library members. A library member is a named collection of records or statements.

Library Members

A library can contain four types of members:

- Link-edited load members, message members, or display screen formats (O members)
- Procedure members (P members)
- Subroutine members (R members)
- Source members (S members)

A library member occupies physically contiguous sectors within the library. Two members cannot share a sector.

Library Member Names

A library member name:

- Can be from 1 through 8 characters long
- Can contain all characters except blanks, commas (,), question marks (?), periods (.), slashes (/), hyphens (-), and single quotes (')
- Must have an alphabetic character (A through Z, #, $, or @) as its first character (however, you should avoid the use of #, $, or @ as the first character because most IBM program names begin with those characters)
- Cannot be ALL, DIR, NEW, or SYSTEM
- Cannot be any of the procedure command names, control command names, or their abbreviations
- Cannot be the same as the name of a member of the same type within the same library

STORING LIBRARY MEMBERS IN DISK OR DISKETTE FILES

Library members can be stored in disk or diskette files in *record mode* or *sector mode*. A record-mode file on diskette can be exchanged with other systems if it is written as a basic data exchange file (BASIC-YES is specified on the COPY utility control statement for the $MAINT utility program). A sector-mode file on diskette can be exchanged with another System/34 or with IBM System/32, subject to the restrictions stated under *Sector-Mode Files* later in this chapter.

Record-Mode Files

Only source and procedure members can be stored as record-mode files. Records in a record-mode file can be from 40 through 120 characters long, but all records in a file must have the same length. The SSP pads records with blanks (hex 40s) or truncates records to the specified record length. The first record in a member stored in a record-mode file on disk or diskette is a statement with the following format:

```
// COPY LIBRARY-{P} ,NAME-member name [,RETAIN-S]
             {S}

   [,MRT-YES] [,PDATA-YES] [,HIST-NO]
```

The COPY statement parameters have the following meanings:

Parameter	Meaning
LIBRARY-P	Specifies that the member is a procedure member.
LIBRARY-S	Specifies that the member is a source member.
NAME-member name	Specifies the name of the library member.
RETAIN-S	Specifies that the member is an SSP member. If RETAIN-S is not specified, the member is not an SSP member.
MRT-YES	Specifies that the procedure member is an MRT (multiple requesting terminal) procedure. If MRT-YES is not specified, the member is not an MRT procedure.

Parameter	Meaning
PDATA-YES	Specifies that data (not parameters) can be passed on the procedure command statement that causes the procedure to be executed. The data starts with the first nonblank character following the procedure name and ends with the last nonblank character in the statement. The data is passed to the first program in the procedure. The data is passed on the first input operation from the requesting display station. Every MRT procedure has this attribute whether or not PDATA-YES is specified. If PDATA-YES is not specified and if the procedure is not an MRT procedure, parameters are passed on the procedure command statement.
HIST-NO	Specifies that OCL statements generated by the procedure when it was executed are not logged in the history file. If HIST-NO is not specified, the OCL statements generated by the procedure are logged in the history file.

The last record in a member stored in a record-mode file is a statement with the following format:

```
// CEND
```

When you use the $MAINT utility program to copy a source or procedure member into a record-mode file, $MAINT places the COPY and CEND statements into the file. If you create a record-mode file to be copied into a library member by the TOLIBR procedure or by the $MAINT utility program, *you* must place the COPY and CEND statements into the file.

If the record-mode file is organized as a direct file, you must insert an END statement following the CEND statement that terminates the last member in the file. The format of the END statement is:

```
// END
```

where only one blank must separate // and END.

Sector-Mode Files

Any library member can be stored as a sector-mode file. All members copied into a sector-mode file have 40 bytes of control information preceding the member. The control record contains 28 bytes for the member's directory entry, 8 bytes of PTF information, and 4 reserved bytes.

If you copy a System/32 sector-mode file to a System/34 library:

- System/32 SCP members are not copied to the System/34 library.
- The SSP sets the release level of the copied members to 232 to indicate that the members came from a System/32.
- Source or procedure members copied from the sector-mode file may require more library space than they did on System/32.
- System/32 files created by versions 1 through 4 of System/32 cannot be read by System/34 until they are updated by the System/32 CONVERT procedure.

Notes:

1. You can copy subroutine members (R members) or load members (O members) from System/32, but you should test them carefully, because you must often modify and recompile subroutines for use on System/34.
2. You cannot copy a System/34 sector-mode file to a System/32 library.

USING DISPLAY SCREEN FORMATS

Programs that you write for System/34 can use the display screen at the display station as an input/output device. A display screen format defines to the SSP an entire display station display. Some advantages of using display screen formats are:

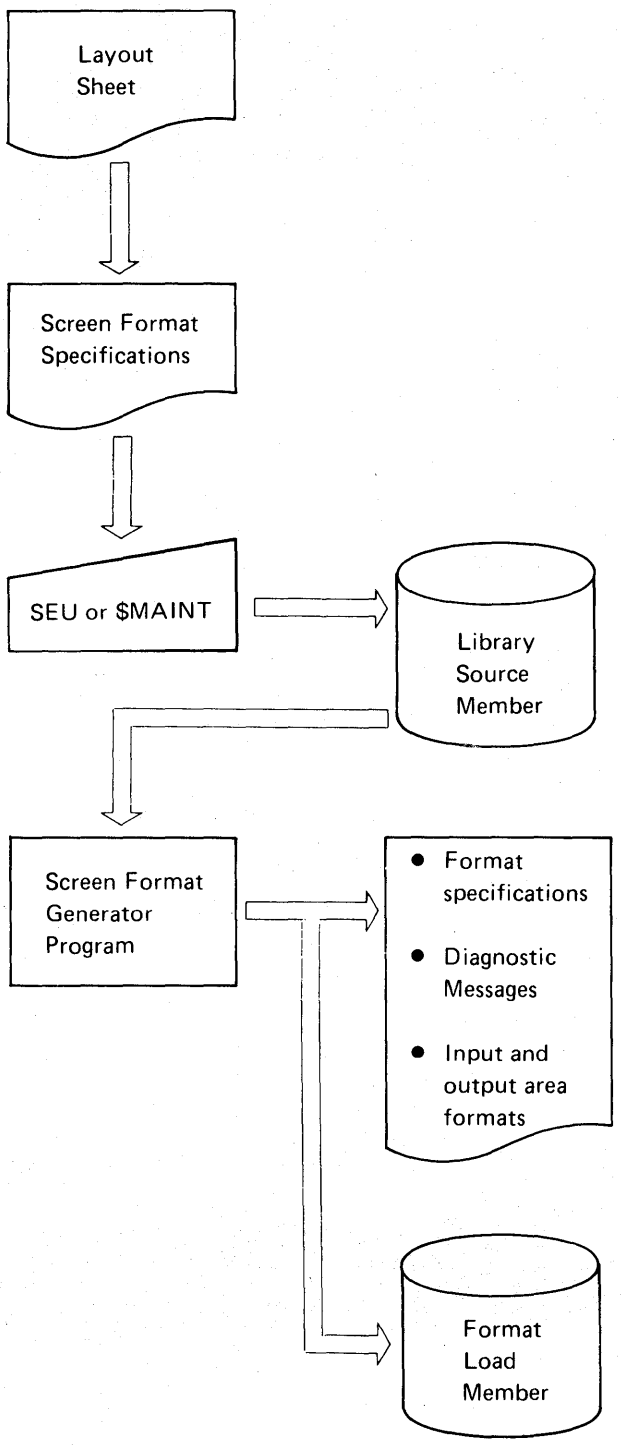
- The formats are program independent. (They are stored in a library; they are not part of the program that uses them.)
- The formats provide an easy way of handling data (all data is arranged in fields).
- The formats provide two ways of specifying information to be displayed: as part of the field definition when the format is generated, or supplied by the program when it displays the format.
- The display station operator can change the contents of a field when it is displayed.

Note: Although display screen formats are stored independent of the using program, all formats used by an RPG II or a WSU program must be stored in a single library member. The member name for an RPG II program must be the program name followed by the characters FM. For example, the formats used by a program called ACCTS must all be in a library member called ACCTSFM.

Steps in Creating a Display Screen Format

Figure 6-1 shows the steps in creating a display screen format if you use the FORMAT procedure or the $SFGR utility program.

Note: You can also use the screen design aid (SDA) portion of the Utilities Program Product to design display screen formats interactively. For information about SDA, see the *Screen Design Aid Programmer's Guide and Reference Manual.*



1. Arrange all fields on the layout sheet, just as they will appear on the display screen.

2. Use the completed layout sheet as a guide for filling out the two parts of the display screen format specifications.

3. Use SEU or $MAINT to transfer the specifications to a library source member.

4. Run the screen format generator utility program ($SFGR) by entering the required control statements or by running the FORMAT procedure. $SFGR produces the following output:
   a. A list of format specifications
   b. Diagnostic messages
   c. Information about the input and output records for the format
   d. The display screen format (or formats) in the specified load member
   e. The size of each format that was created

Figure 6-1. Steps for Creating a Display Screen Format

Step 1. Arrange the Fields on the Layout Sheet

Use the *IBM 5251 Display Station Keyboard Template Assignment Sheet and Display Screen Layout* to lay out all fields exactly as they will appear on the display. When laying out the fields, remember that:

- The first position on the display screen (line 1, position 1) is reserved for a nondisplay screen attribute (a character that does not appear on the display screen); therefore, a field cannot start on line 1, position 1.

- A nondisplay control character (the character does not appear on the display screen) that defines the characteristics of the field precedes each displayed field. Be sure to allow one space before each field for the control character.

- If the fields in this format appear on the display screen along with fields previously displayed, the following rules apply:
  - For any output-only field for which no display attributes or indicators are specified in columns 39 through 49 of the D-specification, at least one space must be left between the field and any previously displayed field that follows.
  - For any other field, at least two spaces must be left between the field and any previously displayed field that follows.
  - If this format does not contain any input fields, at least one space must be left following any input field previously displayed.

- If this format is to be displayed on a 960-character display screen, the following rules apply:
  - Any output-only field for which no display attributes or indicators are specified in columns 30 through 49 of the D-specifications, may end in position 80 of line 12.
  - Any other field must end on or before position 74 of line 12.

- A maximum of 256 fields are allowed in a display. Of those fields, the maximum number of input fields is 127. However, if you do not specify input fields in ascending screen sequence order, if you have operator ID fields (R in column 27), or if you specify modulus 10 or modulus 11 self-check fields, the maximum number of input fields is less. Use the following equation to determine the maximum number of input fields:

$$\text{Maximum number of input fields} = \frac{255 - \text{SEQ} - \text{Number of modulus 10 and modulus 11 fields (T or E in column 30)} - \text{Number of secure operator ID fields (R in column 27)} - \text{1/2 length of the longest operator ID field (R in column 27)} - \text{Number of fields that can contain ideographic characters}}{2}$$

where SEQ is equal to 0 if no input fields are out of order. If one or more fields are out of sequence, SEQ is equal to the number of out of sequence fields plus 1.

For example, if 20 fields are specified out of sequence and 10 modulus 10 and modulus 11 fields are specified, the maximum number of input fields is 112, or:

$$\frac{255 - 21 - 10}{2} = 112 \text{ input fields.}$$

Step 2. Fill Out the Screen Format Specifications

Each field in the record is described by entries on the *Display Screen Format Specifications*. You can specify three types of fields: output, input, and output/input.

An output field contains data that cannot be changed by the operator. The data in the field is supplied as part of the field definition when the format is generated, or is supplied by the program when the format is displayed.

An input field is a field on the display reserved for keyboard entry. Data for the field is entered by the display station operator.

An output/input field contains data that was supplied as part of the field definition when the format was generated or that was supplied by the program displaying the format. The data can be changed by the display station operator using the keyboard.

All three classes of fields can be used in the same display.

In addition to specifying the class of the field in the source specifications, the programmer specifies the type of information that can be contained in the field. The types of data that can be specified are:

- Alphabetic only
- Numeric only
- Alphameric
- Signed numeric (the last position of the field is reserved for a sign)

You can also specify other attributes of the field, such as display intensity, whether the field is a blinking field, and whether data must be entered into the field by the operator.

For a complete description of all the field attributes that can be specified in the display specifications, see the description of the $SFGR utility program in Chapter 4.

Step 3. Enter the Specifications into a Source Library Member

The $MAINT utility program (described in Chapter 4) or the SEU portion of the Utilities Program Product (described in the *Source Entry Utility Reference Manual*) can be used to enter the specifications into a source member.

Step 4. Run the Screen Format Generator Utility Program

The $SFGR utility program processes the source specifications and produces:

- A list of the source specifications
- Diagnostic messages
- Descriptions of the input and execution time output records for the screen format
- A list of the formats created and their size
- The screen format (or formats) in the specified load member

All formats used by a program can be placed in one load member (up to a maximum of 32 formats in one member), or they can be placed in more than one member. The program must open each load member containing formats used by the program.

Note: All formats used by an RPG II or a WSU program must be stored in a single library member. The member's name must be the program name followed by the characters FM. For example, the formats used by a program called ACCTS must all be in a library member called ACCTSFM.

Computing Self-Check Digits

Modulus 10

To compute the modulus 10 self-check digit, do the following:

1. Multiply the units position and every alternate position of the base number by 2.
2. Add the digits in the products to the digits in the base number that were not multiplied.
3. Subtract the sum from the next higher number ending in 0.

The difference is the self-check digit.

For example:

Base number	6 1 2 4 8
Units position and every alternate position	6 2 8
Multiply by 2	12 4 16
Digits not multiplied	1 4
Add	1 + 2 + 1 + 4 + 4 + 1 + 6 = 19
Next higher number ending in 0	20
Subtract	19
Self-check digit	1

To compute the modulus 11 self-check digit, do the following:

1. Assign a weighting factor to each digit position of the base number. These factors are: 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, 2, 3, . . . starting with the units position of the number and progressing toward the high-order digit. For example, the base number 991246351 would be assigned the weighting factors as follows:

Base number	9 9 1 2 4 6 3 5 1
Weighting factor	4 3 2 7 6 5 4 3 2

2. Multiply each digit by its weighting factor.

3. Add the products.

4. Divide this sum by 11.

5. Subtract the remainder from 11.

The difference is the self-check digit.

For example:

Base number	1 3 7 3 9
Weighting factors	6 5 4 3 2
Multiply	6 15 28 9 18
Add	6 + 15 + 28 + 9 + 18 = 76
Divide	76/11 = 6 plus a remainder of 10
Subtract	11 - 10 = 1
Self-check digit	1

Note: If the remainder from step 4 is 0, the self-check digit is 0. If the remainder is 1, the base number has no self-check digit; you must ensure that base numbers with remainders of 1 in step 4 are not used in the fields you define as self-check fields.

Multiple Formats

Each format that is displayed does not have to replace the previous format completely; that is, information from several formats can appear on the screen at one time. This technique is useful if some information on the screen is to remain unchanged while other information is to be replaced. If you use multiple formats, (1) you save the time required to send unnecessary information to the display station, (2) the format specifications for the new (overlay) screen are simpler, and (3) the screen format for the new screen may be smaller. When using multiple formats, be careful not to clear or to replace any information that should remain on the screen. The following paragraphs describe considerations for using multiple formats. For rules concerning the placement of fields on the display, see *Steps in Creating a Display Screen Format* in the earlier part of this chapter.

When multiple formats are displayed before information has been read from the display screen, only input fields from the last format displayed with input fields are read. This means when a format that defines input fields is displayed, all previous input field definitions are lost, even if the new input fields are at different locations on the screen.

To provide a performance improvement when a program reads data from a display, the SSP will read the data from the display as soon as the operator presses the Enter key. This data will be passed to the program when the program asks for it. This action provides an overlap between the swapping of the program into storage and the reading of the input data from the display.

In order for this to be valid, the display screen input fields must be valid whenever the keyboard is enabled for input. Thus, if a format is displayed that:

- Specifies a value of other than 24 or blanks for the number of lines to clear in columns 19 and 20 of the S-specification, *and*
- Overlays or clears a line that contains an input field created by the last format displayed with input fields defined, *and*
- Does not define a new input field(s), *and*
- Does not specify suppress input (namely, enables the keyboard for input)

then a work station error may occur when the operator presses the Enter/Rec Adv key, even though the program might never ask for the data or might display additional formats before asking for the input data from the display.

To avoid this problem and to reduce system overhead when multiple formats are displayed, suppress input should be specified in columns 35 and 36 of the S-specification for *all* but the last display format, even if the previous formats do not define any input fields. In addition, the last format displayed *must* define at least one input field if:

- A value of other than 24 or blanks is specified as the number of lines to clear in columns 19 and 20 of the S-specification, *and*
- The format overlays or clears a line that contains an input field created by the last format displayed with input fields defined, *and*
- Suppress input is not specified (the keyboard is enabled for input)

Notes:

1. For all RPG II, COBOL, BASIC, and FORTRAN programs, every format displayed without suppress input specified causes the keyboard to be enabled for input.
2. If an RPG II or COBOL MRT program transmits multiple displays and does not suppress input on all but the last display and if the operator interrupts the program by means of the Attn key before the last display has been transmitted, *the program will be suspended*. No other requestors will be serviced during the inquiry request. No indication that the program has been suspended is given to the operators at the other display stations.

Read Under Format

A read under format allows one program in a procedure to display a format and the next program in the procedure to read it. The first program displays the format and then goes to end of job or releases the display station. While the second program is initiating, the operator keys information into the display. When the operator presses the Enter/Rec Adv key, the input information from the display is sent to the second program. Following is a general description of the steps in a read under format:

	For an RPG II, COBOL, or BASIC Program	For an Assembler Program
Step 1:	The first program displays the format using a normal output operation with suppress input not specified.	The first program displays the format using a put operation with put-nowait and invite input (PTI) specified.
Step 2:	The first program goes to end of job (if the program is an SRT program) or releases the requesting display station (if the program is an MRT program).	The first program goes to end of job (if the program is an SRT program) or releases the requesting display station (if the program is an MRT program).
Step 3:	The second program is initiated. (Data cannot be passed to the second program from an INCLUDE OCL statement. The SSP ignores any data coded on the INCLUDE statement.)	The second program is initiated. (Data should not be passed to the second program from an INCLUDE OCL statement.)
Step 4:	The second program performs a normal read operation.	The second program performs a get (GET) or an accept (ACI) operation.

INITIAL PROGRAM LOAD (IPL)

This section gives a general description of the system parameters the operator can change when performing an IPL (initial program load). For detailed operating information, see the *Operator's Guide*. During IPL, the operator can override many of the parameters specified during system configuration. For detailed information about system configuration, see the *Program Product Installation and Modification Reference Manual*.

During IPL, the operator can:

- Specify the following file rebuild options:
  - Whether or not the disk VTOC should be rebuilt
  - Whether or not files containing errors should be deleted
  - Whether or not old files should be checked
  - Whether or not file labels of files containing errors should be displayed
  - Whether or not checkpoint status should be removed from checkpoint files
- Override the following general system parameters set during system configuration:
  - The system date format
  - Single-program or multiple-program mode
  - The startup procedure name
- Activate remote work stations
- Automatically vary on remote work stations
- Activate SSP-ICF
- Activate autocall/X.21 task
- Specify whether display station data management is transient, partially transient, or completely resident
- Cancel spooling (remains canceled until the next IPL)
- Delete the spool file
- Cancel the input job queue (remains canceled until the next IPL)
- Delete the input job queue

- Override the following spooling parameters set during system configuration:
  - Whether or not all printers should have spooling
  - The size of the spool writer buffer
  - Whether or not the spool writer(s) should start automatically
  - The size of the spool file
  - The size of the spool file segments
  - The preferred location of the spool file
  - Whether or not the spool file should be reformatted (whether or not existing entries should be deleted)

- Override the following input job queue parameters set during system configuration:
  - The size of the input job queue
  - Whether or not the input job queue should be reformatted (whether or not existing jobs should be deleted)
  - Whether or not jobs from the input job queue should be started when IPL is complete

- Override the following history parameters set during system configuration:
  - Whether or not the history overflow file will be used
  - Whether or not the overflow file should be deleted
  - The size of the overflow file
  - Whether or not the overflow file should be reformatted
  - The preferred location of the overflow file

- Override the following performance parameters set during system configuration
  - The size of the display station buffers
  - The size of the assign/free area
  - The size of the trace buffer

During IPL, the local data area for each command display station is set to blanks.

FORMS ALIGNMENT WHEN CHANGING LINES PER PAGE

No halt is given when lines per page changes on a printer. If programs use the same printer with different values for lines per page, forms alignment cannot be controlled unless the programs specify first page alignment or make use of the FORMSNO parameter. The following describes how the FORMSNO parameter can be assigned to ensure that halts are issued when printer alignment is required:

- A different FORMSNO should be used for each forms or lines per page combination. This will cause a change FORMS message (SYS–1404) to be given whenever the forms or lines per page are changed by a program.

  For example:
  - If a work station is configured with nonstandard lines per page, a nonstandard FORMSNO should also be configured.
  - If nonstandard lines per page is used in a program (by FORMS statement, PRINTER statement, or RPG line counter specification) or in an Assembler $DTFP macroinstruction, a nonstandard FORMSNO should also be specified (by FORMS or PRINTER statement).

- When message SYS–1404 is issued, the 1 option should be taken even if the forms are not changed. This insures that the system will issue a change FORMS message for the next program using the printer with a different forms or lines per page combination.

- Alignment can then be done prior to printing when the change FORMS message is issued.

Appendix A. Records, Blocks, and Sector Conversion

RECORDS TO BLOCKS CONVERSION FOR DISK—RECORDS GIVEN ON FILE STATEMENT

Determining the Number of Blocks in a Sequential or Direct File

To determine the number of blocks in a sequential or direct file:

1. *Multiply:* number of records x record length = number of characters

2. *Divide:* $\frac{\text{number of characters (from step 1)}}{\text{number of characters per block (2560)}}$ = number of blocks (If there is a remainder, round up to the next whole number.)

Determining the Number of Blocks in an Indexed File

First, determine the number of data sectors in an indexed file:

1. *Multiply:* number of records x record length = number of characters

2. *Divide:* $\frac{\text{number of characters (from step 1)}}{\text{number of characters per sector (256)}}$ = number of data sectors (If there is a remainder, round up to the next whole number.)

Next, determine the number of index sectors in an indexed file:

3. *Add:* key field length + 3 = index entry length

4. *Divide:* $\frac{\text{number of characters in a sector (256)}}{\text{index entry length (from step 3)}}$ = number of entries per sector (drop fraction)

5. *Divide:* $\frac{\text{number of records}}{\text{number of entries per sector (from step 4)}}$ = number of index sectors (If there is a remainder, round up to the next whole number.)

Then, determine the total number of blocks required for an indexed file:

6. *Add:* number of data sectors (from step 2) + number of index sectors (from step 5) = minimum total number of sectors

7. *Divide:* $\frac{\text{minimum total number of sectors (from step 6)}}{\text{number of sectors per block (10)}}$ = number of blocks (If there is a remainder, round up to the next whole number.)

DISK SECTOR ADDRESS TO BLOCK ADDRESS CONVERSION

To convert sector address to block address, subtract 1 from the sector address, divide the result by 10, and drop the remainder. Examples:

10511 = sector address
(10511 - 1) ÷ 10 = 1051.0
1051 = block address

10520 = sector address
(10520 - 1) ÷ 10 = 1051.9
1051 = block address

DISK BLOCK ADDRESS TO FIRST SECTOR IN BLOCK CONVERSION

To find the first sector in a block, multiply the block address by 10 and add 1. Example:

1051 = block address
(1051 x 10) + 1 = 10511
10511 = first sector in block 1051

Appendix B. Hexadecimal and Decimal Conversion

The following chart can be used to convert a decimal value to a hexadecimal value or to convert a hexadecimal value to a decimal value. Conversion examples are shown on the following page.

Byte				Byte				Byte			
0123		4567		0123		4567		0123		4567	
Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec
0	0	0	0	0	0	0	0	0	0	0	0
1	1 048 576	1	65 536	1	4096	1	256	1	16	1	1
2	2 097 152	2	131 072	2	8192	2	512	2	32	2	2
3	3 145 728	3	196 608	3	12 288	3	768	3	48	3	3
4	4 194 304	4	262 144	4	16 384	4	1024	4	64	4	4
5	5 242 880	5	327 680	5	20 480	5	1280	5	80	5	5
6	6 291 456	6	393 216	6	24 576	6	1536	6	96	6	6
7	7 340 032	7	458 752	7	28 672	7	1792	7	112	7	7
8	8 388 608	8	524 288	8	32 768	8	2048	8	128	8	8
9	9 437 184	9	589 824	9	36 864	9	2304	9	144	9	9
A	10 485 760	A	655 360	A	40 960	A	2560	A	160	A	10
B	11 534 336	B	720 896	B	45 056	B	2816	B	176	B	11
C	12 582 912	C	786 432	C	49 152	C	3072	C	192	C	12
D	13 631 488	D	851 968	D	53 248	D	3328	D	208	D	13
E	14 680 064	E	917 504	E	57 344	E	3584	E	224	E	14
F	15 728 640	F	983 040	F	61 440	F	3840	F	240	F	15
6		5		4		3		2		1	
Position											

Hexadecimal-to-Decimal Example

To find the decimal value of the hexadecimal value 1FA, you would find that in position 3 of the preceding hexadecimal and decimal chart, hex 1 equals decimal 256; that in position 2 of the chart, hex F equals decimal 240; and that in position 1, hex A equals decimal 10. If you add these decimal values together, you have the decimal value of hex 1FA: 256 + 240 + 10 = 506.

Hex	Dec	Hex	Dec	Hex	Dec
1	256	F	240	A	10
3		2		1	
Position					

Decimal-to-Hexadecimal Example

To find the hexadecimal value of the decimal number 534, you would find that the next lower decimal number in the preceding hexadecimal and decimal chart is 512 in position 3, equal to hex 2. You then subtract 512 from 534 and use the difference (22) to find the next hexadecimal value. The next lower decimal number in the chart is 16 in position 2, equal to hex 1. The difference between 22 and 16 is 6. The remaining 6 is found in position 1 of the chart, equal to hex 6. Therefore, the hexadecimal value of decimal 534 is 216.

Hex	Dec	Hex	Dec	Hex	Dec
2	512	1	16	6	6
3		2		1	
Position					

Appendix C. Diskette Formats and Diskette Data Files

DISKETTE TYPES

Two type of diskettes can be used on System/34:

- Diskette 1 is a one-sided, single-density diskette that has information recorded on only one side. A diskette 1 diskette can contain up to 19 file labels.
- Diskette 2D is a two-sided, double-density diskette that has information recorded on both sides. In addition, information is written at twice the density of the one-sided diskette. That is, twice as much information is written on the same surface area. A diskette 2D diskette can contain up to 71 file labels.

DISKETTE DRIVES

One of three diskette drives is available on System/34:

- A diskette drive with one read/write head that can be used to read from and to write to only a diskette 1 diskette.
- A diskette drive with two read/write heads that can be used to read from and to write to a diskette 1 diskette or a diskette 2D diskette.
- A diskette magazine drive with two read/write heads that can be used to read from and write to diskette 1 or diskette 2D diskettes. The magazine drive can accommodate three individual diskettes, and two magazines, each of which can contain 10 diskettes.

DISKETTE FORMATS

Each type of diskette can be initialized in two formats.

Formats for Diskette 1 Diskette

- 128-bytes-per-sector format: All sectors on the diskette contain 128 bytes.
- 512-bytes-per-sector format: On track 0 (the index track), each sector contains 128 bytes. On all other tracks, each sector contains 512 bytes.

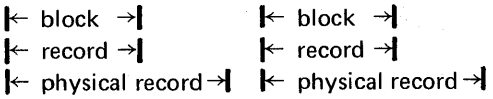
Formats for Diskette 2D Diskette

- 256-bytes-per-sector format: For cylinder 0, track 0, all sectors contain 128 bytes. For cylinder 0, track 1 and all other cylinders, each sector contains 256 bytes.
- 1,024-bytes-per-sector format: For cylinder 0, track 0, all sectors contain 128 bytes. For cylinder 0, track 1, all sectors contain 256 bytes. For all other cylinders, each sector contains 1,024 bytes.

DISKETTE RECORD ATTRIBUTES

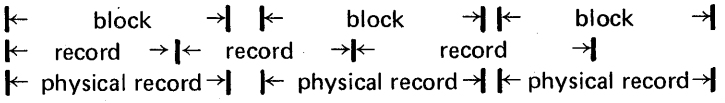
Records Unblocked and Unspanned

Records are unblocked and unspanned when there is only one record per block, regardless of the physical record length.



Records Blocked and Spanned

For optimal use of diskette capacity, fixed-length logical records can be recorded contiguously, independent of physical record length and of how logical records may start or end with respect to physical record boundaries. This type of record attribute is called blocked and spanned.



DISKETTE DATA FILE TYPES

System/34 creates and processes three kinds of diskette data files: basic data exchange files, I exchange files, and System/34 system files.

Basic Data Exchange Files

Basic data exchange files can be written only on one-sided, single-density diskettes initialized in the 128-bytes-per-sector format or on two-sided, double-density diskettes initialized in the 256-bytes-per-sector format. These files can be used for exchanging diskettes between systems and devices. See *The IBM Diskette General Information Manual* for a description of the data set label fields.

Basic data exchange files are created by the TRANSFER procedure, by the $BICR SSP utility program, or by the $MAINT utility program with BASIC-YES specified. The records contained in a basic exchange file are not blocked and cannot span diskette sectors; that is, only one sector (128 bytes for the diskette 1 diskettes, 256 bytes for the diskette 2D diskettes) is used per record. The data is truncated if the record length of the file being transferred is greater than the sector size. The copy of a diskette file created by the COPYI1 procedure or by the $DUPRD system utility program is a basic data exchange file if the original diskette file is a basic data exchange file.

I Exchange Files

I exchange files can reside on either type of diskette that is initialized in either of the valid formats for the diskette. These files can be used for exchanging diskettes between systems and devices that support diskette exchange type I.

I exchange files are created by the TRANSFER procedure or by the $BICR SSP utility program. The copy of a diskette file created by the COPYI1 procedure or the $DUPRD system utility program is an I exchange file if the original diskette file is an I exchange file. The records contained in I exchange files are blocked and can span diskette sectors; that is, several records and parts of records can be placed in a diskette sector, or a record can extend from one sector to another.

System Files

System files can reside on either type of diskette initialized in either of the valid formats for the diskette. System files created by System/32 can be read by System/34; however, System/32 files created with revisions 1 through 4 of System/32 SCP cannot be read by System/34 until they are updated by the System/32 CONVERT procedure. System files created on a diskette 2D diskette cannot be exchanged with other systems.

System files are created by the FROMLIBR, ORGANIZE, and SAVE procedures, and by the $COPY and $MAINT SSP utility programs.

A type of system file called a special E format file is produced by the IBM 5260 Retail System. The POST procedure or the $POST utility program can be used to convert a special E format diskette file to a sequential or indexed file on disk.

Appendix D. SSP Service Procedures and Commands

The SSP service procedures and the SETDUMP control command are designed to help you and IBM service personnel diagnose and correct system problems that might occur.

The following quick-reference chart, Figure D-1, shows the formats of the command statements that run the service procedures and the format of the SETDUMP command. Descriptions of the procedures and command statements follow the chart. The descriptions provide the same kind of information that was provided by the procedure descriptions in Chapter 2. The conventions used for statement formats are described at the front of this manual.

```
APAR vol-id, [object program name] , [source program name]

APPLYPTF { OLD          } , { ALLPTF  } , librname
         { ALL          }   { SS1nn   }
         { PTF log number }  { UT1nn   }
                             { AS1nn   }
                             { RG1nn   }
                             { FO1nn   }
                             { CB1nn   }
                             { IC1nn   }
                             { BA1nn   }
                             { EM1nn   }
                             { IG1nn   }
                             { IS1nn   }

DFA nn

DUMP [ MAIN    ] , [ PRINTER ] , [ F1 ]
     [ CONTROL ]   [ CRT     ]   [ I1 ]
     [ IOC     ]
     [ TRACE   ]
     [ CONFIG  ]
     [ PTF     ]
     [ DISK    ]
     [ VTOC    ]
     [ SPOOL   ]
     [ TWA     ]
     [ JOBQ    ]
     [ XTRACE  ]

ERAP

PATCH [ F1 ]
      [ I1 ]

SEC

SETDUMP [ MENU    ]
        [ ADDRESS ]
        [ DISPLAY ]
        [ RESUME  ]
        [ RESET   ]

STATEST

TRACE
```

Figure D-1. Command Statement Formats for SSP Service Procedures

APAR Procedure

Function

The APAR procedure collects diagnostic information that helps IBM service personnel isolate and correct programming problems that might occur in the system. APAR creates one or more diskette files that contain the following:

- Control storage and main storage dump areas
- I/O controller storage dump area
- System configuration record
- System library PTF log
- Disk trace file (optional)
- Task work area (optional)
- History file (optional)
- Spool file (optional)
- Job queue file (optional)
- Disk VTOC (optional)
- Extended trace files (optional)
- Checkpoint record file (optional)

In addition, you can request that a specified load member be copied to a diskette file called APARLOAD or that a specified source member be copied to a diskette file called APARSRCE. When APAR begins executing, it prompts you for the optional system data areas to be copied to diskette.

Most of the areas copied onto the APAR diskette(s) can be displayed by the DUMP service procedure, or the diskette(s) can be submitted with an APAR (authorized program analysis report).

The APAR procedure runs the $FEAPR utility program and can be run only from the system console.

If the APAR procedure is being run on a system with a main storage size of 256 K bytes, then a diskette format of 128 bytes per sector cannot be used. For additional information about the APAR procedure and for information about the $FEAPR utility program, see the *Data Areas and Diagnostic Aids Manual.*

Command Statement Format

APAR vol-id, [object program name] , [source program name]

Parameters

vol-id: The volume ID of the diskette(s) that will contain output from the APAR procedure.

object program name: The name of the object program causing the program check interrupt. The object program is placed in a file labeled APARLOAD on the specified diskette.

source program name: The name of the source program from which the object program was created. The source program is placed in a file labeled APARSRCE on the specified diskette.

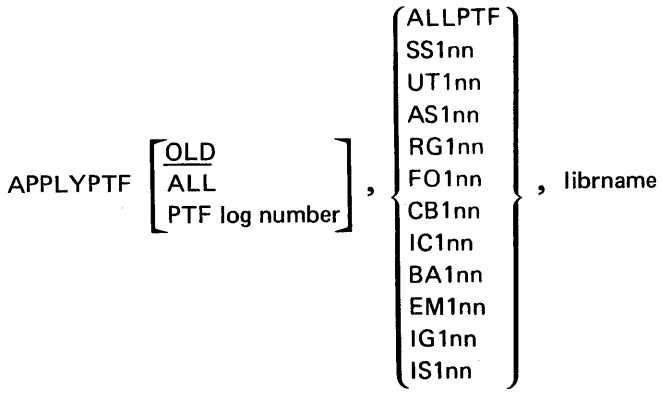
APPLYPTF Procedure

Function

The APPLYPTF procedure applies PTFs (program temporary fixes) to a specified library. Care should be taken to always include applied PTFs in backup copies of the system library or of program products. PTFs applied by the APPLYPTF procedure are read from a PTF diskette. The APPLYPTF procedure cannot be run while:

- Any other jobs are being run
- A user is signed on at any other display station
- Remote work stations are varied online
- SSP-ICF subsystems are enabled
- A communications line is enabled

Command Statement Format



Parameters

OLD: Apply PTFs to existing modules only.

ALL: Apply all PTFs from the selected file.

PTF log number: Apply only the PTF corresponding to the number given. This number is the PTF log number and is indicated in the PTFXREF source member on each PTF diskette.

ALLPTF: Apply all PTFs that apply to the current system configuration. This includes SSP, RPG, utilities, FORTRAN, and so on.

SS1nn: PTFs that change the SSP are applied; nn is the release number of the system.

UT1nn: PTFs that change the IBM System/34 Utilities Program Product (DFU/sort/WSU/SEU/SDA) are applied; nn is the release number of the utility.

AS1nn: PTFs that change the Basic Assembler Program Product are applied; nn is the release number of the basic assembler.

RG1nn: PTFs that change the RPG II Program Product are applied; nn is the release number of RPG II.

FO1nn: PTFs that change the FORTRAN IV Program Product are applied; nn is the release number of FORTRAN IV.

CB1nn: PTFs that change the COBOL Program Product are applied; nn is the release number of COBOL.

IC1nn: PTFs that change the SSP-ICF feature are applied; nn is the release number of the system.

BA1nn: PTFs that change the BASIC Program Product are applied; nn is the release number of the product.

EM1nn: PTFs that change the 3270 Device Emulation Program Product (BSC and SNA) are applied; nn is the release number of the product.

IG1nn: PTFs that change the ideographic utilities are applied; nn is the release number of the product.

IS1nn: PTFs that change the ideographic version of the SSP are applied; nn is the release number of the product.

librname: The specified library where PTFs are to be applied. For RPG II, it is #RPGLIB; for assembler, it is #ASMLIB; for FORTRAN IV, it is #FORTLIB. If ALLPTF is selected, this parameter is not needed. PTFs will be applied to corresponding libraries.

DFA Procedure

Function

The DFA (dump file analysis) procedure retrieves selected information from the dump file, formats the information, and prints or displays it. While DFA is running, the dump file is protected (the dump file cannot be overlaid).

The DFA procedure runs the $FECRE and $FEDSP utility programs and can be run only from the system console.

For additional information about the DFA procedure and for information about the $FECRE and $FEDSP utility programs, see *Appendix A. Diagnostic Aids* in the *Data Areas and Diagnostic Aids Manual*.

Command Statement Format

DFA nn

Parameter

nn: The number of blocks to be allocated for the DFA file. If a value is not specified, a file length of 10 blocks is assumed.

DUMP Procedure

Function

The DUMP procedure prints or displays any of the following areas from disk or from a diskette previously created by the APAR procedure:

- Control storage and main storage dump areas (Information might have been saved in these areas because of a program check interrupt, because option D was selected by the operator, or because the Reset and CE Start keys on the CE panel were pressed.)
- I/O controller storage dump area
- Disk trace file
- System configuration record
- Library PTF log
- Disk VTOC
- Spool file(s)
- Task work area
- Input job queue
- Extended trace file(s)

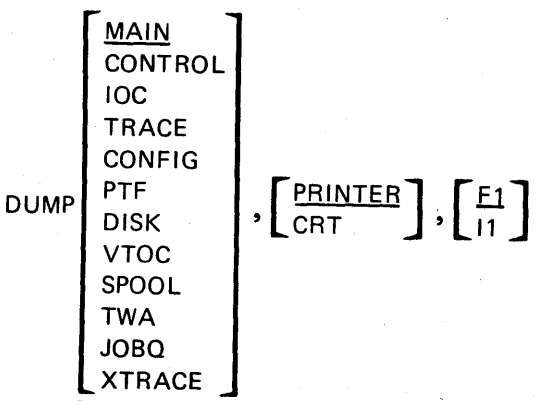
DUMP can also be used to print or display selected sectors from the disk or from a diskette.

The DUMP procedure runs the $FEDMP utility program and can be run only from the system console.

If password security is active, running the DUMP procedure can be restricted to certain operators. For more information about restricting the use of the DUMP procedure, see *Security* in the *Installation and Modifications Reference Manual*.

For additional information about the DUMP procedure and for information about the $FEDMP utility program, see *Appendix A. Diagnostic Aids* in the *Data Areas and Diagnostic Aids Manual*.

Command Statement Format



Parameters

MAIN or *CONTROL:* Displays or prints selected portions of the main storage or control storage dump area.

If the information is displayed, the DUMP procedure displays the following:

- The first segment of main storage (MAIN) or control storage (CONTROL)
- The address of the abnormally terminated task control block
- The last set of saved register values for the terminated task

To display other portions of the dump area, you can specify a different storage address and storage type. The possible storage types are:

- Control storage
- Untranslated main storage
- Translated storage
- Translated extended address mapping (exam) storage

To display the translated storage of another task, change the address of the task control block displayed on the screen.

If the information is printed, the DUMP procedure prompts for the area to be printed. You can select one of the following:

- Control storage dump between limits
- Untranslated main storage dump between limits
- Translated task storage dump between limits
- Translated exam storage dump between limits
- Nucleus storage dump from the start of main storage to the start of the user area
- Formatted control block dump, where frequently used control storage and main storage data areas are printed
- System dump, which includes a formatted control block dump, a nucleus storage dump, and a dump of the translated storage for each task and exam in the system at the time of the abnormal termination
- All existing extended trace tables

IOC: Prints selected portions of the I/O controller storage dump area. The DUMP procedure prompts for the control unit address corresponding to the storage area required. I/O controller storage cannot be displayed on the display screen.

TRACE: Displays or prints the disk trace file, if one exists, starting with the oldest entry.

CONFIG: Displays or prints selected fields from the system configuration record.

PTF: Displays or prints a library PTF log, which identifies all PTFs applied to a library. You will be prompted for the library name.

DISK: Displays or prints selected disk or diskette sectors. You are prompted for the address of the initial sector to be displayed. If the information is printed, you are prompted for the number of sectors to be printed.

VTOC: Displays or prints the disk VTOC.

SPOOL: Displays or prints the spool file.

TWA: Displays or prints the task work area.

JOBQ: Displays or prints the input job queue.

XTRACE: Displays or prints an extended trace file, starting with the oldest entry.

PRINTER: Print output from the DUMP procedure on the printer assigned to the display station. If the second parameter is not specified, PRINTER is assumed.

CRT: Display on the display screen at the display station the output from the DUMP procedure.

F1: Dump information from the disk. If the third parameter is not specified, F1 is assumed.

I1: Dump information from an APAR diskette. If DISK was selected for the first parameter, then information from specified diskette sectors is dumped for any diskette.

ERAP Procedure

Function

The ERAP procedure displays or prints data that was logged for the devices on the system. Depending upon the device, the logged data is contained in one or more of the following tables:

- An *I/O counter table* that contains accumulated statistics reflecting the amount of activity for the device. (For example, the number of verify, write, read or scan read, and nonzero seek operations on a disk drive.)
- An *error counter table* that contains accumulated totals of specific types of errors for the device.
- An *error history table* that contains a series of fixed-length fields, with each field representing an error on the device. The first entry in the table represents the most recent entry. After the table is filled, the oldest entry is dropped from the table each time a new entry is added.

In addition to printing or displaying the logged information, the ERAP procedure allows you to reset the information in the I/O counter tables and the error counter tables.

When the ERAP procedure begins execution, it displays a series of menus from which you can select:

- The logged information to be displayed or printed
- The device, or devices, for which logged information is to be displayed or printed
- The printer where the output is to be printed (if printing is selected)

The ERAP procedure runs the $ERAP utility program.

Note: You can run a limited version of $ERAP by pressing the Test Req key when the Sign On display is active.

For information about the I/O counter tables, error counter tables, and error history tables, see *Section 2. Data Areas* of the *Data Areas and Diagnostic Aids Manual.* For additional information about the ERAP procedure and for information about the $ERAP utility program, see *Appendix A. Diagnostic Aids* in the *Data Areas and Diagnostic Aids Manual.*

Command Statement Format

ERAP

Parameters

None

PATCH Procedure

Function

The PATCH procedure displays selected disk or diskette sectors and allows you to modify the data in those sectors.

PATCH prompts you for the sector number of the sector to be displayed. The first 256 bytes of the selected sector are then displayed along with the sector address. Other portions of the disk or diskette can be displayed by entering a new sector address or by using the display screen function keys to scroll the disk or diskette storage data. You can modify disk or diskette information by replacing data displayed on the display screen with new data.

Because PATCH allows you to alter disk data, PATCH can be run only from the system console. If password security is active, running the PATCH procedure can be restricted to certain operators. For more information about restricting the use of the PATCH procedure, see *Security* in the *Installation and Modifications Reference Manual.*

The PATCH procedure runs the $FEPCH utility program.

For additional information about the PATCH procedure and for information about the $FEPCH utility program, see *Appendix A. Diagnostic Aids* in the *Data Areas and Diagnostic Aids Manual.*

Command Statement Format

PATCH [F1 / I1]

Parameters

F1: A disk sector is to be patched. If a parameter is not specified, F1 is assumed.

I1: A diskette sector is to be patched.

SEC Procedure

Function

The SEC procedure displays, prints, or prints and resets the system event counters. The contents of the 21 system event counters in control storage are written to disk every 6 minutes. It is this copy of the system event counters contents that is displayed or printed by the SEC procedure. The contents of the counters are displayed as decimal values along with the time and date when they were copied from control storage and the date when the counters were last reset.

The SEC procedure runs the $FEEV utility program and can be run only from the system console.

For additional information about the SEC procedure and the system event counters, see the *Diagnostic Aids* section and the *Troubleshooting Aids* section in the *Data Areas and Diagnostic Aids Manual*.

Command Statement Format

SEC

Parameters

None

SETDUMP Control Command

Function

The SETDUMP control command controls the address compare dump function of System/34; this function allows the programmer to request that a dump be taken when an address compare occurs at a specified main storage address. The address compare dump takes a dump without affecting other active jobs in the system.

If password security is active, running the SETDUMP procedure can be restricted to certain operators. For more information about restricting the use of the SETDUMP procedure, see *Security* in the *Installation and Modifications Reference Manual.*

For additional information about the SETDUMP command, see Appendix A, *Diagnostic Aids,* in the *Data Areas and Diagnostic Aids Manual.*

Note: For the ideographic version of the SSP, SETDUMP cannot be used to interrupt the extended character task.

Command Statement Format

```
SETDUMP [MENU    ]
        [ADDRESS ]
        [DISPLAY ]
        [RESUME  ]
        [RESET   ]
```

Parameters

MENU: Displays a menu from which a SETDUMP function can be selected. If a parameter is not specified, MENU is assumed.

ADDRESS: Prompts for the address compare dump values.

DISPLAY: Displays on the display screen the last dump taken.

RESUME: Resumes the job that was suspended for the address compare dump.

RESET: Resets the address compare dump function so that another dump can be taken at a previously entered address.

STATEST Procedure

Function

The STATEST procedure determines whether a communications line and any secondary SDLC stations on the line are operational. This procedure can be executed only from the system console. Refer to *SDLC Station Test* in the *Operator's Guide* for more information.

Command Statement Format

STATEST

Parameters

None

Example

This example shows how to start the SDLC station test procedure:

```
STATEST
```

TRACE Procedure

Function

The TRACE procedure enables you to keep a history of events that occur in the system. Selected system functions are recorded in a variable-length, wrap-around table in main storage as they occur. Optionally, system functions can also be recorded in a separate, variable-length, wrap-around extended trace table. In addition to recording events in main storage, the table also can be written to a system file on disk prior to being reused. As a result, data can be collected over a greater period of time. The TRACE procedure allows setting the events to be traced, initiates extended trace, and initializes a file on disk.

When TRACE begins execution, it prompts you for the events to be traced. You can select tracing of any of the following events:

- Any one or a combination of main storage supervisor calls
- Task dispatches
- Task swaps
- Control storage processor (CSP) supervisor calls
- Main storage processor (MSP) transient scheduler calls and loader requests
- Disk I/O operations
- Extended trace
- Communications line activity

If extended trace is selected, TRACE prompts you for information to create/delete one or more extended trace tables and files.

Extended trace can be selected for one or more communications lines.

After you select the events to be traced, TRACE prompts you for disk logging options.

The TRACE procedure runs the $FETRC utility program. The TRACE procedure can be run only from the system console.

For additional information about the TRACE procedure and for information about the $FETRC utility program and extended trace, see Appendix A, *Diagnostic Aids*, in the *Data Areas and Diagnostic Aids Manual*.

Command Statement Format

TRACE

Parameters

None

Appendix E. SSP Procedure Contents

This appendix shows the resulting OCL statements and utility control statements that are generated and written to the System/34 history file when each of the SSP procedures described in Chapter 2 of this manual is run. The names of parameter values correspond to the names of parameter values on the procedure command formats shown in Chapter 2. This appendix can be used as a guide for programmers who want to know what statements are executed when a procedure is run.

ALTERBSC

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 3337
// LOAD $SETCF
// RUN
```

```
// SETB [BRATE–{F|H}] [,CLOCK–{Y|N}] [,ERC–number]
       [,SLINE–{Y|N}] [,TEST–{Y|N}] [,TONE–{Y|N}]
       [,LNUM–{1|2|3|4}]
// END
```

ALTERSDL

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 3401
// LOAD $SETCF
// RUN
```

```
// SETS [BRATE–{F|H}] [,CLOCK–{Y|N}] [,LNUM–{1|2|3|4}]
       [,SLINE–{Y|N}] [,TEST–{Y|N}] [,TONE–{Y|N}]
// END
```

BACKUP

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2055
// LOAD $BACK

// FILE NAME–#LIBRARY, LABEL–{label / #LIBRARY}, RETAIN–{retention days / 1}
      , PACK–vol-id, UNIT–I1 [, LOCATION–{S1 / S2 / S3 / M1.nn / M2.nn}] [, AUTO–{NO / YES}]
// RUN
// END
```

BLDFILE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 1625
// LOAD $FBLD
// RUN

// FILE LABEL–label, ATTRIB–{S / I / D}, RECL–recl, {BLOCKS–value / RECORDS–value}
      , LOCATION–{A1 / A2 / location}, RETAIN–{S / J / T / P}
      [, POSITION–key position, LENGTH–key length] [, DFILE–{YES / NO}]
// END
```

BLDLIBR

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2545
// LOAD $MAINT
[// FILE NAME–filename,UNIT–{F1 / I1}  [,DATE–{mmddyy / ddmmyy / yymmdd}]
   [,DISP–SHR]  [,LOCATION–{S1 / S2 / S3 / M1.nn / M2.nn}]  [,AUTO–{NO / YES}]
// RUN
// ALLOCATE LIBRSIZE–blocks,LIBRNAME–library name,STATUS–CREATE,
        DIRSIZE–directory sectors,LOCATION–{A1 / A2}
[// COPY FROM–DISK,TO–library name,FILE–file name]
// END
```

BLDMENU

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5774
[// LOAD $MAINT
// RUN

// COPY FROM– {inlib / #LIBRARY}, TO– {outlib / #LIBRARY}, LIBRARY–S,

        NAME–menuname##, RETAIN–R

[// COPY FROM– {inlib / #LIBRARY}, TO– {outlib / #LIBRARY}, LIBRARY–S,

        NAME–textname, RETAIN–R]

// END
// LOAD $MAINT
// RUN

[// DELETE NAME–menuname##, LIBRARY–O, LIBRNAME– {outlib / #LIBRARY}]

[// DELETE NAME–textname, LIBRARY–O, LIBRNAME– {outlib / #LIBRARY}]

// END
```

BLDMENU (continued)

```
// LOAD $MGBLD
// RUN

// MGBLD SOURCE–menuname##, SSP–NO, REPLACE–NO,

        LIBRARY–{outlib
                 #LIBRARY}

// END
[// LOAD $MGBLD
// RUN

// MGBLD SOURCE–textname, SSP–NO, REPLACE–NO, LIBRARY–{outlib
                                                       #LIBRARY}

// END]
// LOAD $BMENU
// RUN

// MENU INPMSG–menuname## [,MENMSG–textname] ,INLIB–{outlib
                                                     #LIBRARY}

      [,REPLACE–YES]

// END
// LOAD $MAINT
// RUN

[// DELETE LIBRARY–O, NAME–textname, LIBRNAME–{outlib
                                               #LIBRARY}]

[// DELETE LIBRARY–S, NAME–menuname##, LIBRNAME–{outlib
                                                 #LIBRARY}]

[// DELETE LIBRARY–S, NAME–textname, LIBRNAME–{outlib
                                               #LIBRARY}]

// END
```

BUILD

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2059
// SWITCH xxxxxxx0
// LOAD $BUILD
// PRINTER NAME–$SYSLIST, SPOOL–NO, LINES–66
// RUN
 [COMPRESS]
```

CATALOG

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2016
// LOAD $LABEL
// RUN

// DISPLAY UNIT–{I1 / F1},LABEL–{label / ALL}

[,LOCATION–{S1 / S2 / S3 / M1.nn / M2.nn}] [,AUTO–{NO / YES}]

[,SORT–{LOCATION / NAME}]

// END
```

COMPRESS

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2026
// LOAD $PACK
// RUN
```

CONDENSE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2544
// LOAD $MAINT
// RUN

// COMPRESS LIBRNAME–{library name / #LIBRARY}

// END
```

COPYI1

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 1653
// LOAD $DUPRD

// FILE NAME–COPYI1 [,DATE–vol-id] ,UNIT–I1

// RUN

// COPYI1 NAME– {label / ALL} ,PACK–vol-id [,DELETE– {YES / NO}]

    [,PRESERVE– {YES / NO}] [,COPIES– {nn / 1}]

    [,LOCATION– {M1 / M1.01 / S1}]

// END
```

COPYPRT

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5048
// LOAD $UASF
// RUN
// SPOOL [SPOOL–ID {ALL / SYSTEM / spool-id / Fxxxx / NOCOPY}] [,NAME–filename]

    [,RELCANS–{RELEASE / CANCEL}]

// END
// LOAD $UASC
// FILE NAME–file label
// RUN
```

CREATE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2033
// LOAD $MGBLD
// RUN

// MGBLD SOURCE–sourcename, REPLACE–{YES / NO}, LIBRARY–{library name / #LIBRARY}

// END
```

CRESTART

```
// LIBRARY NAME–0
// MEMBER USER–##MSG2
// * 6271
// LOAD $RSTRT
// RUN

// CHKPT LABEL–label [,RESTART–{NO / YES}]

// END
```

DATE

```
// DATE–date
```

DEFINEID

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5011
// LOAD $IDSET
// RUN

// DEFINEID MODE–{UPDATE / DISPLAY / DELETE}

// END
```

DEFINEPN

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 6247
// LOAD $PNLM
// RUN
// DEFINEPN INAME–source member name , INLIB–source member library ,
// OUTNAME–load member name ,
// OUTLIB–load library name [,MODE– {DELAY / NODELAY}]
// END
```

DEFINX21

```
// LIBRARY NAME–0
// MEMBER PROGRAM1–#XN#M1
// MEMBER PROGRAM2–#XN#M2
// LOAD $XNLM
// RUN
```

DELETE

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 1629
// LOAD $DELET
// RUN
```

```
// SCRATCH LABEL-label [,DATE-date] ,UNIT-{F1 / I1} ,USERLIBS-{YES / NO}

        [,LOCATION-{S1 / S2 / S3 / M1.nn / M2.nn}]
```

and/or

```
// REMOVE LABEL-label,DATA-{YES / NO} [,DATE-date] ,

        UNIT-{F1 / I1} ,USERLIBS-{YES / NO}
```

```
// END
```

DISABLE

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 6266
// LOAD $IEDS
// RUN
// DISABLE SUBSYS-name [,LOCATION-location name]
// END
```

DISPLAY

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2058
// LOAD $COPY
// FILE NAME–COPYIN, LABEL–label [,DATE–date] ,UNIT–F1
// RUN
// COPYFILE OUTPTX–PRINT
[// SELECT RECORD, FROM–value1 [,TO–value2]]
// END
```

ENABLE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 6265
// LOAD $IENBL
// RUN

// ENABLE SUBSYS–name [,LIBRNAME–{#LIBRARY / library name}]

[,LINE–{1 / 2 / 3 / 4}] [,SHOW–{YES / NO}]

[,LOCATION–location name]

// END
```

FORMAT

To create, update, or add to a format load member:

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5411
// LOAD $SFGR
// RUN
// LOADMBR NAME–load member name [,REPLACE–{NO / YES}]
// INOUT INLIB–{input library name / #LIBRARY},OUTLIB–{output library name / #LIBRARY}
   [,PRINT–{YES / NO / PARTIAL}] [,HALT–{YES / NO}]
{ // CREATE SOURCE–source member name,NUMBER–{1 / number of formats}
{ // UPDATE SOURCE–source member name
{ // ADD SOURCE–source member name,NUMBER–{1 / number of formats}
// END
```

To delete a format from a format load member:

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5411
// LOAD $SFGR
// RUN
// LOADMBR NAME–load member name
// INOUT OUTLIB–{library name / #LIBRARY}
// DELETE FORMAT–format name
// END
```

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2540
// LOAD $MAINT
```

```
// FILE NAME– { { file label 1 / member name } / { file label 2 / name } } ,RETAIN– { { S / J / T / P } / { retention days / 1 } } ,
      [ BLOCKS– { blocks / 8 } / PACK–vol-id ] ,UNIT– { F1 / I1 } [,DISP–NEW]
      [ ,LOCATION– { S1 / S2 / S3 / M1.nn / M2.nn } ] [,AUTO– { NO / YES } ]
```

or, if ADD is specified:

```
// FILE NAME– { { file label 1 / member name } / { file label 2 / name } } [,PACK–vol-id] ,UNIT– { F1 / I1 }
      [,DISP–OLD]
      [ ,LOCATION– { S1 / S2 / S3 / M1.nn / M2.nn } ] [,AUTO– { NO / YES } ]
// RUN
```

```
// COPY FROM– { F1 / library name } ,LIBRARY– { S / P / O / R / ALL } ,NAME– { member name / name.ALL } ,
      FILE– { { file label 1 / member name } / { file label 2 / name } } ,TO–DISK,OMIT–SYSTEM [,ADD–YES]
// END
```

HELP

```
// LOAD $HELP
// RUN
```

HISTCRT

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5039
// LOAD $HSML
// RUN

// DISPLAY [SYSTEM– {NO / YES}]

// END
```

HISTORY

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2076
// LOAD $HIST
// RUN

[// DISPLAY [SYSTEM– {NO / YES / OVERFLOW}] , [USER-userid], [WORKSTN-wsid] ,

        { ALLOCATE , RESET
          DELETE , RESET
          [ACTIVE / ALL / jobname / VIEWED / NOLIST] , [RESET / NORESET] } , {CURRENT / TOTAL} , {EONLY / TEXTONLY / CONTROLS}]

// END
// END
```

INIT

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 1676
// LOAD $INIT
// RUN
// UIN OPTION– {RENAME | DELETE | FORMAT | FORMAT2} [,LOCATION– {S1 | S2 | S3 | M1.nn | M2.nn}]
[// VOL PACK– {vol-id | program date} ,ID– {owner-id | OWNERID}]
// END
```

JOBSTR

```
// LIBRARY NAME–0
// MEMBER USER1–## MSG2
// * 2261
// LOAD $MAINT
// FILE NAME–file label ,UNIT–I1
[,LOCATION– {S1 | S2 | S3 | M1.nn | M2.nn}] [,AUTO– {NO | YES}]
// RUN
// COPY FROM–DISK ,FILE–file label ,TO– {library name | # LIBRARY}
// END
// LIBRARY NAME– {library name | 0}
{// INCLUDE procname
   or if Q or jobq prty is specified:
// JOBQ [jobq prty | 3 ,] [library name | #LIBRARY] , procname}
```

KEYSORT

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2271
// LOAD $DDST
// RUN
// KEYSORT LABEL–file label [,DATE–{yymmdd|ddmmyy|mmddyy}][,RETAIN–{P|T|J}]

// END
```

LIBRLIBR

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2546
// LOAD $MAINT
// RUN

// COPY FROM–from library name, LIBRARY–{S|P|O|R|ALL} ,NAME–{name|name . ALL|ALL} ,
        TO–{to library name|from library name} [,NEWNAME–new name] [,RETAIN–R]

// END
```

LINES

```
// FORMS LINES–{number|66} [,CPI–{10|15}] [,LPI–{4|6|8}]
```

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 2031
```

If LIBRARY is specified:

```
LISTLIBR DIR,LIBRARY
```

or, if LIBRFILE is specified:

```
// LOAD $MAINT
// FILE NAME-label,UNIT-{F1|I1} [,DATE-date]
     [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}] [,AUTO-{NO|YES}]
// RUN
// COPY FROM-DISK,TO-PRINT,FILE-label
```

or, if EXCHANGE or IFORMAT is specified:

```
// LOAD $BICR
// FILE NAME-COPYIN,UNIT-I1,LABEL-label [,DATE-date]
     [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}] [,AUTO-{NO|YES}]
// RUN
// DISPLAY
```

or, if APARFILE is specified:

```
// LOAD $FEDMP
// RUN
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-CONFIG
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-PTF
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-TRACE
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-MAIN
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-CONTROL
// DUMP INPUT-I1,OUTPUT-{CRT|PRINTER},LIST-IOC
```

or, if BACKUP or PROFILE is specified:

```
// LOAD $FEDMP
// RUN
// DUMP INPUT-I1,OUTPUT-PRINTER,LIST-DISK
```

or, if COPYFILE is specified:

```
// LOAD $COPY
// FILE NAME-COPYIN,LABEL-label,UNIT-{F1|I1} [,DATE-date]
     [,LOCATION-{S1|S2|S3|M1.nn|M2.nn}] [,AUTO-{NO|YES}]
// RUN
// COPYFILE OUTPTX-PRINT
```

```
// END
```

LISTLIBR

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2541
// LOAD $MAINT
// RUN
```

```
// COPY FROM– { library name } ,NAME– { DIR           } ,
              { F1           }          { member name  }
                                        { name.ALL     }
                                        { ALL          }

          LIBRARY– { S      } ,TO–PRINT [,OMIT–SYSTEM]
                   { P      }
                   { O      }
                   { R      }
                   { ALL    }
                   { SYSTEM }
```

```
// END
```

LOG

```
// LOG {PRINTER} {,EJECT   }
      {CRT     } {,NOEJECT}
```

ORGANIZE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2110
// LOAD $COPY
// FILE NAME–COPYIN, LABEL–label1 [,DATE–date] ,UNIT–F1

{ // FILE NAME–COPYO, LABEL–label2, RETAIN–{S}, UNIT–F1
                                          {J}
                                          {T}
                                          {P}

    or

  // FILE NAME–COPYO, LABEL–label1, RETAIN– {retention days}, 
                                            {     1       }

        PACK–vol-id, UNIT–I1

        [,LOCATION–{S1}]  [,AUTO–{NO}]
                   {S2}          {YES}
                   {S3}
                   {M1.nn}
                   {M2.nn}
}

// RUN

// COPYFILE OUTPUT–DISK [,DELETE–{'position,character'}],REORG–YES
                                 {SYSDEL              }

// END
```

OVERRIDE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 3400
// LOAD $SETCF
// RUN
// SETR [REMID–xxxxxxxx , LOCID–xxxxxxxx
                or
      LOCID–xxxxxxxx , REMID–xxxxxxxx
                or
      LOCID–xxxxxxxx
                or
      REMID–xxxxxxxx ] [, ADDR– {nn / 00}]

     [, LINE– {P / U / S / T}] [, SWTYP– {AA / MA / MC}] [, WAIT–number]

     [, BLANK– {C / N / T}] [, RCSP– {00 / nn}] [, MLTFL– {Y / N}]

     [, LNUM– {1 / 2 / 3 / 4}]

// END
```

POST

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2183
// LOAD $POST

// FILE NAME–COPYIN, LABEL–label1 [, DATE–date] , UNIT– {F1 / I1}

     [, LOCATION– {S1 / S2 / S3 / M1.nn / M2.nn}] [, AUTO– {NO / YES}]
```

Copy disk to diskette:

```
// FILE NAME–COPYO, LABEL–label1, PACK–vol-id [, RETAIN– {retention days / 1}] ,
        UNIT–I1
   [ , LOCATION– {S1 / S2 / S3 / M1.nn / M2.nn} ]  [, AUTO– {NO / YES}]
```

Copy diskette to disk, with ADD:

```
// FILE NAME–COPYO, LABEL– {label2 / label1}  [, DATE–date]  [, UNIT–F1]
```

Copy diskette to disk, without ADD, size specified:

```
// FILE NAME–COPYO, LABEL–label1 {, RECORDS–number of records / , BLOCKS–number of blocks}
      [, UNIT–F1]
```

Copy diskette to disk, without ADD, using the size of input file:

No COPYO FILE statement is generated.

```
// RUN

// TRANSFER [ADD– {NO / YES}]  [, EOD– {NO / YES}]
            [KEYLEN–key length, KEYLOC–key location]
// END
```

PRESTOR

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 5775
// LOAD $PRST
// FILE NAME-PROFILE, LABEL-label, UNIT-11, PACK-vol-id
// RUN
// END
```

PRLIST

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 5797
// LOAD $PRLT
// RUN
```

PRMENU

```
// LIBRARY NAME-0
// LOAD $PRMN
// RUN
```

PROF

```
// LIBRARY NAME-0
// LOAD $PROF
// RUN
```

PRSAVE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 5778
// LOAD $PRSV
// FILE NAME–PROFILE, LABEL–label, UNIT–I1, RETAIN–999, PACK–vol-id
// RUN
// END
```

PRSRC

```
// LIBRARY NAME–0
// LOAD $PRES
// RUN
```

PRSRCID

```
// LIBRARY NAME–0
// LOAD $PRON
// RUN
```

RELOAD

```
// LOAD $LOADI
```

// FILE NAME–#LIBRARY, LABEL– {label / #LIBRARY} [, DATE–date]

[, PACK–vol-id] , UNIT–I1

[, LOCATION– { S1 / S2 / S3 / M1.nn / M2.nn }] [, AUTO– { NO / YES }]

```
// RUN
```

REMOVE

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 2542
// LOAD $MAINT
// RUN
```

```
// DELETE NAME-{member name / name.ALL / ALL} ,LIBRARY-{S / P / O / R / ALL}
                LIBRNAME-{library name / #LIBRARY}
```

```
// END
```

RENAME

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 1631
// LOAD $RENAM
// RUN
```

```
// RENAME LABEL-label1,NEWLABEL-label2 [,DATE-{mmddyy / ddmmyy / yymmdd}]
```

```
// END
```

REQUESTX

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 2119
// LOAD #GCFR
// RUN
```

RESPONSE

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 6236
// LOAD $ARSP
// RUN
```

```
// RESPONSE SOURCE-source name [,LIBRARY-{library name / #LIBRARY}]
```

```
// END
```

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2530
// LOAD $COPY

// FILE NAME–COPYIN, LABEL– {label   } [,DATE–date] ,UNIT–I1
                            {#SAVE   }

      [                 {S1    }]
      [                 {S2    }]
      [,LOCATION–       {S3    }]  [,AUTO– {NO  }]
      [                 {M1.nn }]          {YES }]
      [                 {M2.nn }]

// FILE NAME–COPYO [,LABEL–label2] {,RECORDS–value1}
                                   {,BLOCKS–value2 }

      [,UNIT–F1] [,LOCATION–value3]

// RUN
{// COPYALL TO–F1                          }
{// COPYFILE OUTPUT–DISK, REORG–NO        }
// END
```

SAVE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2156
// LOAD $COPY
// FILE NAME–COPYIN [,LABEL–label2] [,DATE–date] [,UNIT–F1]
// FILE NAME–COPYO [,RETAIN– {retention days / 1}] [,LABEL– {label / #SAVE}]
      PACK–vol-id,UNIT–I1
      [,LOCATION– {S1 / S2 / S3 / M1.nn / M2.nn}] [,AUTO– {NO / YES}]
// RUN
{ // COPYALL TO–I1 [,GROUP– {ALL / file group}]
      or
  // COPYFILE OUTPUT–DISK,REORG–NO
      or
  // COPYADD }
// END
```

SAVELIBR

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2547
// LOAD $MAINT
// FILE NAME–file label , UNIT–I1 , PACK–vol-id
```

[, RETAIN– { retention days / 1 }] [, LOCATION– { S1 / S2 / S3 / M1.nn / M2.nn }] [, AUTO– { NO / YES }]

```
// RUN
// COPY FROM–library name , TO–DISK , FILE–file label ,
   NAME–ALL , LIBRARY–ALL , OMIT–SYSTEM
// END
```

SET

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2166
// LOAD $SETCF
[// IMAGE MEM, source name]
// RUN
```

// SETCF [LINES–value] [, IMAGE– { YES / NO }] [, FORMAT– { MDY / DMY / YMD }]

[, DATE– { mmddyy / ddmmyy / yymmdd }] [, LIBRARY– { #LIBRARY / name / 0 }]

[, PRINTER– { SYS / ws-id }] [, FORMSNO–nnnn] [, RGSIZE–nn]

```
// END
```

SETFILE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2286
// LOAD $SLFL
// RUN
```

// SETFILE LABEL–file label , DATE– { mmddyy / ddmmyy / yymmdd } , IFILE– { YES / NO }

```
// END
```

SETRETRY

```
// LOAD $PDSR
// RUN
```

SPECIFY

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// *
// LOAD $SETCF
// RUN
// SETP [ADDR-nn] [,LINE-{C|P|S|T}] [,SWTYP-{AA|MC|MA}]
        [,LNUM-{1|2|3|4}] [,ID-nnnnn]
// END
```

STARTM

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 6248
// LOAD $MMST
// RUN
// STARTM [,LINE-{1|2|3|4}] [,CODE-{E|A}] [,STATION-nn]
// END
```

STOPM

```
// LIBRARY NAME-0
// MEMBER USER1-##MSG2
// * 6249
// LOAD $MMSP
// RUN
// STOPM [,LINE-{1|2|3|4}]
// END
```

SYSLIST

```
// SYSLIST [PRINTER] [ , EXTN   ]
           [CRT    ] [ , NOEXTN]
           [OFF    ]
```

TOLIBR

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2543
// LOAD $MAINT

// FILE NAME–file label, UNIT– {F1} [,DISP–SHR]
                              {I1}

      [,LOCATION– {S1   }]  [,AUTO– {NO }]
                  {S2   }           {YES}
                  {S3   }
                  {M1.nn}
                  {M2.nn}

// RUN

// COPY FROM–DISK, TO– {library name} , RETAIN– {P} , FILE–file label
                       {F1          }          {R}

// END
```

TRANSFER

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2183
// LOAD $BICR

// FILE NAME–COPYIN, LABEL–label1 [,DATE–date] ,UNIT– {F1 / I1}

        [ [,LOCATION– {S1 / S2 / S3 / M1.nn / M2.nn}] ]  [,AUTO– {NO / YES}]

{ Transfer disk to diskette:

// FILE NAME–COPYO, LABEL–label1, PACK–vol-id [,RETAIN– {retention days / 1}] ,

        UNIT–I1

        [ [,LOCATION– {S1 / S2 / S3 / M1.nn / M2.nn}] ]  [,AUTO– {NO / YES}]

Transfer diskette to disk, with ADD:

// FILE NAME–COPYO, LABEL– {label2 / label1}  [,DATE–date]  [,UNIT–F1]

Transfer diskette to disk, without ADD; size specified:

// FILE NAME–COPYO, LABEL–label1 {,RECORDS–number of records / ,BLOCKS–number of blocks}

        [,UNIT–F1]

Transfer diskette to disk, without ADD, using size of input file:

No COPYO FILE statement is generated. }

// RUN
```

Diskette basic data exchange or I exchange file to disk sequential file, or disk sequential, indexed, or direct file to diskette basic data exchange file:

[// TRANSFER]

Disk sequential, indexed, or direct file to diskette I exchange file:

// TRANSFER FORMAT–IFORMAT

Diskette basic data exchange or I exchange file to disk sequential file with ADD, or disk sequential, indexed, or direct file to diskette basic data exchange file with ADD:

// TRANSFER ADD–YES

Disk sequential, indexed, or direct file to diskette I exchange file with ADD:

// TRANSFER ADD–YES,FORMAT–IFORMAT

Diskette basic data exchange file to disk indexed file, without ADD:

// TRANSFER ADD–NO,KEYLEN–key length,KEYLOC–key location
// END

XREST

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2276
// LOAD $XREST
```

```
// FILE [LABEL–{label / #EXTN}] [,DATE–date] [,AUTO–{NO / YES}]

     [,LOCATION–{S1 / S2 / S3 / M1.nn / M2.nn}] ,UNIT–I1 ,NAME–#EXTN

// RUN
```

XSAVE

```
// LIBRARY NAME–0
// MEMBER USER1–##MSG2
// * 2281
// LOAD $XSAVE
```

```
// FILE [LABEL–{label / #EXTN}] [,PACK–vol-id]

     [,AUTO–{NO / YES}] [,LOCATION–{S1 / S2 / S3 / M1.nn / M2.nn}]

     ,UNIT–I1 ,NAME–#EXTN [,RETAIN–{retention days / 999}]

// RUN
```

Appendix F. System/34 Characters

This appendix lists the System/34 print belt characters and indicates which print belt(s) the characters are on. If you want information on print belts other than the ones shown, contact your IBM sales representative.

Note: This character set is shown in ascending sorting sequence; that is, the blank character is the lowest—it will be sorted before any other character—and the 9 character is the highest.

		Print Belt				
Character	**Hexadecimal Equivalent**	**48 (BELT48)**	**48HN (BELT48HN)**	**64 (BELT64, BELT.64B, or BELT64C)**	**96 (BELT96)**	**188 (BELT188)**
Blank (not represented on the print belt)	40					•
( )	41					•
â	42					•
ä	43					•
à	44					•
á	45					•
ã	46					•
å	47					•
ç	48					•
ñ	49					•
¢	4A			•	•	[
. (period)	4B	•	•	•	•	•
<	4C			•	•	•
(	4D		•	•	•	•
+	4E	•	•	•	•	•
|	4F			•	•	!
&	50	•	•	•	•	•
é	51					•
ê	52					•
ë	53					•
è	54					•
í	55					•
î	56					•
ï	57					•
ì	58					•
ß	59					•
!	5A			•	•	]
$	5B	•	•	•	•	•
*	5C	•	•	•	•	•
)	5D		•	•	•	•
;	5E			•	•	•

Character	Hexadecimal Equivalent	Print Belt 48 (BELT48)	48HN (BELT48HN)	64 (BELT64, BELT64B, or BELT64C)	96 (BELT96)	188 (BELT188)
¬	5F			•	•	^
- (minus)	60	•	•	•	•	•
/	61	•	•	•	•	•
Â	62					•
Ä	63					•
Ã	64					•
Á	65					•
À	66					•
Å	67					•
Ç	68					•
Ñ	69					•
¦	6A				•	•
, (comma)	6B	•	•	•	•	•
%	6C	•		•	•	•
_ (underscore)	6D			•	•	•
>	6E			•	•	•
?	6F			•	•	•
ø	70					•
É	71					•
Ê	72					•
Ë	73					•
È	74					•
Í	75					•
Î	76					•
Ï	77					•
İ	78					•
` (grave accent)	79			•	•	•
:	7A			•	•	•
#	7B	•		•	•	•
@	7C	•		•	•	•
' (single quote)	7D	•	•	•	•	•
=	7E		•	•	•	•
"	7F			•	•	•
Ø	80					•
a	81				•	•
b	82				•	•
c	83				•	•
d	84				•	•
e	85				•	•
f	86				•	•
g	87				•	•
h	88				•	•
i	89				•	•
«	8A					•
»	8B					•
ð	8C					•
≤	8D					•
Þ	8E					•
±	8F					•
°	90					•
j	91				•	•

Character	Hexadecimal Equivalent	Print Belt 48 (BELT48)	48HN (BELT48HN)	64 (BELT64, BELT64B, or BELT64C)	96 (BELT96)	188 (BELT188)
k	92				•	•
l	93				•	•
m	94				•	•
n	95				•	•
o	96				•	•
p	97				•	•
q	98				•	•
r	99				•	•
a	9A					•
º	9B					•
¤	9C				•	&
≤	9D					•
Æ	9E					•
⊕	9F				•	¤
µ	A0					•
~	A1				•	•
s	A2				•	•
t	A3				•	•
u	A4				•	•
v	A5				•	•
w	A6				•	•
x	A7				•	•
y	A8				•	•
z	A9				•	•
i	AA					•
¿	AB					•
Ð	AC					•
↑	AD					•
§	AE					•
®	AF					•
¢	B0					•
£	B1					•
¥	B2					•
₧	B3					•
ƒ	B4					•
§	B5					•
¶	B6					•
¼	B7					•
½	B8					•
¾	B9					•
¬	BA					•
|	BB					•
≠	BC					•
¨	BD					•
'	BE					•
=	BF					•
{	C0					•
A	C1	•	•	•	•	•
B	C2	•	•	•	•	•
C	C3	•	•	•	•	•
D	C4	•	•	•	•	•

Character	Hexadecimal Equivalent	Print Belt 48 (BELT48)	48HN (BELT48HN)	64 (BELT64, BELT64B, or BELT64C)	96 (BELT96)	188 (BELT188)
E	C5	•	•	•	•	•
F	C6	•	•	•	•	•
G	C7	•	•	•	•	•
H	C8	•	•	•	•	•
I	C9	•	•	•	•	•
–	CA					•
ô	CB					•
ö	CC					•
ò	CD					•
ó	CE					•
õ	CF					•
}	D0				•	•
J	D1	•	•	•	•	•
K	D2	•	•	•	•	•
L	D3	•	•	•	•	•
M	D4	•	•	•	•	•
N	D5	•	•	•	•	•
O	D6	•	•	•	•	•
P	D7	•	•	•	•	•
Q	D8	•	•	•	•	•
R	D9	•	•	•	•	•
≥	DA					•
û	DB					•
ü	DC					•
ù	DD					•
ú	DE					•
ÿ	DF					•
\	E0			•	•	•
S	E2	•	•	•	•	•
T	E3	•	•	•	•	•
U	E4	•	•	•	•	•
V	E5	•	•	•	•	•
W	E6	•	•	•	•	•
X	E7	•	•	•	•	•
Y	E8	•	•	•	•	•
Z	E9	•	•	•	•	•
²	EA					•
Ô	EB					•
Ö	EC					•
Ò	ED					•
Ó	EE					•
Õ	EF					•
0	F0	•	•	•	•	•
1	F1	•	•	•	•	•
2	F2	•	•	•	•	•
3	F3	•	•	•	•	•
4	F4	•	•	•	•	•
5	F5	•	•	•	•	•
6	F6	•	•	•	•	•
7	F7	•	•	•	•	•

Character	Hexadecimal Equivalent	Print Belt 48 (BELT48)	48HN (BELT48HN)	64 (BELT64, BELT64B, or BELT64C)	96 (BELT96)	188 (BELT188)
8	F8	•	•	•	•	•
9	F9	•	•	•	•	•
³	FA					•
Û	FB					•
Ü	FC					•
Ù	FD					•
Ú	FE					•

Appendix G. Polling and Addressing Characters for System/34 BSC Tributary Stations

Polling and addressing characters must be used together in certain pairs; that is, once a polling character is selected, the complementary addressing character is determined; once an addressing character is selected, the complementary polling character is determined.

EBCDIC

The pairs of valid polling and addressing characters for EBCDIC are as follows:

Polling Character	Hexadecimal Representation	Addressing Character	Hexadecimal Representation
BB	C2C2	SS	E2E2
CC	C3C3	TT	E3E3
DD	C4C4	UU	E4E4
EE	C5C5	VV	E5E5
FF	C6C6	WW	E6E6
GG	C7C7	XX	E7E7
HH	C8C8	YY	E8E8
II	C9C9	ZZ	E9E9
JJ	D1D1	11	F1F1
KK	D2D2	22	F2F2
LL	D3D3	33	F3F3
MM	D4D4	44	F4F4
NN	D5D5	55	F5F5
OO	D6D6	66	F6F6
PP	D7D7	77	F7F7
QQ	D8D8	88	F8F8
RR	D9D9	99	F9F9

ASCII

The pairs of valid polling and addressing characters for ASCII are as follows:

Polling Character	Hexadecimal Representation	Addressing Character	Hexadecimal Representation
AA	4141	aa	6161
BB	4242	bb	6262
CC	4343	cc	6363
DD	4444	dd	6464
EE	4545	ee	6565
FF	4646	ff	6666
GG	4747	gg	6767
HH	4848	hh	6868
II	4949	ii	6969
JJ	4A4A	jj	6A6A
KK	4B4B	kk	6B6B
LL	4C4C	ll	6C6C
MM	4D4D	mm	6D6D
NN	4E4E	nn	6E6E
OO	4F4F	oo	6F6F
PP	5050	pp	7070
QQ	5151	qq	7171
RR	5252	rr	7272
SS	5353	ss	7373
TT	5454	tt	7474
UU	5555	uu	7575
VV	5656	vv	7676
WW	5757	ww	7777
XX	5858	xx	7878
YY	5959	yy	7979
ZZ	5A5A	zz	7A7A

To specify the addressing characters in the ADDR-nn parameter for the $SETCF utility program or for the OVERRIDE procedure, give the hexadecimal representation of 1 of the characters. It will be duplicated by the system to provide the 2 addressing characters.

For example, ADDR-F7 is given to specify the EBCDIC addressing characters 77 which correspond to EBCDIC polling characters PP. ADDR-70 is given to specify the ASCII addressing characters pp which correspond to ASCII polling characters PP.

Appendix H. EBCDIC and ASCII Codes

The coded character sets for EBCDIC (extended binary coded decimal interchange code) and ASCII (American National Standard Code for Information Interchange) are shown in the following tables. Use the set that your programming system supports.

EBCDIC

		Main Storage Bit Positions 0, 1, 2, 3															
Main Storage Bit Positions		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
4, 5, 6, 7	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE	DS		SP	&	–						{	}	\	0
0001	1	SOH	DC1	SOS		RSP		/		a	j	~		A	J	NSP	1
0010	2	STX	DC2	FS	SYN					b	k	s		B	K	S	2
0011	3	ETX	DC3	WUS	IR					c	l	t		C	L	T	3
0100	4	SEL	ENP RES	INP BYP	PP					d	m	u		D	M	U	4
0101	5	HT	NL	LF	TRN					e	n	v		E	N	V	5
0110	6		BS	ETB	NBS					f	o	w		F	O	W	6
0111	7	DEL	POC	ESC	EOT					g	p	x		G	P	X	7
1000	8	GE	CAN		SBS					h	q	y		H	Q	Y	8
1001	9	SPS	EM		IT				`	i	r	z		I	R	Z	9
1010	A	RPT	UBS	SM/SW	RFF	¢	!	¦	:					SHY			I
1011	B	VT	CU1	FMT	CU3	.	$	,	=								
1100	C	FF	IFS		DC4	<	*	%	@					⌠		⌐	
1101	D	CR	IGS	ENQ	NAK	(	)	_	'								
1110	E	SO	IRS	ACK		+	;	>	=					⌡			
1111	F	SI	ITB/IUS	BEL	SUB	|	¬	?	"								E0

Duplicate Assignment

ASCII

Main Storage Bit Positions 4, 5, 6, 7	Hex	Main Storage Bit Positions 0, 1, 2, 3															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE	SP	0	@	P	`	p								
0001	1	SOH	DC1	!	1	A	Q	a	q								
0010	2	STX	DC2	"	2	B	R	b	r								
0011	3	ETX	DC3	#	3	C	S	c	s								
0100	4	EOT	DC4	$	4	D	T	d	t								
0101	5	ENQ	NAK	%	5	E	U	e	u								
0110	6	ACK	SYN	&	6	F	V	f	v								
0111	7	BEL	ETB	'	7	G	W	g	w								
1000	8	BS	CAN	(	8	H	X	h	x								
1001	9	HT	EM	)	9	I	Y	i	y								
1010	A	LF	SUB	*	:	J	Z	j	z								
1011	B	VT	ESC	+	;	K	[	k	{								
1100	C	FF	FS	,	<	L	\	l	|							
1101	D	CR	GS	-	=	M	]	m	}								
1110	E	SO	RS	.	>	N	^	n	~								
1111	F	SI	US	/	?	O	_	o	DEL								

Appendix J. Printed Messages

This appendix describes messages that are normally printed and are intended for the use of the programmer. These messages are generated by the SSP utility programs. Also included in this appendix are messages that are written to the history file when an authorized operator uses a protected file or library for which the audit function was selected. Where applicable, the following items are included in the message descriptions:

Item	Explanation
Message Identifier	An 8-character identifier of the form: SYS–xxxx. xxxx is the 4-digit MIC (message identification code) for the message. Every message described in this appendix has a message identifier.
Severity Code	Where applicable, the severity of the message (I for information, W for warning, T for terminal) appears after the message identifier and before the message text. The messages for which severity codes are listed are issued by the $SFGR (screen format generator) utility program. If an informational or a warning message is listed, $SFGR continues processing. However, if a terminal message is printed, $SFGR terminates.
Message Text	The actual printed message.
Specification Type	The $SFGR messages, the type of specification containing the error is listed. (S indicates the display control specification; D indicates the field definition specification.)
Additional Explanantion	Any information that is required *in addition to the message text* is listed in the additional explanation. If no additional information is required, an additional explanation is not listed.

Messages SYS–1705 through SYS–1708 are generated by the $LABEL (VTOC display) utility program.

SYS–1705 $FREE MUST BE RUN BEFORE DATA CAN BE OBTAINED FROM THE ABOVE FILE.

Additional Explanation: A system failure occurred during the running of the $FREE utility program, the $PACK utility program, or the COMPRESS procedure. $FREE, $PACK, or COMPRESS must be run again to ensure data integrity on the disk. No other programs, except $LABEL, should be run until $FREE, $PACK, or COMPRESS runs to completion.

SYS–1706 THE ABOVE FILE CONTAINS AN INVALID DATA AREA.

Additional Explanation: The file contains an unreadable sector. The $BUILD utility program or the BUILD procedure must be run before information in the sector can be read.

SYS–1707 THE ABOVE FILE EITHER HAS AN INVALID INDEX OR KEYSORT IS RUNNING.

Additional Explanation: One of the following conditions exist:

- The file contains an invalid index. The $BUILD utility program or the BUILD procedure must be run before information in the sector can be read.
- A key sort is being performed on the indexed file. For information about sorting keys, see the *Concepts and Design Guide*.

SYS–1708 THE ABOVE AND THE FOLLOWING FILE HAVE OVERLAPPING EXTENTS.

Additional Explanation: The above file and the following file overlap. To correct the condition, you can delete one of the overlapping files and run the COMPRESS procedure.

Messages SYS–3100 through SYS–3135 are overlay linkage editor messages and are described in the *Overlay Linkage Editor Reference Manual.*

Messages SYS–4655 through SYS–4661 are data communications messages and are described in the *Data Communications Reference Manual.*

Messages SYS–5050 through SYS–5232 are generated by the $SFGR utility program (not all message identifiers in this range are used).

SYS–5050 T VALUE SPECIFIED IN FORM TYPE ENTRY IS INVALID. MUST BE S.

Specification Type: None

Additional Explanation: The first noncomment statement in the display screen format specifications is not an S specification (S in column 6).

SYS–5051 T INVALID FIRST CHARACTER IN DISPLAY SCREEN FORMAT NAME ENTRY. MUST BE ALPHABETIC, @, #, or $.

Specification Type: S

Additional Explanation: The format name entry is in columns 7 through 14.

SYS–5052 T FORMAT NAME ENTRY CONTAINS AN EMBEDDED BLANK.

Specification Type: S

Additional Explanation: The format name entry is in columns 7 through 14.

SYS–5053 T FORMAT NAME ENTRY CONTAINS A QUOTE OR COMMA.

Specification Type: S

Additional Explanation: The format name entry is in columns 7 through 14.

SYS–5054 T ADD SPECIFIED, BUT A DUPLICATE DISPLAY SCREEN FORMAT NAME WAS FOUND IN THE FORMAT LOAD MEMBER.

Specification Type: S

Additional Explanation: An attempt is being made to add a display screen format to a format load member, but a format with the name specified in the format name entry (columns 7 through 14) already exists in the format load member.

SYS–5055 T UPDATE SPECIFIED, BUT THE FORMAT LOAD MEMBER DOES NOT CONTAIN A FORMAT WITH THE SPECIFIED NAME.

Specification Type: S

Additional Explanation: An attempt is being made to update a display screen format, but a format with the name specified in the format name entry (columns 7 through 14) does not exist in the format load member.

SYS–5056 T DUPLICATE FORMAT NAMES IN INPUT SOURCE MEMBERS SPECIFIED FOR THIS RUN OF $SFGR.

Specification Type: S

Additional Explanation: The same format name entry (columns 7 through 14) was specified on two different S specifications during this run of $SFGR.

SYS–5057 W FIRST POSITION IN START LINE NUMBER ENTRY IS V. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The start line number entry is in columns 17 and 18.

SYS–5058 W SECOND POSITION IN START LINE NUMBER ENTRY IS INVALID. 01 IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The start line number entry is in columns 17 and 18.

SYS–5059 W VALUE SPECIFIED IN START LINE NUMBER ENTRY IS NOT NUMERIC. 01 IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The start line number entry is in columns 17 and 18.

SYS–5060 W VALUE SPECIFIED IN START LINE NUMBER ENTRY EXCEEDS THE NUMBER OF LINES ON THE SCREEN. 01 IS ASSUMED.

Specification Type: S

Additional Explanation: The start line number entry is in columns 17 and 18.

SYS–5061 W VALUE SPECIFIED IN START LINE NUMBER ENTRY IS ZERO. 01 IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The start line number entry is in columns 17 and 18.

SYS–5062 W SECOND POSITION IN NUMBER OF LINES TO CLEAR ENTRY IS NOT NUMERIC. 24 IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The number of lines to clear entry is in columns 19 and 20.

SYS–5063 W VALUE SPECIFIED IN NUMBER OF LINES TO CLEAR ENTRY IS NOT NUMERIC. 24 IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The number of lines to clear entry is in columns 19 and 20.

SYS–5064 W VALUE SPECIFIED IN NUMBER OF LINES TO CLEAR ENTRY EXCEEDS THE NUMBER OF LINES ON THE SCREEN.

Specification Type: S

Additional Explanation: The value specified in the number of lines to clear entry (columns 19 and 20) is ignored and all lines are cleared.

SYS–5070 W FIRST POSITION IN SOUND ALARM ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The sound alarm entry is in columns 25 and 26.

SYS–5071 W FIRST POSITION IN SOUND ALARM ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The sound alarm entry is in columns 25 and 26.

SYS–5072 W SECOND POSITION IN SOUND ALARM ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED.

Specification Type: S

Additional Explanation: The sound alarm entry is in columns 25 and 26. Column 25 contains a blank and column 26 contains a zero or a nonnumeric character.

SYS–5073 W INDICATOR SPECIFIED IN SOUND ALARM ENTRY IS NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The sound alarm entry is in columns 25 and 26.

SYS–5074 W INDICATOR SPECIFIED IN SOUND ALARM ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The sound alarm entry is in columns 25 and 26.

SYS–5075 W ERASE INPUT IS Y AND OVERRIDE FIELDS IS NOT Y. FIELDS ARE IGNORED WHEN THIS FORMAT IS DISPLAYED.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32 and the override fields entry is in columns 33 and 34. The D specifications for the format are checked for errors, but are ignored when the format is displayed.

SYS–5076 I KEY MASK ENTRY CONTAINS DUPLICATE NUMBERS OR CHARACTERS. DUPLICATE IS IGNORED.

Specification Type: S

Additional Explanation: The key mask entry is in columns 64 through 79.

SYS–5077 I FORMAT CONTAINS FIELDS, PART OF A FIELD, OR AN ATTRIBUTE ON A LINE NOT CLEARED BY THE FORMAT.

Specification Type: S or D

Additional Explanation: If the fields in this format appear on the display screen along with fields previously displayed, the following rules apply:

- For any output-only field for which no display attributes or indicators are specified in columns 39 through 49 of the D-specification, at least one space must be left between the field and any previously displayed field that follows.
- For any other field, at least two spaces must be left between the field and any previously displayed field that follows.

SYS–5078 T A PRECEDING INPUT FIELD HAS A POSITION CURSOR ENTRY OF Y.

Specification Type: D

Additional Explanation: More than one field has Y specified in the position cursor entry (columns 32 and 33). Y can be specified for only one field in a display.

SYS–5080 W FIRST POSITION IN BLINK CURSOR ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The blink cursor entry is in columns 29 and 30.

SYS–5081 W FIRST POSITION IN BLINK CURSOR ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The blink cursor entry is in columns 29 and 30.

SYS–5082 W SECOND POSITION IN BLINK CURSOR ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The blink cursor entry is in columns 29 and 30. Column 29 contains a blank and column 30 contains a zero or a nonnumeric character.

SYS–5083 W INDICATOR SPECIFIED IN BLINK CURSOR ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The blink cursor entry is in columns 29 and 30. Valid indicator values are 01 through 99.

SYS–5084 W INDICATOR SPECIFIED IN BLINK CURSOR ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The blink cursor entry is in columns 29 and 30.

SYS–5085 W CHARACTER SPECIFIED IN LOWERCASE ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The lowercase entry is in column 21 and must be Y, N, or blank.

SYS–5086 W FORM TYPE ENTRY IS BLANK. THIS RECORD IS IGNORED.

Specification Type: S or D

Additional Explanation: The form type entry is in column 6 and must be either S or D.

SYS–5087 T FORM TYPE ENTRY IS INVALID. THIS RECORD IS IGNORED.

Specification Type: S or D

Additional Explanation: The form type entry is in column 6 and must be either S or D.

SYS–5088 T FIELD LENGTH ENTRY IS BLANK OR CONTAINS A CHARACTER THAT IS NOT NUMERIC.

Specification Type: D

Additional Explanation: The field length entry is in columns 15 through 18.

SYS–5089 T VALUE SPECIFIED IN FIELD LENGTH ENTRY EXCEEDS THE NUMBER OF POSITIONS ON THE SCREEN.

Specification Type: D

Additional Explanation: The field length entry is in columns 15 through 18.

SYS–5090 T VALUE SPECIFIED IN FIELD LENGTH ENTRY IS ZERO.

Specification Type: D

Additional Explanation: The field length entry is in columns 15 through 18.

SYS–5091 T SECOND POSITION IN LINE NUMBER ENTRY IS ZERO OR NOT NUMERIC.

Specification Type: D

Additional Explanation: The line number entry is in columns 19 and 20. Column 19 contains a blank and column 20 contains a zero or a nonnumeric character.

SYS–5092 T VALUE SPECIFIED IN LINE NUMBER ENTRY IS NOT NUMERIC.

Specification Type: D

Additional Explanation: The line number entry is in columns 19 and 20.

SYS–5093 T VALUE SPECIFIED IN LINE NUMBER ENTRY IS ZERO.

Specification Type: D

Additional Explanation: The line number entry is in columns 19 and 20.

SYS–5094 T VALUE SPECIFIED IN LINE NUMBER ENTRY EXCEEDS THE NUMBER OF LINES ON THE SCREEN.

Specification Type: D

Additional Explanation: The line number entry is in columns 19 and 20.

SYS–5095 T SECOND POSITION IN HORIZONTAL POSITION ENTRY IS ZERO OR NOT NUMERIC.

Specification Type: D

Additional Explanation: The horizontal position entry is in columns 21 and 22. Column 21 contains a blank and column 22 contains a zero or nonnumeric character.

SYS–5096 T VALUE SPECIFIED IN HORIZONTAL POSITION IS NOT NUMERIC.

Specification Type: D

Additional Explanation: The horizontal position entry is in columns 21 and 22.

SYS–5097 T VALUE SPECIFIED IN HORIZONTAL POSITION ENTRY IS ZERO.

Specification Type: D

Additional Explanation: The horizontal position entry is in columns 21 and 22.

SYS–5098 T VALUE IN HORIZONTAL POSITION ENTRY EXCEEDS THE NUMBER OF HORIZONTAL POSITIONS ON THE SCREEN.

Specification Type: D

Additional Explanation: The horizontal position entry is in columns 21 and 22.

SYS–5099 T LINE NUMBER PLUS START LINE NUMBER EXCEEDS THE NUMBER OF LINES ON THE SCREEN.

Specification Type: S and D

Additional Explanation: The line number entry (columns 19 and 20 on the D specification) plus the start line number entry (columns 17 and 18 on the S specification) exceeds the number of lines on the screen.

SYS-5100 W SCREEN POSITION IS EQUAL TO 0101. IF VARIABLE START LINE NUMBER IS SET TO 1, AN ERROR WILL OCCUR.

Specification Type: S and D

Additional Explanation: On the D specification, the line number entry (columns 19 and 20) and the horizontal position entry (columns 21 and 22) were both 01; and, on the S specification, a variable line number was specified (V in column 17). Because a field cannot begin in position 0101, an error will occur if the variable start line number is 01 when the format is displayed.

SYS-5101 T SCREEN POSITION SPECIFIED IS EQUAL TO 0101. THIS POSITION CANNOT BE USED.

Specification Type: S and D

Additional Explanation: On the D specification, the line number entry (columns 19 and 20) and the horizontal position entry (columns 21 and 22) were both 01; and, on the S specification, the start line number entry (columns 17 and 18) was 01. However, a field cannot begin in position 0101.

SYS-5102 T THIS FIELD OCCUPIES A SCREEN POSITION ALREADY DEFINED BY A PREVIOUS FIELD IN THIS FORMAT.

Specification Type: D

SYS-5103 T MORE THAN 256 FIELDS WERE DEFINED FOR THIS FORMAT.

Specification Type: D

SYS-5104 W FIRST POSITION IN OUTPUT DATA ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The output data entry is in columns 23 and 24.

SYS-5105 W FIRST POSITION IN OUTPUT DATA ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The output data entry is in columns 23 and 24.

SYS-5106 W SECOND POSITION IN OUTPUT DATA ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The output data entry is in columns 23 and 24. Column 23 contains a blank and column 24 contains a zero or a nonnumeric character.

SYS-5107 W INDICATOR SPECIFIED IN OUTPUT DATA ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The output data entry is in columns 23 and 24. Valid indicator values are 01 through 99.

SYS-5108 W INDICATOR SPECIFIED IN OUTPUT DATA ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The output data entry is in columns 23 and 24.

SYS-5109 W CHARACTER SPECIFIED IN INPUT ALLOWED ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The input allowed entry is in column 26 and must be Y, N, or blank.

SYS-5110 W FIRST POSITION IN PROTECT FIELD ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38.

SYS-5111 W FIRST POSITION IN PROTECT FIELD ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38.

SYS-5112 W SECOND POSITION IN PROTECT FIELD ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38. Column 37 contains a blank and column 38 contains a zero or a nonnumeric character.

SYS-5113 W INDICATOR SPECIFIED IN PROTECT FIELD ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38. Valid indicator values are 01 through 99.

SYS-5114 W INDICATOR SPECIFIED IN PROTECT FIELD ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38.

SYS–5115 W FIRST POSITION IN POSITION CURSOR ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33.

SYS–5116 W FIRST POSITION IN POSITION CURSOR ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33.

SYS–5117 W SECOND POSITION IN POSITION CURSOR ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33. Column 32 contains a blank and column 33 contains a zero or a nonnumeric character.

SYS–5118 W INDICATOR SPECIFIED IN POSITION CURSOR ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33. Valid indicator values are 01 through 99.

SYS–5119 W INDICATOR SPECIFIED IN POSITION CURSOR ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33.

SYS–5120 W CHARACTER SPECIFIED IN MANDATORY ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The mandatory entry is in column 29 and must be Y, N, or blank.

SYS–5121 W CHARACTER SPECIFIED IN MANDATORY FILL ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The mandatory fill entry is in column 28 and must be Y, N, or blank.

SYS–5122 W CHARACTER SPECIFIED IN ADJUST/FILL ENTRY IS INVALID. BLANK IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The adjust/fill entry is in column 31 and must be Z, B, or blank.

SYS–5123 W CHARACTER SPECIFIED IN COLUMN SEPARATORS ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The column separators entry is in column 49 and must be Y, N, or blank.

SYS–5124 W CHARACTER SPECIFIED IN CONTROLLED FIELD EXIT ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The controlled field exit entry is in column 35 and must be Y, N, or blank.

SYS–5125 W CHARACTER SPECIFIED IN AUTO RECORD ADVANCE ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The auto record advance entry is in column 36 and must be Y, N, or blank.

SYS–5126 W CHARACTER SPECIFIED IN ENABLE DUP ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The enable dup entry is in column 34 and must be Y, N, or blank.

SYS–5128 W CHARACTER SPECIFIED IN SELF-CHECK ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The self-check entry is in column 30 and must be T, E, or blank.

SYS–5129 W CHARACTER SPECIFIED IN DATA TYPE ENTRY IS INVALID. ALPHAMERIC IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The data type entry is in column 27 and must be A, N, B, S, K, R, or blank. A is assumed.

SYS–5130 W FIRST POSITION IN HIGH INTENSITY ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The high-intensity entry is in columns 39 and 40.

SYS-5131 W FIRST POSITION IN HIGH INTENSITY ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The high-intensity entry is in columns 39 and 40.

SYS-5132 W SECOND POSITION IN HIGH INTENSITY ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The high-intensity entry is in columns 39 and 40. Column 39 contains a blank and column 40 contains a zero or a nonnumeric character.

SYS-5133 W INDICATOR SPECIFIED IN HIGH INTENSITY ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The high-intensity entry is in columns 39 and 40. Valid indicator values are 01 through 99.

SYS-5134 W INDICATOR SPECIFIED IN HIGH INTENSITY ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The high-intensity entry is in columns 39 and 40.

SYS-5135 W FIRST POSITION IN NONDISPLAY ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44.

SYS–5136 W FIRST POSITION IN NONDISPLAY ENTRY IS N. THE SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44.

SYS–5137 W SECOND POSITION IN NONDISPLAY ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44. Column 43 contains a blank and column 44 contains a zero or a nonnumeric character.

SYS–5138 W INDICATOR SPECIFIED IN NONDISPLAY ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44. Valid indicator values are 01 through 99.

SYS–5139 W INDICATOR SPECIFIED IN NONDISPLAY ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44.

SYS–5140 W FIRST POSITION IN BLINK FIELD ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The blink field entry is in columns 41 and 42.

SYS–5141 W FIRST POSITION IN BLINK FIELD ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The blink field entry is in columns 41 and 42.

SYS–5142 W SECOND POSITION IN BLINK FIELD ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The blink field entry is in columns 41 and 42.

SYS–5143 W INDICATOR SPECIFIED IN BLINK FIELD ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The blink field entry is in columns 41 and 42. Valid indicator values are 01 through 99.

SYS–5144 W INDICATOR SPECIFIED IN BLINK FIELD ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The blink field entry is in columns 41 and 42.

SYS–5145 W FIRST POSITION IN REVERSE IMAGE ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The reverse image entry is in columns 45 and 46.

SYS–5146 W FIRST POSITION IN REVERSE IMAGE ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The reverse image entry is in columns 45 and 46.

SYS–5147 W SECOND POSITION IN REVERSE IMAGE ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The reverse image entry is in columns 45 and 46. Column 45 contains a blank and column 46 contains a zero or a nonnumeric character.

SYS–5148 W INDICATOR SPECIFIED IN REVERSE IMAGE ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The reverse image entry is in columns 45 and 46. Valid indicator values are 01 through 99.

SYS–5149 W INDICATOR SPECIFIED IN REVERSE IMAGE ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The reverse image entry is in columns 45 and 46.

SYS–5150 T NEITHER INPUT NOR OUTPUT WAS SPECIFIED FOR THIS FIELD.

Specification Type: D

Additional Explanation: Neither input (Y in column 26) nor output (Y or an indicator value in columns 23 and 24) was specified for the field being defined.

SYS-5151 W CHARACTER SPECIFIED IN CONSTANT TYPE ENTRY IS INVALID. BLANK IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The constant type entry is in column 56 and must be C, M, or blank.

SYS-5152 W CONSTANT TYPE ENTRY IS M. MIC DOES NOT START IN FIRST POSITION OF CONSTANT DATA ENTRY.

Specification Type: D

Additional Explanation: The constant type entry (column 56) is M, indicating that a message identification code (MIC) should begin in the first position of the constant data entry (column 57); however, column 57 is blank. The constant data entry is ignored and a message identification code must be specified by the user program when the format is displayed.

SYS-5153 W CONSTANT TYPE ENTRY IS M. MIC SPECIFIED IN CONSTANT DATA FIELD IS NOT NUMERIC AND IS IGNORED.

Specification Type: D

Additional Explanation: The constant type entry (column 56) is M, indicating that the constant data entry contains a message identification code (MIC) in columns 57 through 60; however, the value in columns 57 through 60 is nonnumeric. The constant data entry is ignored and a message identification code must be specified by the user program when the format is displayed.

SYS-5154 W CONSTANT TYPE ENTRY IS M. MESSAGE MEMBER ID FOLLOWING MIC IS INVALID AND DEFAULT IS ASSUMED.

Specification Type: D

Additional Explanation: The constant type entry (column 56) is M, indicating that the constant data entry contains a message identification code (MIC) in columns 57 through 60 and a message member identifier in columns 61 and 62; however, an invalid message member identifier is in columns 61 and 62. The valid identifiers are U1, U2, P1, P2, M1, and M2. The invalid identifier is ignored and U1 is assumed.

SYS–5155 W FIELD LENGTH IS LESS THAN LENGTH OF CONSTANT DATA ENTRY. CONTINUATION ENTRY IS X BUT IS IGNORED.

Specification Type: D

Additional Explanation: Continuation was specified (X in column 80), but the field length entry (columns 15 through 18) is less than the length of the constant data entry (columns 57 through 79). The X in column 80 is ignored.

SYS–5157 W FIELD SPECIFIED AS OUTPUT-ONLY. DATA TYPE ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: A data type entry (column 27) was specified for a field that was defined as output only. A data type entry is invalid for output-only fields.

SYS–5158 W FIELD SPECIFIED AS OUTPUT-ONLY. POSITION CURSOR ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The position cursor entry is in columns 32 and 33 and must be blank or N for an output-only field.

SYS–5159 W FIELD SPECIFIED AS OUTPUT-ONLY. MANDATORY ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The mandatory entry is in column 29 and must be blank or N for an output-only field.

SYS-5160 W FIELD SPECIFIED AS OUTPUT-ONLY. MANDATORY FILL ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The mandatory fill entry is in column 28 and must be blank or N for an output-only field.

SYS-5161 W FIELD SPECIFIED AS OUTPUT-ONLY. ADJUST/FILL ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The adjust/fill entry is in column 31 and must be blank for an output-only field.

SYS-5162 T CHARACTER SPECIFIED IN SELF-CHECK ENTRY BUT FIELD LENGTH EXCEEDS 32.

Specification Type: D

Additional Explanation: A self-check entry (column 30) cannot be specified for any field with a field length entry (columns 15 through 18) greater than 32.

SYS-5163 W FIELD SPECIFIED AS OUTPUT-ONLY. CONTROLLED FIELD EXIT ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The controlled field exit entry is in column 35 and must be blank or N for an output-only field.

SYS-5164 W FIELD SPECIFIED AS OUTPUT-ONLY. AUTO RECORD ADVANCE ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The auto record advance entry is in column 36 and must be blank or N for an output-only field.

SYS–5165 W FIELD SPECIFIED AS OUTPUT-ONLY. PROTECT FIELD ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The protect field entry is in columns 37 and 38 and must be blank or N for an output-only field.

SYS–5166 W FIELD SPECIFIED AS OUTPUT-ONLY. ENABLE DUP ENTRY IS NOT BLANK OR N AND IS IGNORED.

Specification Type: D

Additional Explanation: The enable dup entry is in column 34 and must be blank or N for an output-only field.

SYS–5167 W FIELD WAS SPECIFIED AS OUTPUT-ONLY. SELF-CHECK ENTRY NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The self-check entry is in column 30 and must be blank for an output-only field.

SYS–5168 W FIELD WAS SPECIFIED AS INPUT-ONLY. CONSTANT TYPE ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The constant type entry is in column 56 and must be blank for an input-only field.

SYS–5169 W FIELD SPECIFIED AS INPUT-ONLY. CONSTANT DATA ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The constant data entry is in columns 57 and 58 and must be blank for an input-only field.

SYS-5170 W FIELD SPECIFIED AS INPUT-ONLY. CONTINUATION ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The continuation entry is in column 80 and must be blank for an input-only field.

SYS-5171 W FIRST POSITION IN UNDERLINE ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48.

SYS-5172 W FIRST POSITION IN UNDERLINE ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48.

SYS-5173 W SECOND POSITION IN UNDERLINE ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48. Column 47 contains a blank and column 48 contains a zero or a nonumeric character.

SYS-5174 W INDICATOR SPECIFIED IN UNDERLINE ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48. Valid indicator values are 01 through 99.

SYS–5175 W INDICATOR SPECIFIED IN UNDERLINE ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48.

SYS–5176 W CONSTANT TYPE ENTRY IS M. CONTINUATION ENTRY IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The continuation entry (column 80) must be blank if the constant type entry (column 56) is M.

SYS–5177 T LAST POSITION IN THIS FIELD IS BEYOND THE LAST POSITION ON THE SCREEN.

Specification Type: S

Additional Explanation: The field being defined extends beyond the end of the screen.

SYS–5178 T THIS RECORD FOLLOWS A CONTINUED RECORD BUT THE FORM TYPE ENTRY IS INVALID.

Specification Type: D

Additional Explanation: The form type entry (column 6) must be D for a record that follows a continued record (a record with X in column 80).

SYS–5179 T CONTINUATION ENTRY IS X BUT THIS IS THE LAST RECORD FOR THE FORMAT.

Specification Type: D

Additional Explanation: The continuation entry is in column 80.

SYS-5180 T CONSTANT DATA ENTRY IS AN SSP MESSAGE MEMBER BUT SSP-YES NOT GIVEN IN THE LOADMBR STATEMENT.

Specification Type: D

Additional Explanation: The message member identifier in columns 61 and 62 specifies an SSP message member (M1 or M2); however, M1 and M2 are valid only if SSP-YES is specified on the LOADMBR utility control statement.

SYS-5181 T CONSTANT TYPE ENTRY IS M BUT FIELD LENGTH ENTRY IS LESS THAN 6.

Specification Type: D

Additional Explanation: The field length entry is in columns 15 through 18, and the constant type entry is in column 56.

SYS-5182 T DATA TYPE ENTRY IS S BUT FIELD LENGTH IS LESS THAN 2 OR GREATER THAN 16.

Specification Type: D

Additional Explanation: The field length entry is in columns 15 through 18, and the data type entry is in column 27.

SYS-5183 T MORE THAN 32 FORMATS HAVE BEEN SPECIFIED FOR THIS LOAD MEMBER.

Specification Type: D

SYS-5184 W MANDATORY FILL ENTRY IS Y AND ADJUST/FILL ENTRY IS Z OR B. ADJUST/FILL ENTRY IS IGNORED.

Specification Type: D

Additional Explanation: Mandatory fill (column 28) and adjust/fill cannot both be specified for a field.

SYS–5185 T NUMBER OF LINES TO CLEAR PLUS START LINE NUMBER EXCEEDS THE NUMBER OF LINES ON THE SCREEN.

Specification Type: S

Additional Explanation: The number of lines to clear entry is in columns 19 and 20, and the start line number entry is in columns 17 and 18.

SYS–5186 T NUMBER OF INPUT FIELDS IN THIS FORMAT EXCEEDS MAXIMUM ALLOWED.

Specification Type: D

Additional Explanation: You can use the following equation to determine the maximum number of input fields allowed:

$$\text{Maximum number of input fields} = \frac{254 - \text{Number of fields specified out of sequence} - \text{Number of modulus 10 and modulus 11 fields (T or E in column 30)} - \text{Number of secure operator ID fields} - \text{1/2 length of the longest operator ID field}}{2}$$

SYS–5187 W CHARACTER SPECIFIED IN RETURN INPUT ENTRY IS INVALID. YES IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The return input entry is in column 22 and must be Y, N, or blank.

SYS–5188 W NONDISPLAY ENTRY AND BLINK FIELD ENTRY ARE BOTH Y. BLINK FIELD ENTRY IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44, and the blink field entry is in columns 41 and 42.

SYS–5189 W NONDISPLAY ENTRY AND UNDERLINE ENTRY ARE BOTH Y. UNDERLINE ENTRY IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44, and the underline entry is in columns 47 and 48.

SYS–5190 W NONDISPLAY ENTRY AND HIGH INTENSITY ENTRY ARE BOTH Y. HIGH INTENSITY ENTRY IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44, and the high-intensity entry is in columns 39 and 40.

SYS–5191 W NONDISPLAY ENTRY AND REVERSE IMAGE ENTRY ARE BOTH Y. REVERSE IMAGE ENTRY IS IGNORED.

Specification Type: D

Additional Explanation: The nondisplay entry is in columns 43 and 44, and the reverse image entry is in columns 45 and 46.

SYS–5192 T UNDERLINE, REVERSE IMAGE, AND HIGH INTENSITY ENTRIES ARE Y. ONLY TWO MAY BE GIVEN FOR ANY FIELD.

Specification Type: D

Additional Explanation: The underline entry is in columns 47 and 48, the reverse image entry is in columns 45 and 46, and the high-intensity entry is in columns 39 and 40.

SYS–5193 T NO VALID SOURCE RECORDS WERE CONTAINED IN THIS SOURCE MEMBER.

Specification Type: Not applicable.

SYS–5194 W CONSTANT TYPE ENTRY IS C. OUTPUT DATA ENTRY IS AN INDICATOR BUT YES IS ASSUMED.

Specification Type: D

Additional Explanation: A constant type entry (column 56) of C is invalid if an indicator is specified in the output data entry (columns 23 and 24).

SYS–5195 W FIRST POSITION IN ERASE INPUT FIELDS ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32.

SYS–5196 W FIRST POSITION IN ERASE INPUT FIELDS ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32.

SYS–5197 W SECOND POSITION IN ERASE INPUT FIELDS ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32. Column 31 contains a blank and column 32 contains a zero or a nonnumeric character.

SYS–5198 W INDICATOR SPECIFIED IN ERASE INPUT FIELDS ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32. Valid indicator values are 01 through 99.

SYS–5199 W INDICATOR SPECIFIED IN ERASE INPUT FIELDS ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The erase input fields entry is in columns 31 and 32.

SYS–5200 W FIRST POSITION IN OVERRIDE FIELDS ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The override fields entry is in columns 33 and 34.

SYS–5201 W FIRST POSITION IN OVERRIDE FIELDS ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The override fields entry is in columns 33 and 34.

SYS–5202 W SECOND POSITION IN OVERRIDE FIELDS ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The override fields entry is in columns 33 and 34. Column 33 contains a blank and column 34 contains a zero or a nonnumeric character.

SYS–5203 W INDICATOR SPECIFIED IN OVERRIDE FIELDS ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The override fields entry is in columns 33 and 34. Valid indicator values are from 01 through 99.

SYS–5204 W INDICATOR SPECIFIED IN OVERRIDE FIELDS ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The override fields entry is in columns 33 and 34.

SYS–5205 W FIRST POSITION IN SUPPRESS INPUT ENTRY IS Y. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The suppress input entry is in columns 35 and 36.

SYS–5206 W FIRST POSITION IN SUPPRESS INPUT ENTRY IS N. SECOND POSITION IS NOT BLANK AND IS IGNORED.

Specification Type: S

Additional Explanation: The suppress input entry is in columns 35 and 36.

SYS-5207 W SECOND POSITION IN SUPPRESS INPUT ENTRY IS ZERO OR NOT NUMERIC. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The suppress input entry is in columns 35 and 36. Column 35 contains a blank and column 36 contains a zero or a nonnumeric character.

SYS-5208 W INDICATOR SPECIFIED IN SUPPRESS INPUT ENTRY IS INVALID. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The suppress input entry is in columns 35 and 36. Valid indicator values are 01 through 99.

SYS-5209 W INDICATOR SPECIFIED IN SUPPRESS INPUT ENTRY IS 00. NO IS ASSUMED FOR THIS ENTRY.

Specification Type: S

Additional Explanation: The suppress input entry is in columns 35 and 36.

SYS-5210 W ENABLE FUNCTION KEYS ENTRY IS NOT Y, N, R OR BLANK. ENTRY IS IGNORED.

Specification Type: S

Additional Explanation: The enable function keys entry is in columns 27.

SYS-5211 W ENABLE COMMAND KEYS ENTRY IS NOT Y, N, R OR BLANK. ENTRY IS IGNORED.

Specification Type: S

Additional Explanation: The enable command key entry is in column 28.

SYS–5212 W KEY MASK ENTRY CONTAINS A FUNCTION KEY NUMBER, BUT THE ENABLE FUNCTION KEYS ENTRY IS NOT Y OR N.

Specification Type: S

Additional Explanation: The key mask entry is in columns 64 through 79; the enable function keys entry is in column 27.

SYS–5213 S KEY MASK ENTRY CONTAINS A COMMAND KEY CHARACTER, BUT THE ENABLE COMMAND KEYS ENTRY IS NOT Y OR N.

Specification Type: S

Additional Explanation: The key mask entry is in columns 64 through 79; the enable command keys entry is in column 28.

SYS–5214 W KEY MASK ENTRY CONTAINS AN INVALID NUMBER OR CHARACTER. ENTRY IS IGNORED.

Specification Type: S

Additional Explanation: The key mask entry is in columns 64 through 79.

SYS–5215 W KEY MASK ENTRY CONTAINS AN EMBEDDED BLANK. NUMBER OR CHARACTER FOLLOWING BLANK IS IGNORED.

Specification Type: S

Additional Explanation: The key mask entry is in columns 64 through 79.

SYS–5216 T DATA TYPE ENTRY IS R, BUT FIELD LENGTH ENTRY IS GREATER THAN 128.

Specification Type: D

Additional Explanation: The data type entry is in column 27, and the field length entry is in columns 15 through 18. The R data type is used for data read from the magnetic stripe reader and cannot have a field length greater than 128.

SYS–5217 W DATA TYPE ENTRY IS R, BUT A FIELD ATTRIBUTE OTHER THAN NONDISPLAY WAS SPECIFIED. NONDISPLAY ASSUMED.

Specification Type: D

Additional Explanation: The data type entry is in column 27. The R data type is used for data read from the magnetic stripe reader and must be a nondisplay field.

SYS–5218 W THIS FORMAT CONTAINS IDEOGRAPHIC DATA OR FIELD(S) ON A LINE GREATER THAN 12.

Specification Type: D

Additional Explanation: Ideographic fields or data have been defined on a format that can only be displayed on a 24-line screen; however, no 24-line screens exist that are capable of displaying ideographic data.

SYS–5219 T THIS FIELD IS DEFINED AS IDEOGRAPHIC CODE CAPABLE BUT HAS LENGTH LESS THAN 4.

Specification Type: D

Additional Explanation: The input data type of the field specifies ideographic character input, but the field is not long enough to contain an ideographic character.

SYS–5220 T THIS FIELD IS IDEOGRAPHIC CODE CAPABLE BUT HAS AN ODD FIELD LENGTH.

Specification Type: D

Additional Explanation: The data type of the field specifies ideographic character input, which requires an even number of bytes, but the field length is odd.

SYS–5221 T FIELDS WITH SPECIFIED DATA TYPE MUST BE CONTAINED ON ONE LINE.

Specification Type: D

SYS–5222 T THIS FIELD COULD CAUSE AN IDEOGRAPHIC CHARACTER TO BE KEYED ACROSS 2 LINES.

Specification Type: D

Additional Explanation: The field is an ideographic field extending to a second display line and beginning on an odd column, thus causing a character to be split between column 80 of one line and column 1 of the next line.

SYS–5223 W THIS FIELD IS IDEOGRAPHIC CODE CAPABLE AND HAS SELF-CHECK. SELF-CHECK IS IGNORED.

Specification Type: D

Additional Explanation: The field is an ideographic data field and has modulus 10/11 self-check specified. The self-check specification is ignored.

SYS–5224 W DUP MAY NOT BE ALLOWED FOR THE SPECIFIED DATA TYPE.

Specification Type: D

Additional Explanation: The DUP function will be allowed only when the field is displayed at a nonideographic-code-capable work station.

SYS–5225 W ADJUST/FILL MAY NOT BE ALLOWED FOR THE SPECIFIED DATA TYPE.

Specification Type: D

Additional Explanation: The adjust/fill function will be performed only when the field is displayed at a nonideographic-code-capable work station.

SYS–5226 W THIS FIELD IS IDEOGRAPHIC CODE CAPABLE AND ZERO FILL IS SPECIFIED.

Specification Type: D

Additional Explanation: The field is capable of ideographic input and has zero or blank fill specified. The field will be blank filled when in ideographic input mode.

SYS–5227 T THIS LINE CONTAINS AN IDEOGRAPHIC CHARACTER THAT WOULD SPAN TWO DISPLAY LINES.

Specification Type: D

Additional Explanation: An ideographic character will not display properly when split between one display line and the next.

SYS–5228 T THE CONSTANT-MIC FIELD OF THIS SPECIFICATION ENDS IN IDEOGRAPHIC MODE.

Specification Type: D

Additional Explanation: A shift in character is missing at the end of the CONSTANT-MIC field of the D specification.

SYS–5229 W THIS LINE CONTAINS AN EXTENDED IDEOGRAPHIC CHARACTER THAT WILL DISPLAY AS THE DEFAULT CHARACTER.

Specification Type: D

Additional Explanation: An extended ideographic character occurs in a field that does not begin with a *shift out* character; therefore, a special character will appear in place of this ideographic character when it is displayed.

SYS–5230 T THIS FIELD CONTAINS AN IDEOGRAPHIC STRING WITH AN ODD LENGTH.

Specification Type: D

Additional Explanation: There is an ideographic string in the CONSTANT/MIC field of the D specification, but there is an odd number of characters between the SO (shift out) and SI (shift in) delimiting the string.

SYS–5231 LOAD MEMBER NOT GENERATED.

Additional Explanation: The screen format load member could not be generated because terminal errors were encountered.

SYS–5232 W DATA TYPE SPECIFIED IS INVALID ON NON-IDEOGRAPHIC SYSTEMS.

Specification Type: D

Messages SYS–5975 through SYS–5978 are messages that are written to the history file when an unauthorized operator uses a protected file or library for which the audit function was selected.

SYS–5975 library name–MEMBERS FROM THIS LIBRARY WERE RUN.

Additional Explanation: An authorized user executed a library member from the named library. This message is written to the history file if the audit option was selected when the library was protected.

SYS–5976 {library name / file label} –USER READ FROM THIS FILE OR LIBRARY.

Additional Explanation: An authorized user read from the named file or library. This message is written to the history file if the audit option was selected for the secured file or library.

SYS–5977 {library name / file label} –USER CHANGED THIS FILE OR LIBRARY.

Additional Explanation: An authorized user changed the contents of the named file or library. This message is written to the history file if the audit option was selected for the secured file or library.

SYS–5978 old name → new name–FILE OR LIBRARY RENAMED.

Additional Explanation: An authorized user changed the name of a secured file or library. This message is written to the history file if the audit option was selected for either the old name or the new name.

Appendix K. Summary of Display Screen Format Specifications

This appendix summarizes the entries on the display screen format specifications. For a more detailed explanation of the entries, see the description of the $SFGR utility program in Chapter 4.

S Specifications

Columns	Name	Entry	Explanation
1-5	Sequence number	Line number	Entry used to number the specification lines.
6	Form type	S	Identification for an S specification.
7		*	Asterisk in this column identifies this line as a comment line.
7-14	Format name	Display screen format name	Name of the display screen format that the $SFGR utility program creates from the S and D specifications.
15-16		Blank	
17-18	Start line number	01-24	Number of the line at which the display begins.
		V (column 17)	Start line number is determined by the user program.
19-20	Number of lines to clear	01-24	Number of lines to clear, including and following the starting line. The specified number of lines are cleared, beginning with the start line specified in columns 17 and 18.
		Blank	All lines are cleared.
21	Lowercase	Y	With the Shift key, operators key in uppercase characters. Without the Shift key, operators key in lowercase characters.
		N or blank	Operators key in uppercase characters only.
22	Return input	Y or blank	Input fields on this display are returned to the user program, even if the operator enters no data.
		N	Input fields on this display are not returned to the user program unless the operator enters data in one or more of the fields. Then, all input fields are returned to the program.
23-24		Blank	

S Specifications (continued)

Columns	Name	Entry	Explanation
25-26	Sound alarm	Y (column 25)	The alarm sounds when this display appears.
		N (column 25) or blank	The alarm does not sound when this display appears.
		01-99	The alarm sounds when this display appears only if the specified indicator is on.
27	Enable function keys	Y	The function control keys identified by numbers in the key mask entry (columns 64 through 79) are enabled (allowed). If the key mask entry contains no numbers, all function control keys are disabled.
		N	The function control keys identified by numbers in the key mask entry are disabled (not allowed). If the key mask entry contains no numbers, all function control keys are enabled. If the operator presses a disabled function control key, an error message is displayed. The operator can then press the Error Reset key and then the correct function key.
		R	The function control key mask that is active for the display station is retained when this format is displayed.
		Blank	All function control keys are enabled. In this case, the key mask entry must not contain any numbers. *Note:* Function control keys that are not masked off and that are not supported by the program cause an error message to be displayed, which indicates that an invalid key was pressed.
28	Enable command keys	Y	The command keys identified by alphabetic characters in the key mask entry (columns 64 through 79) are enabled (allowed). If the key mask entry contains no alphabetic characters, all command keys are disabled.
		N	The command keys identified by alphabetic characters in the key mask entry are disabled (not allowed). If the key mask entry contains no alphabetic characters, all command keys are enabled. If the operator presses a disabled command key, an error message is displayed. The operator can then press the Error Reset key and then the correct command key.
		R	The command key mask that is active for the display station is retained when this format is displayed.
		Blank	All command keys are enabled. In this case, the key mask entry must not contain any alphabetic characters. If a command key is pressed, the corresponding indicator (KA through KN, KP through KY) is set on in an RPG II program; 1 through 24 is returned in the control area function key area in a COBOL program.

S Specifications (continued)

Columns	Name	Entry	Explanation
29-30	Blink cursor	Y (column 29)	The cursor blinks when this display appears.
		N (column 29) or blank	The cursor does not blink.
		01-99	The cursor blinks only if the specified indicator is on.
31-32	Erase input fields	Y (column 31)	All unprotected input fields on the screen are erased, the keyboard is unlocked, and no output occurs. All D specifications are ignored. The use of Y is not recommended.
		N (column 31) or blank	The input fields are not erased.
		01-99	All unprotected input fields on the screen are erased and the keyboard is unlocked if the specified indicator is on.
33-34	Override fields	Y (column 33)	An override operation is performed. The use of Y is not recommended.
		N (column 33) or blank	The operation is not an override operation.
		01-99	An override operation is performed if the specified indicator is on. An override operation allows the screen to remain unchanged except for those fields that have indicators specified for them in columns 23 and 24 of the D specification, and except for those indicators that are on. The record displayed by RPG II/COBOL/BASIC is exactly the same whether or not override is specified when the indicator in columns 23 and 24 of the D specification is on.
35-36	Suppress input	Y (column 35)	No input is returned to the user program until a format is displayed with suppress input specified as N or with the specified indicator off.
		N (column 35) or blank	Input is returned to the user program.
		01-99	Input to the user program is suppressed if the specified indicator is on.
37-63		Blank	

S Specifications (continued)

Columns	Name	Entry	Explanation
64-79	Key mask		The key mask is a string of numbers and/or alphabetic characters that identify keys to be enabled or disabled when this format is displayed. The key mask must begin in column 64 and cannot contain embedded blanks. The numbers and the alphabetic characters can be intermixed. Numbers in the key mask identify function control keys: **Number** — **Function Control Key** 1 — Print 2 — Roll Up 3 — Roll Down 4 — Clear 5 — Help 6 — Record Backspace Alphabetic characters in the key mask identify command keys: **Alphabetic Character** — **Command Keys** A through N — 1 through 14, respectively P through Y — 15 through 24, respectively

D Specifications

Columns	Name	Entry	Explanation
1-5	Sequence number	Line number	Entry used to number the specification line.
6	Form type	D	Identification for a D specification.
7		*	Asterisk in this column identifies this line as a comment line.
7-12	Field name	Field name	Name of an input field, output field, or output/input field.
		Blank	This D specification line specifies only constant data.
13-14		Blank	
15-18	Field length	1-1919	The entry must be right-justified, but leading zeros are not required.
19-20	Line number	01-nn	Relative line number on which data appears. The actual line number is start line number (column 17 and 18 on the S specification) plus this line number, minus one. nn (maximum) = 24 - starting line number
21-22	Horizontal position	01-80	Column number of the first position of the field. Columns 19 through 22 cannot be 0101.
23-24	Output data	Y (column 23)	If constant data or a message identification code is also specified in columns 57 through 79, that constant data or the specified message is displayed in the field. If no constant data or message identification code is specified in columns 57 through 79, data from the user program output record is displayed.
		N (column 23) or blank	The field is not an output field.
		01-99	If the specified indicator is on when the format is displayed, data supplied by the user program is displayed in the field. If the specified indicator is off when the format is displayed, data specified in columns 57 through 79 is displayed. If no data is specified in columns 57 through 79, blanks are displayed. If the user program performs an override operation and the specified indicator is on, data supplied by the user program is displayed in the field. If the user program performs an override operation and the specified indicator is off, the field is unchanged.
25		Blank	

D Specifications (continued)

Columns	Name	Entry	Explanation
26	Input allowed	Y	The field is an input field.
		N or blank	The field is not an input field.
27	Data type	A	The field can contain only alphabetic data.
		B or blank	The field can contain only alphameric data.
		K	The field can contain Katakana characters.
		N	The field can contain only numeric data. Commas, a period, a plus sign, or a minus sign can also be entered in this field.
		S	The field can contain only signed numeric data; the last position of the field is reserved for a sign. Only decimal digits (0 through 9) can be entered in the field. The field can be from 2 to 16 characters long.
		R	The field contains data read from the magnetic stripe reader. The field can be up to 128 characters long. Nondisplay must be specified in columns 43 and 44 for the field.
		E	The field can contain alphanumeric and Katakana or ideographic data, but not both. The display station is set to enter alphameric data.
		F	The field can contain alphanumeric and Katakana or ideographic data, but not both. The display station is set to enter ideographic data.
		X	The field can contain only ideographic data.
28	Mandatory fill	Y	Operators must key in all or none of the field.
		N or blank	Operators can key in all, none, or part of the input field.
			Note: Mandatory fill and adjust/fill (column 31) cannot be specified for the same field.
29	Mandatory entry	Y	Operators must enter at least one character or a blank in the input field.
		N or blank	Operators can bypass the input field.
30	Self check	T	The input field is a modulus 10 self-check field.
		E	The input field is a modulus 11 self-check field.
		Blank	The input field is not a self-check field.

D Specifications (continued)

Columns	Name	Entry	Explanation
31	Adjust/fill	Z	Information entered into the field is right-justified, and unused positions are filled with zeros.
		B	Information entered into the field is right-justified, and unused positions are filled with blanks.
		Blank	Information entered in the field is right-justified, and blank fill is assumed only for signed numeric fields. *Note:* Mandatory fill (column 28) and adjust/fill cannot be specified for the same field.
32-33	Position cursor	Y (column 32)	Cursor appears at the first position of the input field when this format is displayed.
		N (column 32) or blank	Cursor does not appear at the first position of the input field.
		01-99	Cursor appears at the first position of the input field only if the specified indicator is on.
34	Enable Dup	Y	When the Dup key is pressed, the position of the cursor and the remainder of the field are filled with the duplicate character value (hex 1C), which is displayed as an asterisk (*). The duplicate characters must be processed by the user porgram.
		N or blank	The Dup key has no effect in the field.
35	Controlled field exit	Y	Cursor does not leave the input field until the operator presses a field exit key (Field Adv, Enter/Rec Adv, Field Exit, Field +, Field - [if the field is a signed-numeric field], Field Backspace, Home, Erase Input, or Dup).
		N or blank	Cursor automatically skips to the next unprotected field when the operator keys the last position of the field.
36	Auto record advance	Y	The input fields on the screen automatically return to the user program when one of the following occurs: • The operator enters the last character in the field. • The cursor is in the input field and the operator presses the Field Exit, Field +, or Field - key (if the field is a signed-numeric field).
		N or blank	Automatic record advance does not occur for this field.

D Specifications (continued)

Columns	Name	Entry	Explanation
37-38	Protect field	Y (column 37)	The cursor skips the field.
		N (column 37) or blank	The cursor does not skip the field.
		01-99	The cursor skips the field if the specified indicator is on. *Note:* If an override operation is used, this indicator is ignored.
39-40	High intensity	Y (column 39)	The field is displayed with high intensity.
		N (column 39) or blank	The field is displayed with normal intensity.
		01-99	The field is displayed with high intensity if the specified indicator is on. *Note:* High intensity, reverse image (columns 45-46), and underline (columns 47-48) cannot all be specified for the same field at the same time.
41-42	Blink field	Y (column 41)	The field blinks.
		N (column 41) or blank	The field does not blink.
		01-99	The field blinks if the specified indicator is on when the format is displayed.
43-44	Nondisplay	Y (column 43)	The field is nondisplay; that is, information in the field when the format is displayed or information entered into the field by the operator is not visible on the screen.
		N (column 43) or blank	The information in the field is displayed.
		01-99	The field is a nondisplay field if the specified indicator is on when the format is displayed.
45-46	Reverse image	Y (column 45)	The characters in the field appear as dark characters on a light background.
		N (column 45) or blank	The characters in the field appear as light characters on a dark background.
		01-99	The characters in the field appear as dark characters on a light background if the specified indicator is on when the format is displayed.

D Specifications (continued)

Columns	Name	Entry	Explanation
47-48	Underline	Y (column 47)	The field is underlined.
		N (column 47) or blank	The field is not underlined.
		01-99	The field is underlined if the specified indicator is on.
49	Column separators	Y	Each character position in the field is preceded by a column separator (a vertical line). The column separator does not require an additional character position.
		N or blank	Column separators are not used.
50-55		Blank	
56	Constant type	C	The constant information in columns 57 through 79 is to be displayed in the output field. C is required only if columns 57 through 79 are blank and you want to display all blanks in the field. C is invalid if an indicator is specified in columns 23 and 24.
		M	A message identification code and a message member identifier are entered in columns 57 through 79.
		Blank	If columns 57 through 79 contain constant information, that information is displayed. If columns 57 through 79 are blank, then information from the program output record area is displayed.
57-79	Constant data		This field specifies the information to be placed in an output or output/input field when the format is generated. If information is to be placed in the field, columns 57 through 79 should contain one of the following: • The actual information to be displayed. • A 4-digit message identification code in columns 57 through 60 and a 2-character message member identifier in columns 61 and 62. *Notes:* 1. If columns 57 through 79 are blank and the field is an output field (Y in column 23), then information from the program output record is displayed. 2. If a message identification code is specified in columns 57 through 79, then only 6 bytes need to be reserved for the field in the program output record area.

D Specifications (continued)

Columns	Name	Entry	Explanation
80	Continuation	X	If more than 23 characters of data are required, an X in column 80 indicates that the record is continued. Use columns 7 through 79 of the following record for the continued constant data. *Note:* A comment cannot follow a record with X in column 80.

192- TO 96-CHARACTER SET FOLD (#188E96)

From Character	From Code	To Character	To Code
( )	40	( )	40
( )	41	( )	40
â	42	a	81
ä	43	a	81
à	44	a	81
á	45	a	81
ã	46	a	81
å	47	a	81
ç	48	c	83
ñ	49	n	95
[	4A	( )	40
.	4B	.	4B
⟨	4C	⟨	4C
(	4D	(	4D
+	4E	+	4E
!	4F	!	4F
&	50	&	50
ê	51	e	85
ê	52	e	85
ë	53	e	85
è	54	e	85
í	55	i	89
î	56	i	89
ï	57	i	89
ì	58	i	89
β	59	s	A2
]	5A	( )	40
$	5B	$	5B
*	5C	*	5C
)	5D	)	5D
;	5E	;	5E
∧	5F	( )	40
–	60	–	60
/	61	/	61
Â	62	A	C1
Ä	63	A	C1
Ã	64	A	C1
Á	65	A	C1
À	66	A	C1
Å	67	A	C1
Ç	68	C	C3
Ñ	69	N	D5
¦	6A	¦	6A
,	6B	,	6B
%	6C	%	6C
_	6D	_	6D
⟩	6E	⟩	6E
?	6F	?	6F
ø	70	o	96
É	71	E	C5
Ê	72	E	C5
Ë	73	E	C5
È	74	E	C5
Í	75	I	C9
Î	76	I	C9
Ï	77	I	C9
Ì	78	I	C9
`	79	`	79
:	7A	:	7A
#	7B	#	7B
@	7C	@	7C
'	7D	'	7D
=	7E	=	7E
"	7F	"	7F
Ø	80	0	D6
a	81	a	81
b	82	b	82
c	83	c	83
d	84	d	84
e	85	e	85
f	86	f	86
g	87	g	87
h	88	h	88
i	89	i	89
«	8A	( )	40
»	8B	( )	40
ð	8C	d	84
≤	8D	( )	40
Þ	8E	( )	40
±	8F	( )	40
°	90	( )	40
j	91	j	91
k	92	k	92
l	93	l	93
m	94	m	94
n	95	n	95
o	96	o	96
p	97	p	97
q	98	q	98
r	99	r	99
a	9A	a	81
º	9B	o	96
æ	9C	( )	40
ª	9D	( )	40
Æ	9E	( )	40
¤	9F	( )	40

192- TO 96-CHARACTER SET FOLD (#188E96) (continued)

From		To		From		To	
Character	Code	Character	Code	Character	Code	Character	Code
μ	A0	( )	40	}	D0	}	D0
~	A1	~	A1	J	D1	J	D1
s	A2	s	A2	K	D2	K	D2
t	A3	t	A3	L	D3	L	D3
u	A4	u	A4	M	D4	M	D4
v	A5	v	A5	N	D5	N	D5
w	A6	w	A6	O	D6	O	D6
x	A7	x	A7	P	D7	P	D7
y	A8	y	A8	Q	D8	Q	D8
z	A9	z	A9	R	D9	R	D9
¡	AA	( )	40	≥	DA	( )	40
¿	AB	( )	40	û	DB	u	A4
Ð	AC	D	C4	ü	DC	u	A4
†	AD	( )	40	ù	DD	u	A4
Þ	AE	( )	40	ú	DE	u	A4
®	AF	( )	40	ÿ	DF	y	A8
¢	B0	( )	40	\	E0	\	E0
£	B1	( )	40	( )	E1	( )	E1
¥	B2	( )	40	S	E2	S	E2
Pts	B3	( )	40	T	E3	T	E3
ƒ	B4	( )	40	U	E4	U	E4
§	B5	( )	40	V	E5	V	E5
¶	B6	( )	40	W	E6	W	E6
¼	B7	( )	40	X	E7	X	E7
½	B8	( )	40	Y	E8	Y	E8
¾	B9	( )	40	Z	E9	Z	E9
¬	BA	( )	40	²	EA	( )	40
|	BB	( )	40	ô	EB	O	D6
≠	BC	( )	40	ö	EC	O	D6
··	BD	( )	40	ò	ED	O	D6
'	BE	( )	40	ó	EE	O	D6
=	BF	( )	40	õ	EF	O	D6
(	C0	(	C0	0	F0	0	F0
A	C1	A	C1	1	F1	1	F1
B	C2	B	C2	2	F2	2	F2
C	C3	C	C3	3	F3	3	F3
D	C4	D	C4	4	F4	4	F4
E	C5	E	C5	5	F5	5	F5
F	C6	F	C6	6	F6	6	F6
G	C7	G	C7	7	F7	7	F7
H	C8	H	C8	8	F8	8	F8
I	C9	I	C9	9	F9	9	F9
-	CA	( )	40	³	FA	( )	40
ô	CB	o	96	Û	FB	U	E4
ö	CC	o	96	Ü	FC	U	E4
ò	CD	o	96	Ù	FD	U	E4
ó	CE	o	96	Ú	FE	U	E4
õ	CF	o	96				

192- TO 64-CHARACTER SET FOLD (#188E64)

From		To	
Character	Code	Character	Code
( )	40	( )	40
( )	41	( )	40
â	42	A	C1
ä	43	A	C1
à	44	A	C1
á	45	A	C1
ã	46	A	C1
å	47	A	C1
ç	48	C	C3
ñ	49	N	D5
[	4A	( )	40
.	4B	.	4B
‹	4C	‹	4C
(	4D	(	4D
+	4E	+	4E
!	4F	!	4F
&	50	&	50
ê	51	E	C5
ê	52	E	C5
ë	53	E	C5
è	54	E	C5
í	55	I	C9
î	56	I	C9
ï	57	I	C9
ì	58	I	C9
β	59	S	E2
]	5A	( )	40
$	5B	$	5B
*	5C	*	5C
)	5D	)	5D
;	5E	;	5E
∧	5F	( )	40
–	60	–	60
/	61	/	61
Â	62	A	C1
Ä	63	A	C1
Ã	64	A	C1
Á	65	A	C1
À	66	A	C1
Å	67	A	C1
Ç	68	C	C3
Ñ	69	N	D5
¦	6A	( )	40
,	6B	,	6B
%	6C	%	6C
_	6D	–	6D
›	6E	›	6E
?	6F	?	6F

From		To	
Character	Code	Character	Code
ø	70	O	D6
É	71	E	C5
Ê	72	E	C5
Ë	73	E	C5
È	74	E	C5
Í	75	I	C9
Î	76	I	C9
Ï	77	I	C9
Ì	78	I	C9
`	79	`	79
:	7A	:	7A
#	7B	#	7B
@	7C	@	7C
'	7D	'	7D
=	7E	=	7E
"	7F	"	7F
Ø	80	O	D6
a	81	A	C1
b	82	B	C2
c	83	C	C3
d	84	D	C4
e	85	E	C5
f	86	F	C6
g	87	G	C7
h	88	H	C8
i	89	I	C9
«	8A	( )	40
»	8B	( )	40
ð	8C	D	C4
≤	8D	( )	40
Þ	8E	( )	40
±	8F	( )	40
°	90	( )	40
j	91	J	D1
k	92	K	D2
l	93	L	D3
m	94	M	D4
n	95	N	D5
o	96	O	D6
p	97	P	D7
q	98	Q	D8
r	99	R	D9
ª	9A	A	C1
º	9B	O	D6
æ	9C	( )	40
»	9D	( )	40
Æ	9E	( )	40
¤	9F	( )	40

192- TO 64-CHARACTER SET FOLD (#188E64) (continued)

From		To	
Character	Code	Character	Code
μ	A0	( )	40
~	A1	( )	40
s	A2	S	E2
t	A3	T	E3
u	A4	U	E4
v	A5	V	E5
w	A6	W	E6
x	A7	X	E7
y	A8	Y	E8
z	A9	Z	E9
¡	AA	( )	40
¿	AB	( )	40
Ð	AC	D	C4
†	AD	( )	40
Þ	AE	( )	40
®	AF	( )	40
¢	B0	( )	40
£	B1	( )	40
¥	B2	( )	40
Pts	B3	( )	40
ƒ	B4	( )	40
§	B5	( )	40
¶	B6	( )	40
¼	B7	( )	40
½	B8	( )	40
¾	B9	( )	40
¬	BA	( )	40
¦	BB	( )	40
≠	BC	( )	40
..	BD	( )	40
'	BE	( )	40
=	BF	( )	40
{	C0	( )	40
A	C1	A	C1
B	C2	B	C2
C	C3	C	C3
D	C4	D	C4
E	C5	E	C5
F	C6	F	C6
G	C7	G	C7
H	C8	H	C8
I	C9	I	C9
–	CA	( )	40
ô	CB	O	D6
ö	CC	O	D6
õ	CD	O	D6
ő	CE	O	D6
Õ	CF	O	D6

From		To	
Character	Code	Character	Code
}	D0	( )	40
J	D1	J	D1
K	D2	K	D2
L	D3	L	D3
M	D4	M	D4
N	D5	N	D5
O	D6	O	D6
P	D7	P	D7
Q	D8	Q	D8
R	D9	R	D9
≥	DA	( )	40
û	DB	U	E4
ü	DC	U	E4
ù	DD	U	E4
ú	DE	U	E4
ÿ	DF	Y	E8
\	E0	\	E0
( )	E1	( )	E1
S	E2	S	E2
T	E3	T	E3
U	E4	U	E4
V	E5	V	E5
W	E6	W	E6
X	E7	X	E7
Y	E8	Y	E8
Z	E9	Z	E9
²	EA	( )	40
ô	EB	O	D6
ö	EC	O	D6
ò	ED	O	D6
ó	EE	O	D6
ō	EF	O	D6
0	F0	0	F0
1	F1	1	F1
2	F2	2	F2
3	F3	3	F3
4	F4	4	F4
5	F5	5	F5
6	F6	6	F6
7	F7	7	F7
8	F8	8	F8
9	F9	9	F9
³	FA	( )	40
û	FB	U	E4
ü	FC	U	E4
ù	FD	U	E4
ú	FE	U	E4

192- TO 48-CHARACTER SET FOLD (#188E48)

From		To	
Character	Code	Character	Code
( )	40	( )	40
( )	41	( )	40
â	42	A	C1
ä	43	A	C1
à	44	A	C1
á	45	A	C1
ã	46	A	C1
å	47	A	C1
ç	48	C	C3
ñ	49	N	D5
[	4A	( )	40
.	4B	.	4B
<	4C	( )	40
(	4D	( )	4D
+	4E	+	4E
!	4F	( )	4F
&	50	&	50
é	51	E	C5
ê	52	E	C5
ë	53	E	C5
è	54	E	C5
í	55	I	C9
î	56	I	C9
ï	57	I	C9
ì	58	I	C9
β	59	S	E2
]	5A	( )	40
$	5B	$	5B
*	5C	*	5C
)	5D	( )	5D
;	5E	( )	5E
∧	5F	( )	40
–	60	–	60
/	61	/	61
Â	62	A	C1
Ä	63	A	C1
Ã	64	A	C1
Á	65	A	C1
À	66	A	C1
Å	67	A	C1
Ç	68	C	C3
Ñ	69	N	D5
¦	6A	( )	40
,	6B	,	6B
%	6C	%	6C
_	6D	( )	40
>	6E	( )	40
?	6F	( )	40

From		To	
Character	Code	Character	Code
ø	70	O	D6
É	71	E	C5
Ê	72	E	C5
Ë	73	E	C5
È	74	E	C5
Í	75	I	C9
Î	76	I	C9
Ï	77	I	C9
Ì	78	I	C9
`	79	( )	40
:	7A	( )	40
#	7B	#	7B
@	7C	@	7C
'	7D	'	7D
=	7E	( )	40
"	7F	( )	40
Ø	80	O	D6
a	81	A	C1
b	82	B	C2
c	83	C	C3
d	84	D	C4
e	85	E	C5
f	86	F	C6
g	87	G	C7
h	88	H	C8
i	89	I	C9
«	8A	( )	40
»	8B	( )	40
ð	8C	D	C4
≤	8D	( )	40
þ	8E	( )	40
±	8F	( )	40
°	90	( )	40
j	91	J	D1
k	92	K	D2
l	93	L	D3
m	94	M	D4
n	95	N	D5
o	96	O	D6
p	97	P	D7
q	98	Q	D8
r	99	R	D9
ª	9A	A	C1
º	9B	O	D6
æ	9C	( )	40
¸	9D	( )	40
Æ	9E	( )	40
¤	9F	( )	40

192- TO 48-CHARACTER SET FOLD (#188E48) (continued)

From		To	
Character	Code	Character	Code
µ	A0	( )	40
~	A1	( )	40
s	A2	S	E2
t	A3	T	E3
u	A4	U	E4
v	A5	V	E5
w	A6	W	E6
x	A7	X	E7
y	A8	Y	E8
z	A9	Z	E9
¡	AA	( )	40
¿	AB	( )	40
Ð	AC	D	C4
†	AD	( )	40
Þ	AE	( )	40
®	AF	( )	40
¢	B0	( )	40
£	B1	( )	40
¥	B2	( )	40
Pts	B3	( )	40
ƒ	B4	( )	40
§	B5	( )	40
¶	B6	( )	40
¼	B7	( )	40
½	B8	( )	40
¾	B9	( )	40
¬	BA	( )	40
¦	BB	( )	40
≠	BC	( )	40
..	BD	( )	40
'	BE	( )	40
=	BF	( )	40
{	C0	( )	40
A	C1	A	C1
B	C2	B	C2
C	C3	C	C3
D	C4	D	C4
E	C5	E	C5
F	C6	F	C6
G	C7	G	C7
H	C8	H	C8
I	C9	I	C9
–	CA	–	CA
ô	CB	O	D6
ö	CC	O	D6
ò	CD	O	D6
ó	CE	O	D6
õ	CF	O	D6
}	D0	( )	40
J	D1	J	D1
K	D2	K	D2
L	D3	L	D3
M	D4	M	D4
N	D5	N	D5
O	D6	O	D6
P	D7	P	D7
Q	D8	Q	D8
R	D9	R	D9
≥	DA	( )	40
û	DB	U	E4
ü	DC	U	E4
ù	DD	U	E4
ú	DE	U	E4
ÿ	DF	Y	E8
\	E0	( )	E0
( )	E1	( )	E1
S	E2	S	E2
T	E3	T	E3
U	E4	U	E4
V	E5	V	E5
W	E6	W	E6
X	E7	X	E7
Y	E8	Y	E8
Z	E9	Z	E9
²	EA	( )	40
ô	EB	O	D6
ö	EC	O	D6
ò	ED	O	D6
ó	EE	O	D6
õ	EF	O	D6
0	F0	0	F0
1	F1	1	F1
2	F2	2	F2
3	F3	3	F3
4	F4	4	F4
5	F5	5	F5
6	F6	6	F6
7	F7	7	F7
8	F8	8	F8
9	F9	9	F9
³	FA	( )	40
û	FB	U	E4
ü	FC	U	E4
ù	FD	U	E4
ú	FE	U	E4

96- TO 64-CHARACTER SET FOLD (#96E64)

From		To	
Character	Code	Character	Code
( )	40	( )	40
( )	41	( )	40
( )	42	( )	40
( )	43	( )	40
( )	44	( )	40
( )	45	( )	40
( )	46	( )	40
( )	47	( )	40
( )	48	( )	40
( )	49	( )	40
¢	4A	¢	4A
.	4B	.	4B
<	4C	<	4C
(	4D	(	4D
+	4E	+	4E
|	4F	|	4F
&	50	&	50
( )	51	( )	40
( )	52	( )	40
( )	53	( )	40
( )	54	( )	40
( )	55	( )	40
( )	56	( )	40
( )	57	( )	40
( )	58	( )	40
( )	59	( )	40
!	5A	!	5A
$	5B	$	5B
*	5C	*	5C
)	5D	)	5D
;	5E	;	5E
¬	5F	¬	5F
–	60	–	60
/	61	/	61
( )	62	( )	40
( )	63	( )	40
( )	64	( )	40
( )	65	( )	40
( )	66	( )	40
( )	67	( )	40
( )	68	( )	40
( )	69	( )	40
¦	6A	( )	40
,	6B	,	6B
%	6C	%	6C
_	6D	_	6D
>	6E	>	6E
?	6F	?	6F

From		To	
Character	Code	Character	Code
( )	70	( )	40
( )	71	( )	40
( )	72	( )	40
( )	73	( )	40
( )	74	( )	40
( )	75	( )	40
( )	76	( )	40
( )	77	( )	40
( )	78	( )	40
`	79	`	79
:	7A	:	7A
#	7B	#	7B
@	7C	@	7C
'	7D	'	7D
=	7E	=	7E
"	7F	"	7F
( )	80	( )	40
a	81	A	C1
b	82	B	C2
c	83	C	C3
d	84	D	C4
e	85	E	C5
f	86	F	C6
g	87	G	C7
h	88	H	C8
i	89	I	C9
( )	8A	( )	40
( )	8B	( )	40
( )	8C	( )	40
( )	8D	( )	40
( )	8E	( )	40
( )	8F	( )	40
( )	90	( )	40
j	91	J	D1
k	92	K	D2
l	93	L	D3
m	94	M	D4
n	95	N	D5
o	96	O	D6
p	97	P	D7
q	98	Q	D8
r	99	R	D9
( )	9A	( )	40
( )	9B	( )	40
( )	9C	( )	40
( )	9D	( )	40
( )	9E	( )	40
( )	9F	( )	40

96- TO 64-CHARACTER SET FOLD (#96E64) (continued)

From		To	
Character	Code	Character	Code
( )	A0	( )	40
( )	A1	( )	40
s	A2	S	E2
t	A3	T	E3
u	A4	U	E4
v	A5	V	E5
w	A6	W	E6
x	A7	X	E7
y	A8	Y	E8
z	A9	Z	E9
( )	AA	( )	40
( )	AB	( )	40
( )	AC	( )	40
( )	AD	( )	40
( )	AE	( )	40
( )	AF	( )	40
( )	B0	( )	40
( )	B1	( )	40
( )	B2	( )	40
( )	B3	( )	40
( )	B4	( )	40
( )	B5	( )	40
( )	B6	( )	40
( )	B7	( )	40
( )	B8	( )	40
( )	B9	( )	40
( )	BA	( )	40
( )	BB	( )	40
( )	BC	( )	40
( )	BD	( )	40
( )	BE	( )	40
( )	BF	( )	40
(	C0	( )	40
A	C1	A	C1
B	C2	B	C2
C	C3	C	C3
D	C4	D	C4
E	C5	E	C5
F	C6	F	C6
G	C7	G	C7
H	C8	H	C8
I	C9	I	C9
( )	CA	( )	40
( )	CB	( )	40
( )	CC	( )	40
( )	CD	( )	40
( )	CE	( )	40
( )	CF	( )	40

From		To	
Character	Code	Character	Code
( )	D0	( )	40
J	D1	J	D1
K	D2	K	D2
L	D3	L	D3
M	D4	M	D4
N	D5	N	D5
O	D6	O	D6
P	D7	P	D7
Q	D8	Q	D8
R	D9	R	D9
( )	DA	( )	40
( )	DB	( )	40
( )	DC	( )	40
( )	DD	( )	40
( )	DE	( )	40
( )	DF	( )	40
\	E0	\	E0
( )	E1	( )	40
S	E2	S	E2
T	E3	T	E3
U	E4	U	E4
V	E5	V	E5
W	E6	W	E6
X	E7	X	E7
Y	E8	Y	E8
Z	E9	Z	E9
( )	EA	( )	40
( )	EB	( )	40
( )	EC	( )	40
( )	ED	( )	40
( )	EE	( )	40
( )	EF	( )	40
0	F0	0	F0
1	F1	1	F1
2	F2	2	F2
3	F3	3	F3
4	F4	4	F4
5	F5	5	F5
6	F6	6	F6
7	F7	7	F7
8	F8	8	F8
9	F9	9	F9
( )	FA	( )	40
( )	FB	( )	40
( )	FC	( )	40
( )	FD	( )	40
( )	FE	( )	40

96- TO 48-CHARACTER SET FOLD (#96E48)

From		To	
Character	Code	Character	Code
( )	40	( )	40
( )	41	( )	40
( )	42	( )	40
( )	43	( )	40
( )	44	( )	40
( )	45	( )	40
( )	46	( )	40
( )	47	( )	40
( )	48	( )	40
( )	49	( )	40
¢	4A	( )	40
.	4B		4B
<	4C	( )	40
(	4D	( )	40
+	4E	+	4E
|	4F	( )	40
&	50	&	50
( )	51	( )	40
( )	52	( )	40
( )	53	( )	40
( )	54	( )	40
( )	55	( )	40
( )	56	( )	40
( )	57	( )	40
( )	58	( )	40
( )	59	( )	40
!	5A	( )	40
$	5B	$	5B
*	5C	*	5C
)	5D	( )	40
;	5E	( )	40
¬	5F	( )	40
–	60	–	60
/	61	/	61
( )	62	( )	40
( )	63	( )	40
( )	64	( )	40
( )	65	( )	40
( )	66	( )	40
( )	67	( )	40
( )	68	( )	40
( )	69	( )	40
¦	6A	( )	40
,	6B	,	6B
%	6C	%	6C
_	6D	( )	40
>	6E	( )	40
?	6F	( )	40

From		To	
Character	Code	Character	Code
( )	70	( )	40
( )	71	( )	40
( )	72	( )	40
( )	73	( )	40
( )	74	( )	40
( )	75	( )	40
( )	76	( )	40
( )	77	( )	40
( )	78	( )	40
`	79	( )	40
:	7A	( )	40
#	7B	#	7B
@	7C	@	7C
'	7D	'	7D
=	7E	( )	40
"	7F	( )	40
( )	80	( )	40
a	81	A	C1
b	82	B	C2
c	83	C	C3
d	84	D	C4
e	85	E	C5
f	86	F	C6
g	87	G	C7
h	88	H	C8
i	89	I	C9
( )	8A	( )	40
( )	8B	( )	40
( )	8C	( )	40
( )	8D	( )	40
( )	8E	( )	40
( )	8F	( )	40
( )	90	( )	40
j	91	J	D1
k	92	K	D2
l	93	L	D3
m	94	M	D4
n	95	N	D5
o	96	O	D6
p	97	P	D7
q	98	Q	D8
r	99	R	D9
( )	9A	( )	40
( )	9B	( )	40
( )	9C	( )	40
( )	9D	( )	40
( )	9E	( )	40
( )	9F	( )	40

96- TO 48-CHARACTER SET FOLD (#96E48) (continued)

From		To	
Character	Code	Character	Code
( )	A0	( )	40
( )	A1	( )	40
s	A2	S	E2
t	A3	T	E3
u	A4	U	E4
v	A5	V	E5
w	A6	W	E6
x	A7	X	E7
y	A8	Y	E8
z	A9	Z	E9
( )	AA	( )	40
( )	AB	( )	40
( )	AC	( )	40
( )	AD	( )	40
( )	AE	( )	40
( )	AF	( )	40
( )	B0	( )	40
( )	B1	( )	40
( )	B2	( )	40
( )	B3	( )	40
( )	B4	( )	40
( )	B5	( )	40
( )	B6	( )	40
( )	B7	( )	40
( )	B8	( )	40
( )	B9	( )	40
( )	BA	( )	40
( )	BB	( )	40
( )	BC	( )	40
( )	BD	( )	40
( )	BE	( )	40
( )	BF	( )	40
(	C0	( )	40
A	C1	A	C1
B	C2	B	C2
C	C3	C	C3
D	C4	D	C4
E	C5	E	C5
F	C6	F	C6
G	C7	G	C7
H	C8	H	C8
I	C9	I	C9
( )	CA	( )	40
( )	CB	( )	40
( )	CC	( )	40
( )	CD	( )	40
( )	CE	( )	40
( )	CF	( )	40

From		To	
Character	Code	Character	Code
( )	D0	( )	40
J	D1	J	D1
K	D2	K	D2
L	D3	L	D3
M	D4	M	D4
N	D5	N	D5
O	D6	O	D6
P	D7	P	D7
Q	D8	Q	D8
R	D9	R	D9
( )	DA	( )	40
( )	DB	( )	40
( )	DC	( )	40
( )	DD	( )	40
( )	DE	( )	40
( )	DF	( )	40
\	E0	( )	40
( )	E1	( )	40
S	E2	S	E2
T	E3	T	E3
U	E4	U	E4
V	E5	V	E5
W	E6	W	E6
X	E7	X	E7
Y	E8	Y	E8
Z	E9	Z	E9
( )	EA	( )	40
( )	EB	( )	40
( )	EC	( )	40
( )	ED	( )	40
( )	EE	( )	40
( )	EF	( )	40
0	F0	0	F0
1	F1	1	F1
2	F2	2	F2
3	F3	3	F3
4	F4	4	F4
5	F5	5	F5
6	F6	6	F6
7	F7	7	F7
8	F8	8	F8
9	F9	9	F9
( )	FA	( )	40
( )	FB	( )	40
( )	FC	( )	40
( )	FD	( )	40
( )	FE	( )	40

Appendix M. Multinational Character Set Conversion Utility Programs

FUNCTIONS

The Multinational Character Set (MCS) feature is a System/34 enhancement that makes available an expanded multinational character set of 188 characters for most support country language groups on System/34. The set includes 112 alphabetic characters, 10 numeric characters, and 66 special characters. Some characters of the expanded character set have been assigned different hexadecimal values than they had in the past. Because of these changes in hexadecimal values, you need a conversion utility to change the National Language Version (NLV) value to the new MCS value so that printed or displayed graphics correspond to the proper MCS value. See Table M-1 at the end of this chapter for a description of specific language group characters affected by the multinational character set.

The Multinational Character Set Conversion Utility (MCSCU) is a set of utility programs, using the System/34 System Support Program Product, that provide the conversion capability. The MCSCU programs provide assistance in converting library source members, library procedure members, and data files from the NLV hexadecimal representation to the new MCS value, and vice versa.

The MCSCU programs and procedures:

- Prompt for necessary information to execute the utility
- Convert files in place
- Convert library members in a work file and place the results in the same or an alternately specified library
- Print an audit list of examined members
- Print a list of affected records (optional)
- Modify records (optional)
- Count affected records in a member and/or a file (optional)

When using the Multinational Character Set Conversion Utility, you must specify the type of records being converted: source statements, procedure statements or data files. The types of statements in library members that are subject to conversion include RPG, COBOL, FORTRAN, assembler, sort, DFU, message member, screen formats, WSU, OCL, and menu member.

5250 HARDWARE SUPPORT

In addition to currently supported devices, the MCSCU supports different models of the 5250 display stations that have the MCS feature. Installing the 5250 Multinational Character Set feature requires converting any internally stored or saved graphic data that has a new hexadecimal value in the MCS. For more information about installing MCSCU, see the *Installation and Modification Reference Manual.*

The MCS hardware feature does not have to be installed on the system to execute MCSCU. But if it is not, all features may not be accessible; for example, keying in the new hexadecimal values for manual changes, keying in user-supplied alternate conversion tables, or printing new characters.

INITIATING THE CONVERSION UTILITY

On the System/34 command display, enter the Multinational Character Set Conversion Utility procedure MCSCONV. There are no parameters.

You are automatically prompted for the type of conversion work you would like to perform, working either with library members or data files. The prompt display is shown in Figure M-1:

```
                MULTINATIONAL CHARACTER SET CONVERSION UTILITY

        Convert library source and procedure members and data files
        from a National language to Multinational Character Set or
        from Multinational Character Set to National language.

1. Convert Selected Library Members
2. Convert Data Files
3. Terminate Conversion Utility


Enter Number Of Option Desired -->
```

Figure M-1. Conversion Options

If you select option 1, you are prompted for information necessary to process library members. See *Library Member Conversion* later in this appendix.

If you select option 2, you are prompted for information necessary to process a data file. See *Data File Conversion* later in this appendix.

If you select option 3, MCSCU is terminated.

If you select none of the options, a message is issued. If you wish to continue, the prompt screen in Figure M-1 is redisplayed to correct the entered value.

LIBRARY MEMBER CONVERSION

Basic components of any installation are the application programming resources. These resources take on three forms in a library: procedure members, source members, and load members. Source members in turn can be categorized into different types: RPG, COBOL, FORTRAN, assembler, sort, DFU, message members, screen formats, WSU, and menu members.

Each type of source statement may contain characters in the form of constants, screen prompts, printer headings, and so on, that have taken on a new hexadecimal value under MCS. When a program is executed on a system with the MCS feature, screen prompts and printed reports display the graphic associated with the MCS value instead of the NLV graphic. MCSCU assists the user in finding characters within source statements and procedures that are MCS sensitive. The word sensitive means that a character match has been found in the record being scanned with a from character in the conversion table being used. Depending on where that character exists in a record, MCSCU may or may not be able to modify it. MCSCU cannot perform a complete automatic conversion of the entire record in all types of library members. For example, if you have the following OCL statement in a procedure:

```
// LOAD $ABC
```

The dollar sign ($) is an MCS sensitive character (in some languages), but it cannot be automatically converted because $ABC is the name of the program. Another example is Interactive Communication Feature format names that begin with $$ and must not be changed. You must examine the records that were found to be sensitive to determine if a manual change is desirable or required. MCSCU does not process load members. Load members are updated when the associated source is recompiled.

MCSCU copies user-selected source or procedure members to a work file, performs the analysis and updates on the work file, and then copies the members in the work file back to the specified output library.

You must select which library members are to be converted during an execution of MCSCU. The library members selected must all be of the same type; for example, RPG, SFGR, or procedures. Do not mix member types in the work file. To assist you in selecting members and creating the work file, MCSCU prompts you for the necessary information. By responding to the prompts in Figure M-2 you can:

- Obtain a directory listing of a specified library from which to select members
- Copy individual members or groups of members from a library to the work file
- Execute MCSCU against a work file previously created

You should use the directory list (option 1, Figure M-2) to obtain a listing of all procedure or source members in the specified library. If the System/34 MCS feature is installed, characters other than what the user is expecting to see may appear in those member names. If new characters do appear, you must enter those values when selecting the members. It is a good practice to save your original source and procedures on diskette as a precautionary measure.

Figure M-2 is the prompt display for performing the selection process. An explanation of each prompt and option follows the display.

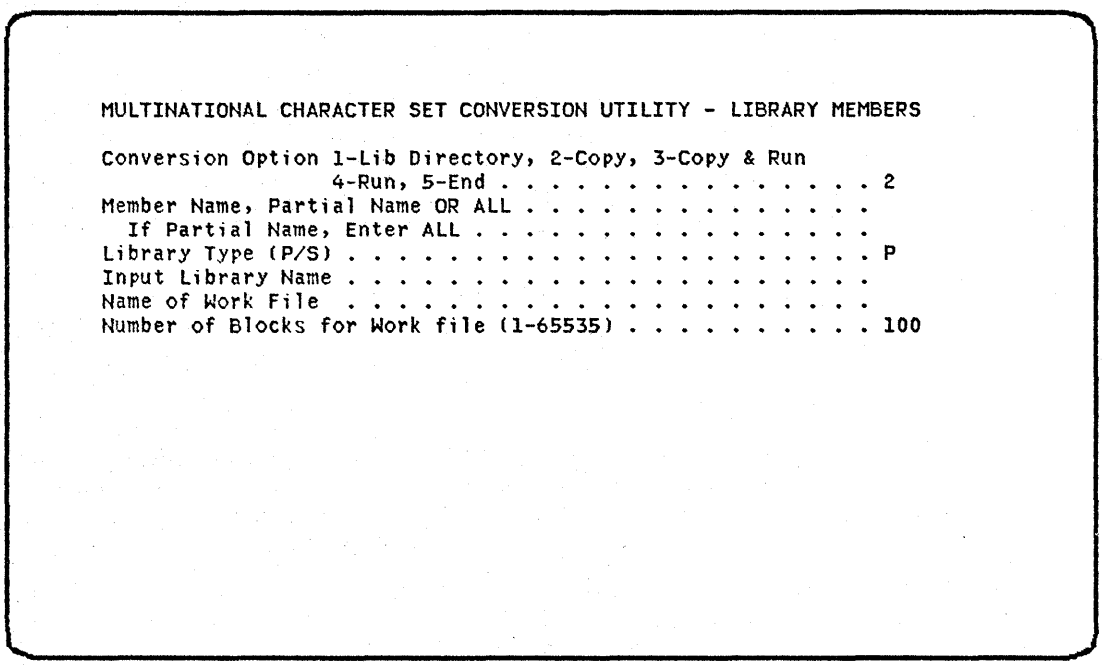


Figure M-2. Library Member Copy

1-Lib Directory: Allows you to obtain a library directory listing of the source or procedure member names in the specified library. The directory listing can be used to select members requiring conversion. MCSCU will then redisplay the library member copy prompt screen to prompt for the next request.

2-Copy: Allows you to copy selected members to the specified file. It does not immediately run the conversion utility. This option redisplays the library member copy prompt screen to prompt for additional members to copy to the work file. This is the default.

For example, you can copy all procedures that begin with XX and YY before running the conversion utility.

3-Copy & Run: Allows you to copy selected members to the specified file. It then begins the analysis of the work file. See Library Member Modification later in this appendix.

4-Run: Does not copy any additional members to the work file, but begins the analysis of the work file. See *Library Member Modification* later in this chapter.

5-End: You want to stop using MCSCU. The procedure terminates.

Member Name, Partial Name OR ALL: Allows you to enter a specific member name or a partial name of a group of members to be copied. This entry is required for copy options. If ALL is entered, all procedures or source members in the library are copied to the work file. You must ensure that all source members in the library are of the same type if ALL is entered.

If Partial Name: You must enter ALL to obtain a group copy. Anything other than ALL is ignored. This parameter is ignored if ALL is entered for a specific member or partial name parameter.

Library Type (P/S): Allows you to specify whether source or procedure statements are desired from the library. P is entered for procedure (default), and S is entered for source. This entry is required for directory listing or copy options.

Input Library Name: Allows you to specify the name of the library from which the members are to be copied. The default is session library. This entry is required for directory listing or copy options.

Name of Work File: The name of the work file the conversion utility will create or add to that contains the members to be analyzed and processed. If the file exists, additional members are copied to it. This entry is required for copy and run options.

Number of Blocks for Work File (1-99999): Specifies the size of the work file in blocks if the file needs to be allocated. You must ensure that the file is big enough to contain all of the members requested. The default is 100 blocks. Each block can accommodate approximately 21 records.

The work file is given the extended disk file attribute (if the support is available) when it is created. This allows the file to expand if it becomes full when copying members to the work file. You do not have to indicate any extend file size value. When support is active, the work file is extended by 25 blocks.

When you have selected all desired procedure members or source members of a specific type and copied them to the work file for this execution of MCSCU, you should select a run option. What occurs during the analysis and modification of the work file is explained in *Library Member Modification.*

LIBRARY MEMBER MODIFICATION

After you have selected the library members to be converted and have copied them to the MCSCU work file, you must specify information specifically related to how the work file should be analyzed. The information needed includes:

- What types of source or procedure members are in the work file?
- Which library should the modified members be copied back to?
- Which National language conversion table should be used?
- What is the direction of the conversion?
  - National language to MCS
  - MCS to National language
- What type of analysis options are desired?

You are prompted for this additional analysis information by the display shown in Figure M-3. A description of each of the types of analysis information follows the display.

```
          MULTINATIONAL CHARACTER SET CONVERSION UTILITY

Member Type In File (1-8) . . . . . . . . . . . . . . . . . . . . . 2
  1 - RPG             4 - Menu Members      7 - WSU
  2 - Procedure       5 - Screen Format     8 - Other
  3 - Message Member  6 - Sort Specification

Output Library Name . . . . . . . . . . . . . . . . . . . . . . . .
National Language . . . . . . . . . . . . . . . . . . . . . . . . .
Conversion Option (1-National to MCS/2-MCS to National) . . . . . 1
List Sensitive Records (1-Yes/0-No) . . . . . . . . . . . . . . . . 1
Count Number of Sensitive Records (1-Yes/0-No). . . . . . . . . . . 1
Modify Sensitive Records (1-Yes/0-No) . . . . . . . . . . . . . . . 0
```

Figure M-3. Library Member Analysis Information

Member Type In File: You select one of the numbered member types listed on the screen:

1—RPG source member.
2—Procedure member.
3—Message member source.
4—Menu member source.
5—Screen format source.
6—Sort specifications source.
7—WSU member source.
8—Other—A source member (COBOL, FORTRAN, and so on) that is not supported by a specific member type option. This option performs the list, count, and modify functions (described later) that were requested for all characters in each record. This entry is required. The default is procedure member.

Note: While performing the copy function, MCSCU did not need to know the type of source member being selected, but, in order for MCSCU to perform the analysis of the members in the work file (being sensitive to the different statement types), you now must specify this information.

Output Library Name: The name of the library where the work file is to be copied after the conversion occurs. The copy function occurs only if modify statements are requested. This is a required entry when the MODIFY-yes option is selected.

National Language: Specify the language group (NLV) requested from the following list:

USA = United States and Canada
GERMANY = Austria/Germany
BELGIUM = Belgium
BRAZIL = Brazil
CANADA = Canada (French)
DENMARK = Denmark/Norway
FINLAND = Finland/Sweden
FRANCE = France
ITALY = Italy
JAPAN = Japan (English)
PORTUGAL = Portugal
SPAIN = Spain
SPANISH = Spanish-Speaking
UK = United Kingdom
OTHER = User-Supplied

This entry is required. MCSCU supplies the conversion tables for each NLV. If you wish to supply your own table, you do so by completing the messages in the MCSCU message member (#MN#M2) for this purpose. When OTHER is selected, the messages in #MN#M2 are used. See *Conversion Tables* in this appendix for additional information.

Conversion Option: MCSCU allows conversion from NLV to MCS by option 1, or MCS to NLV by option 2. This entry is required. The default is option 1.

List Sensitive Records: MCSCU lists the sensitive/affected records within a member. Option 1 is list; option 0 is do not list. The default is 1. See *Library Statement List* later in this appendix for additional information.

Count Number of Sensitive Records: MCSCU counts the number of statements that are sensitive in a given member and the total number for all members in the work file. Option 1 is count; option 0 is do not count. The default is 1. See *Library Statement Count* later in this appendix for additional information.

Modify Sensitive Records: MCSCU either modifies the statements as it scans them or leaves them as is. Option 1 is modify; option 0 is do not modify. The default is 0. See *Library Statement Modify* later in this appendix for an explanation of what modify means per member statement type.

MCSCU for library members allows you to:

1. Obtain an audit list of all members examined in the disk file
2. Obtain a list of the specific statements in a source or procedure member that have MCS sensitive characters
3. Count all the sensitive records in a member and the file
4. Modify the source statements, based on the specific statement type

Note: This conversion utility should not be executed against any System/34 SSP- or PP-supplied message member, SFGR member or procedure member.

If you indicate no for list, count, and modify, an error is issued. All information on the prompt screens that needs to be available to the conversion programs is passed via the local data area. The conversion utility uses the last 100 bytes (bytes 157 through 256) of the local data area.

The program that performs the actual work file analysis and modification is an EVOKED program. This allows the actual conversion program to execute as a separate job and allows control to return immediately to the MCSCONV procedure to prompt for additional conversion activity. Thus, it is possible to have several different work files being analyzed and modified at the same time. Because the analysis and modification of the work file is still in progress, no additional members should be copied to the work file, or the work file should not be deleted, until the EVOKED job has completed.

Conversion Tables

All conversion tables for the valid language groups are supplied in the MCSCU message member #MN#M1. These tables allow you to convert from a specific language group to MCS and vice versa. Two additional messages in the MCSCU message member #MN#M2 allow you to supply your own conversion tables for converting from and converting to languages. If you wish to supply your own tables:

- Modify the specified messages in the source message member (#MN#M2) using SEU, or create a new #MN#M2 using $MAINT.

- Use the following operation control language to rebuild the MCSCU message member #MN#M2:

```
// LOAD $MGBLD
// RUN
// MGBLD SOURCE-#MN#M2,SSP-YES,REPLACE-YES,LIBRARY-xxx
// END
```

  where xxx identifies the library containing MCSCU support.

There are two messages in message member #MN#M2. Message number 0001 allows you to define the conversion table for going from NLV to MCS, and message number 0002 is used to convert MCS to NLV. You must enter contiguous pairs of from-to values. Message 0001, in the following example, maps ¢ to <, $ to {, and # to }.

```
0001 ¢<${#}
```

To translate back, message 0002 would be

```
0002 <¢{$}#
```

#MN#M2 is a second-level message member and allows 112 pairs of from-values and to-values. The first blank from position in the message is the end of the table.

Note: For additional information on character conversion, see Appendix L, *System/34 Translation Tables*.

Source Statement Length

You can have different source statement lengths in your library that vary based on the statement type. When MCSCU copies the members from the library to the work file, all statements will have a length of 120. This may expand your library space requirements by 1 byte per source record. If MCSCU tried to determine statement length, MCSCU could possibly truncate your data.

MCSCU Audit List

As MCSCU examines the selected members in the work file, it prints a simple audit list of members that it has completed. Completed means having performed the requested options on the source member, not necessarily having modified any statements. The format for the audit line printed per member is:

```
                                        {0}        {0}        {0}
*** MEMBER NAME - XXXXXXXX, LIST-{1},COUNT-{1},MODIFY-{1},
    DATE-XX/XX/XX, TIME-YY:YY:YY
*** MEMBER NAME - XXXXXXXX, COMPLETE
```

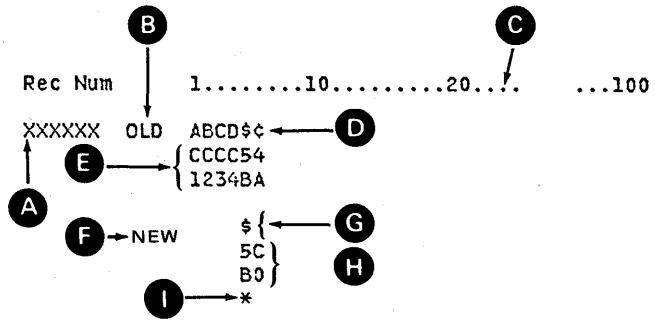
The member name with the options line is printed when the member is started. Sensitive records from the list option follow this record. This information is printed regardless of whether you selected the list option. The audit list is useful as a history of:

- Which members have been processed in the work file
- Which processing options were requested

In the case of recovery, you can determine members needing to be processed by displaying the file and eliminating those on the audit list (after they have been copied to the output library). If the word PARTIAL appears instead of COMPLETE, this indicates the work file was not large enough to contain the entire member when the member selection copy was performed. A message will be issued by $MAINT when the work file is copied back to the library. You should recopy the partial member to a new work file and reprocess the entire member.

LIBRARY STATEMENT LIST

If you want to obtain a list of all statements that have been detected as being sensitive to the MCS, this option can be used. The format of the list is as follows:



Ⓐ The relative record number of this statement within the member being processed.

Ⓑ OLD indicates that ABCDE$¢ is the original record value.

Ⓒ Record number and column counter constants to assist you in determining the column in which the sensitive character was found.

Ⓓ The original record printed showing graphics. A position may be blank if the printer does not support the graphic.

Ⓔ The hexadecimal value of each graphic.

Ⓕ NEW indicates these are the sensitive record positions. The sensitive characters have data printed below the original record. The characters printed on the NEW line flag where sensitive data was found. (In later discussions, this is how the term flag is used.)

Ⓖ The new sensitive graphic. If you selected MODIFY-NO, the original and new graphics will be equal. If you selected MODIFY-YES:

- If it is a record position that MCSCU was allowed to modify, the new substituted graphic is printed.
- If it is a record position that MCSCU was not allowed to modify (in the index, outside limits), the original character is printed.

Ⓗ The hexadecimal value associated with what printed on the NEW line.

Ⓘ An asterisk (*) is printed below each NEW line character that was sensitive but not modified.

LIBRARY STATEMENT COUNT

MCSCU counts how many statements in a member were sensitive to MCS. It also counts all of the statements in the entire work file that are sensitive to MCS. The sensitive record value is printed as an additional value on the audit list member complete line. It appears as:

*** MEMBER NAME-XXXXXXXX, COMPLETE, COUNT-XXXXXX

The count of sensitive records for the entire file is printed when an end-of-disk file is encountered. The format of the total count record is:

TOTAL SENSITIVE RECORD COUNT-XXXXXX

LIBRARY STATEMENT MODIFY

Based on the type of member being processed, MCSCU becomes sensitive to different types of specifications within a source member. Certain columns and fields can be changed and others can only be flagged. Each of the following sections describes the activity that is performed per specification type. If within a specific member, a statement is encountered that does not fit the criteria for that source type, sensitive data is flagged but not modified. For example, a Z in column 6 of RPG is invalid and the statement would be scanned but not modified.

Note: MCSCU assumes that the members to be processed are valid in syntax and would be accepted by the System/34 SSP or PP that would normally be using them.

RPG

If columns 1 and 2 contain two asterisks (**), then all statements for that member following the '**' record are scanned but not modified. Two asterisks indicate compile time table or array data, alternate sequence, and file translation table.

If column 7 contains a single asterisk (*), all columns are scanned and sensitive characters are modified.

If column 6 contains an H:

- Columns 1 through 74 are scanned and sensitive characters are modified.
- Column 18 has the floating currency symbol filled in from the conversion table if (1) column 18 is blank, (2) the conversion table contains a hex 5B in a from comparison value, and (3) you are converting from NLV to MCS. When converting from MCS to NLV, column 18 will be blanked. All NLV-to-MCS tables supplied with MCSCU have a hex 5B value when the to-value is other than hex 5B.
- Columns 75 through 80 are only flagged.

If column 6 contains an F:

- All columns are scanned.
- Data is only flagged.

If column 6 contains an I:

- Columns 27 through 41 are scanned and sensitive characters are modified.
- Other columns are only flagged.

If column 6 contains a C:

- All columns are scanned and flagged.
- Columns 18 through 27 and 33 through 42 are modified if column 18 or 33 contains a single quote (hex 7D) indicating a literal.

If column 6 contains a 0:

- All columns are scanned and flagged.
- Columns 45 through 80 are modified if columns 40 through 43 do not contain a K followed by a number (right justified) from 1 through 8, which denotes a format name.

If column 6 contains an L, E, T, or U, the records are only scanned and flagged.

If when starting in column 7, MCSCU detects a /COPY, /SPACE, /EJECT, or /TITLE, the statements are only flagged.

Procedures

All statements are flagged when an MCS sensitive character is detected. Only the following OCL statements are modified:

- Comment statements in a procedure are modified completely. A comment is denoted by an asterisk (*) in column 1.
- // *, // **, // MSG, and // PAUSE with text have the message text and any comments modified.
- For any statement not beginning with a // ъ, all columns are scanned and flagged.

Note: Because the work file is a record mode file, $MAINT has inserted a // COPY before each member in the file and a // CEND after each member. If you have a procedure that contains internal // CEND statements, the internal // CEND prematurely terminates the procedure processing for that member. It also may cause incorrect audit line printing. Internal // CEND statements are valid in user procedures only when FROM-READER is specified for $MAINT.

Message Members

All statements in a member are examined and processed in the following manner:

- All comments are scanned and modified.
- The control specification in the member is only flagged.
- All message definition statements are scanned and modified.

The assembler user should be aware that messages indicating substitution by # through SYSLOG will have their contents changed if # has an alternate MCS value from NLV.

Menu Members

Same as *Message Members*.

Note: If the command message member for a menu contains OCL, sensitive data is changed.

Screen Formats

All comments are scanned and sensitive characters are modified.

If column 6 contains an S or D, then:

- Columns 7 through 14 are flagged.
- All other columns are scanned, and sensitive characters are modified.
- Columns 7 through 14 of a continued D specification are modified.

Sort Members

All statements in the member are subject to modification except the ALTSEQ statement. MCSCU only flags sensitive characters in the ALTSEQ specifications.

Work Station Utility

If column 7 contains an asterisk (*), indicating a comment, all columns will be scanned and sensitive characters will be modified.

If column 6 contains a J, T, or M, all columns are scanned, and sensitive data is flagged.

If column 6 contains an S or D, then the statement is processed like SFGR (see *Screen Formats* in this appendix).

If column 6 contains a C, the statement is processed like the RPG C-specification described earlier. In addition, IMSG, MSG, and compare to a table operation codes that have sensitive data within single quotes will be modified.

Other Statement Type

If the source member is not uniquely identified by a previously described option, this option could be used. This option scans all statements in the member and modifies any MCS sensitive character that is detected when modify is selected.

LIBRARY MEMBER COPY BACK

When the conversion program is complete, the work file is copied back to the specified output library if modify was selected. It is your responsibility to ensure that the destination library is large enough to hold all converted source. The copy function used is $MAINT, and any halts that $MAINT can issue when doing a TOLIBR-like function can occur (see *Additional Library Member Considerations* in this appendix).

ADDITIONAL LIBRARY MEMBER CONSIDERATIONS

MCSCU copies library members to a work file, modifies the members in the work file, and then copies them to the specified output library.

Because some library member statements cannot be totally modified by MCSCU, some manual examination of the LIST option results should be performed. Any necessary manual changes should be made using SEU. Members that require recompilation/regeneration (such as RPG, COBOL, FORTRAN, assembler, DFU, message members, screen formats, WSU, and menu members) must be recompiled. When you are finished executing MCSCU against a work file, you should delete the work file using the DELETE procedure command.

If the output library becomes filled while the modified source is being copied back to the library from the work file, you can perform the following steps:

- Reallocate the output library to make it larger.
- Enter the following OCL statements, which will copy the work file back to the output library:

```
// LOAD $MAINT
// FILE NAME-xxx
// RUN
// COPY FROM-DISK,TO-yyy,FILE-xxx
// END
```

where xxx is the work file name, and yyy is the output library name.

DATA FILE CONVERSION

You have files that contain data relative to operating business. These data files contain information such as inventory item descriptions, customer names, addresses, and account descriptions. These descriptive data areas may contain characters that have taken on a new hexadecimal value under MCS. When a program displays or prints these data areas, the graphic associated with the MCS value instead of the graphic associated with the NLV value is displayed. MCSCU assists in finding data within records that are MCS sensitive. You should be careful to avoid having MCSCU modify data fields containing binary or packed values that may appear to MCSCU to be sensitive.

If you select data file conversion, the prompt screen shown in Figure M-4 is displayed. A description of each prompt follows the display.

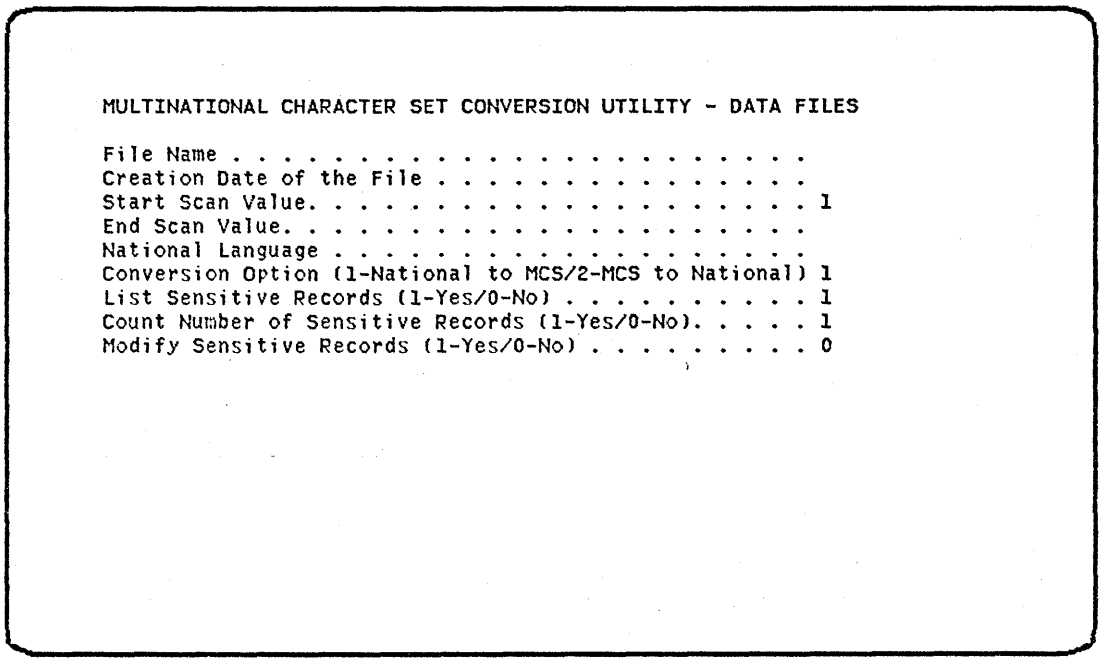


Figure M-4. Data File Conversion

File Name: Specify the name of the data file that is to be converted. This value is required. If an invalid name is entered, the FILE statement OCL processor issues an error.

Creation Date of the File: Enter the 6-digit creation date for this file if you need to distinguish between files with the same name.

Start Scan Value: If you want the record scanned and modified within limits, you can indicate start scan value. The value must be greater than 0 and less than or equal to the record length. The default value is 1.

End Scan Value: You can indicate the end value for the scan within limits. The value must be greater than or equal to the start scan value and less than or equal to the record length. The default is the last byte of the record.

National Language: Specify the language group (NLV) requested from the following list:

USA = United States and Canada
GERMANY = Austria/Germany
BELGIUM = Belgium
BRAZIL = Brazil
CANADA = Canada (French)
DENMARK = Denmark/Norway
FINLAND = Finland/Sweden
FRANCE = France
ITALY = Italy
JAPAN = Japan (English)
PORTUGAL = Portugal
SPAIN = Spain
SPANISH = Spanish-Speaking
UK = United Kingdom
OTHER = User-Supplied

This entry is required. MCSCU supplies the conversion tables for each of the NLVs. If you wish to supply your own tables, you can do so by completing the message in the MCSCU message member (#MN#M2) for this purpose. When OTHER is selected, the messages in #MN#M2 are used. See *Conversion Tables* in this appendix for further explanation.

Conversion Option: MCSCU allows conversion from National language (NLV) to MCS by option 1, or MCS to National language (NLV) by option 2. The default is option 1.

List Sensitive Records: MCSCU lists the sensitive/affected records within the file. Option 1 is list; option 0 is do not list. The default is 1. See *Data File List* later in this appendix for additional information.

Count Number of Sensitive Records: MCSCU counts the number of statements in the specified file that are sensitive. Option 1 is count; option 0 is do not count. The default is 1. See *Data File Record Count* later in this appendix for additional information.

Modify Sensitive Records: MCSCU either modifies the records as it scans them, or leaves them as is. Option 1 is modify; option 0 is do not modify. The default is 0. See *Data File Modify* later in this appendix for a further explanation of what modify means for files.

MCSCU automatically determines the type of file (indexed, consecutive, or direct) and performs modification accordingly.

As with library members, you can cause MCSCU to:

- List the MCS data sensitive records in the file
- Count the MCS sensitive records in the file
- Modify the MCS sensitive records in the file

You have the same options as you did for library member conversion in terms of:

- Determining language group to be converted
- Converting from National to MCS
- Converting from MCS to National
- Providing your own conversion table

Refer to *Library Member Modification* in this appendix for an explanation. The data conversion is an update-in-place operation. The file is not written to an intermediate file. It is recommended that you back up your current files before beginning the conversion. MCSCU automatically determines the type of file (consecutive, indexed, or direct) that was selected and other attributes about the file such as record length, key length, and key location. If you select no for list, count, and modify, a halt is issued. Information is passed to the data file conversion program via the local data area. The last 100 bytes of the local data area may be used. If scan limits are invalid, a halt is issued and MCSCU terminates.

The program that performs the actual work file analysis and modification is an EVOKED program. This allows the actual conversion program to execute as a separate job and allows control to return immediately to the MCSCONV procedure to prompt for additional conversion activity. Thus, it is possible to have several different work files being analyzed and modified at the same time. Because the analysis and modification of the work file is still in progress, no additional members should be copied to the work file, or the work file should not be deleted, until the EVOKED job has completed.

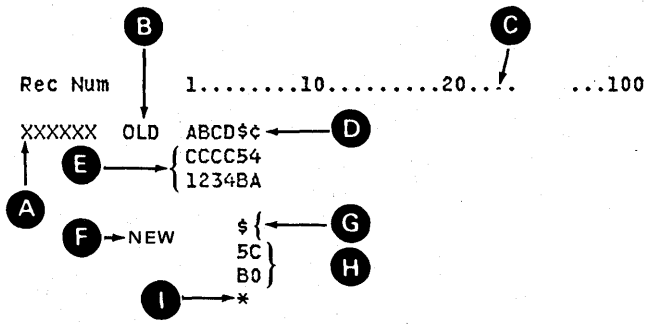
Regardless of the list option being selected, MCSCU prints an informational line about the data set as in the library member audit. The format of the print line is:

```
                                {0}        {0}        {0}
FILE NAME-XXXXXXXX, LIST-{1},COUNT-{1},MODIFY-{1}, START
  SCAN-XXXX, END SCAN-XXXX, DATE-XX/XX/XX, TIME-YY:YY:YY
FILE NAME-XXXXXXXX, COMPLETE
```

The file name with the options line is printed when the file is started. Sensitive records from the list option follow this record.

Data File List

If you want to obtain a list of all records that have been detected as being sensitive to MCS, this option can be used. For files with record lengths greater than 100, the following will be repeated until an entire sensitive record is printed. The format of the list is:



- **A** The relative record number of this statement within the file being processed.

- **B** OLD indicates that ABCDE$¢ is the original record value.

- **C** Record number and column counter constants to assist in determining the column in which the sensitive character was found.

- **D** The original record printed showing graphics. A position may be blank if the printer does not support the graphic.

- **E** The hexadecimal value of each graphic.

- **F** NEW indicates these are the sensitive record positions. The sensitive characters have data printed below the original record. The characters printed on the NEW line flag where sensitive data was found. (In later discussions, this is how the term flag is used.)

- **G** The new sensitive graphic. If you selected MODIFY-NO, the original and new graphic will be equal. If you selected MODIFY-YES:

  - If it is a record position that MCSCU was allowed to modify, the new substituted graphic is printed.

  - If it is a record position that MCSCU was not allowed to modify (in the index, outside limits), the original character is printed.

- **H** The hexadecimal value associated with what printed on the NEW line.

- **I** An asterisk (*) is printed below each NEW line character that was sensitive but not modified.

Data File Record Count

MCSCU counts the records in this file that have MCS sensitive data. This value is printed as an additional value on the audit list file complete line. It appears as follows:

*** FILE NAME-XXXXXXXX,COMPLETE,COUNT-XXXXXX

DATA FILE MODIFY

The modify option uses the specified conversion table; wherever it encounters an MCS sensitive character, it substitutes the alternate value. The only exception is that MCSCU does not modify the key field of an indexed file. It only flags MCS sensitive data in the key field. You should use the scan limit values to avoid having MCSCU accidentally process binary and packed fields that could contain what appear to be sensitive characters. The same is true for fields in direct files that have a hashing technique used against them to locate a record. If MCSCU changes the value in a field that is hashed, the program execution is changed.

If you need to convert indexed files:

- Copy the indexed file to a sequential file under a new file name.
- Execute MCSCU against the sequential file.
- Delete the original user file.
- Copy the sequential file back to a new file with the original file name and attributes (indexed, key placement, delete capable, extendable, and so on).

BATCH INTERFACE

If you want to set up a batch procedure to convert library members or files, you can accomplish this. You might want to do this for a multilocation installation and be able to have a programmer in location X establish procedures for locations X, Y, and Z.

Note: When you execute MCSCU via the batch interface, the programs that do the actual data file or work file analysis and modification are not EVOKED. There are also no parameter defaults from batch interface.

The operation control language necessary to do member conversion is:

- Build OCL statements to execute $MAINT and copy members to a disk file. The following OCL statements copy from library ABC all source members beginning with PAY to a file called PAYR with a record length of 120. Then, it adds members that begin with AR to file PAYR.

```
// LOAD $MAINT
// FILE NAME-PAYR,UNIT-F1,BLOCKS-100,RETAIN-T
// RUN
// COPY FROM-ABC,TO-DISK,FILE-PAYR,RECL-120,
// NAME-PAY.ALL,LIBRARY-S,SVATTR-YES
// COPY FROM-ABC,TO-DISK,FILE-PAYR,ADD-YES,
// NAME-AR.ALL,LIBRARY-S,SVATTR-YES
// END
```

- Set UPSI switch 1 on with the following OCL statement:

```
// SWITCH 10000000
```

  This causes all prompt displays to be suppressed.

- Invoke MCSCU internal procedure MCSLIB with parameters in the following order:

  MCSLIB memtype,outlib,lang,fromto,list,count,modify,filename

  memtype–The value 1 through 8 for member type described in *Library Member Modification* earlier in this appendix

  outlib–The name of the library that the file is to be copied back to

  lang–One of the valid national languages described in *Library Member Modification* earlier in this appendix

  fromto–1 for NLV to MCS, 2 for MCS to NLV

  list–1 for list sensitive records, 0 for do not list

  count–1 for count sensitive records, 0 for do not count

  modify–1 for modify sensitive records, 0 for do not modify

  filename–The name of the work file containing the members to be processed

The same halts that were issued for invalid parameter values are still issued.

Data Files

The operation control language necessary to do file conversion is:

- Set UPSI switch 1 on with the following OCL statement:

  // SWITCH 10000000

  This causes the prompt display to be suppressed.

- Invoke MCSCU internal procedure MCSDATA with parameters in the following order:

  MCSDATA filename,date,start,end,lang,fromto,list,count,modify.

  filename–The name of the file to be converted

  date–If multiple files with this name exist, the 6-digit date on which the file was created

  start–The 4-digit start scan value described in *Data File Conversion* earlier in this appendix

  end–The 4-digit end scan value described in *Data File Conversion* earlier in this appendix

  lang–One of the valid National languages described in *Data File Conversion* earlier in this appendix

  fromto–1 for NLV to MCS, 2 for MCS to NLV

  list–1 for list sensitive records, 0 for do not list

  count–1 for count sensitive records, 0 for do not count

  modify–1 for modify sensitive records, 0 for do not modify

Table M-1. Changed Characters by Language Group

The following table shows, by language group, the System/34 characters that changed with the Multinational Character Set. The column labeled old char. contains the original graphic (character). The column labeled old hex. contains the original hexadecimal representation that has to be changed to the value shown in the column labeled new hex. If this conversion is not done, the new character represented by the old hex value will be displayed or printed.

	Old Char.	Old Hex. (From)	New Hex. (To)	New Character Represented by the Old Hex (If Not Changed)
US & Canada	¢	4A	B0	[
	¦	4F	BB	!
	!	5A	4F	]
	¬	5F	BA	^
Austria/Germany	Ä	4A	63	[
	Ü	5A	FC	]
	ö	6A	CC	|
	§	7C	B5	@
	ß	A1	59	~
	ä	C0	43	{
	ü	D0	DC	}
	Ö	E0	EC	\
Belgium	ù	6A	DD	|
	à	7C	44	@
	¨	A1	BD	~
	é	C0	51	{
	è	D0	54	}
	ç	E0	48	\

	Old Char.	Old Hex. From)	New Hex. (To)	New Character Represented by the Old Hex (If Not Changed)
Brazil	É	4A	71	[
	$	5A	5B	]
	Ç	5B	68	$
	ç	6A	48	¦
	ã	79	46	`
	õ	7B	EF	#
	Ã	7C	66	@
	Õ	C0	CF	{
	é	D0	51	}
Canada(French)	à	4A	44	[
	'	5A	BE	]
	ù	6A	DD	¦
	¨	A1	BD	~
	é	C0	51	{
	è	D0	54	}
	↳	E0	9D	\
Denmark/Norway	#	4A	7B	[
	¤	5A	9F	]
	Å	5B	67	$
	ø	6A	70	¦
	Æ	7B	9E	#
	ø	7C	80	@
	ü	A1	DC	~
	æ	C0	9C	{
	å	D0	47	}

	Old Char.	Old Hex. (From)	New Hex. (To)	New Character Represented by the Old Hex (If Not Changed)
Finland/Sweden	§	4A	B5	[
	Ä	5A	9F	]
	Å	5B	67	$
	Ö	6A	CC	¦
	é	79	51	`
	ä	7B	63	#
	ö	7C	EC	@
	ü	A1	DC	~
	à	C0	43	{
	å	D0	47	}
	É	E0	71	\
France	°	4A	90	[
	§	5A	B5	]
	ù	6A	DD	¦
	£	7B	B1	#
	à	7C	44	@
	¨	A1	BD	~
	é	C0	51	{
	è	D0	54	}
	ç	E0	48	\

	Old Char.	Old Hex. (From)	New Hex. (To)	New Character Represented by the Old Hex (If Not Changed)
Italy	°	4A	90	[
	é	5A	51	]
	ò	6A	CD	¦
	ù	79	DD	`
	£	7B	B1	#
	§	7C	B5	@
	¿	A1	5B	~
	à	C0	44	{
	è	D0	54	}
	ç	E0	48	\
Japan (English)	£	4A	B1	[
	|	4F	BB	!
	!	5A	4F	]
	¥	5B	B2	$
	¬	5F	BA	^
	¯	A1	CA	~
	$	E0	5B	\
Portugal	Ç	4C	68	<
	õ	6A	CF	¦
	Ã	7B	66	#
	Õ	7C	EF	@
	ç	A1	48	~
	ã	C0	46	{
	´	D0	BE	}
	Ç	E0	68	\

	Old Char.	Old Hex. (From)	New Hex. (To)	New Character Represented by the Old Hex (If Not Changed)
Spain	|	4F	BB	!
	₧	5B	B3	$
	¬	5F	BA	^
	ñ	6A	49	¦
	Ñ	7B	69	#
	¨	A1	BD	~
Spanish-Speaking	|	4F	BB	!
	¬	5F	BA	^
	ñ	6A	49	¦
	Ñ	7B	69	#
	¨	A1	BD	~
United Kingdom	$	4A	5B	[
	|	4F	BB	!
	!	5A	4F	]
	£	5B	B1	$
	¬	5F	BA	^
	‾	A1	CA	~

Glossary of Terms and Abbreviations

This glossary of terms and abbreviations includes definitions developed by the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO). This material is reproduced from the American National Dictionary for Information Processing, copyright 1977 by the Computer and Business Equipment Manufacturers Association, copies of which may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York, 10018.

ANSI definitions are identified by an asterisk. An asterisk to the right of the term indicates that the entire entry is reprinted from the *American National Dictionary for Information Processing*. Where definitions from other sources are included in the entry, ANSI definitions are identified by an asterisk to the right of the item number.

access method: A method for moving data between main storage and input/output devices.

acquire: To assign a nonrequesting display station to a program.

active file: A disk or diskette file being used by a program.

active format 1: An SSP control block in main storage that represents a disk VTOC file label or a diskette header label. The SSP uses active format 1's to process files being used by programs currently running on the system.

active message member: A load module, containing informational or diagnostic statements specified by the MEMBER OCL statement, that is accessible through supplied system functions.

add operation: A disk or diskette operation that adds records to an existing file.

allocate: To assign a resource, such as a disk file or a diskette file, to perform a specific task.

alphabetic character: Any one of the letters A through Z, or the special characters #, $, and @.

alphameric (A/N): Consisting of both letters and numbers and often other symbols (such as punctuation marks and mathematical symbols). Contrast with *ideographic*.

alphameric character: A character that requires 1 byte of storage. Contrast with *ideographic character*.

alphameric field: A field that contains only alphamerics.

alternative sector: A sector on a disk or a diskette that is assigned by the system in place of a defective sector.

alternative sector cylinder: The area on a disk that contains sectors that can be assigned in place of defective sectors.

American National Standard Code for Information Interchange, X3.4-1968 (ASCII): *The standard code, using a coded character set consisting of seven-bit coded characters (eight bits including parity check), used for information interchange among data processing systems, communications systems, and associated equipment. The ASCII character set consists of control characters and graphic characters.

APAR: authorized program analysis report.

application program: A program that performs a particular data processing task; for example, inventory control or payroll.

APPLID (application identification): The name of a VTAM (virtual telecommunications access method) application as specified in the APPL VTAM definition statement.

ASCII: American National Standard Code for Information Interchange, X3.4-1968.

authorized program analysis report (APAR): A request for correction of a problem caused by a defect in a current release of a program.

autowriter: A function that causes the spool writer to be loaded without operator intervention whenever output exists in the spool file. See also *spool writer*.

backup copy: A copy of a file or a library member that is kept in case the original file or library member is destroyed.

backup diskette: A diskette that contains information that was copied from another diskette or disk. A backup diskette is used if the original information is unintentionally altered or destroyed.

BASIC: Beginners' all-purpose symbolic instruction code; an easy-to-use interactive programming language.

basic data exchange: A file format for exchanging data on diskettes between systems or devices. Basic data exchange refers to diskette files only, not entire diskettes. For diskette 1 diskettes, use 128-byte format. For diskette 2D diskettes, use 256-byte format.

basic ideographic character set: An ideographic character set defined by IBM that contains 3707 characters consisting of 3226 Kanji characters and 481 additional characters, which include Katakana, Hiragana, the alphabet (A through Z and a through z), numerics (0 through 9), roman numerals (I through X), Greek, Russian, and special symbols. The basic ideographic character set is defined in hardware for each ideographic-capable printer and display station.

BCD: Binary-coded decimal.

BCD character code: A set of 64 six-bit characters. Contrast with *EBCDIC*.

binary: (1) Relating to, being, or belonging to a system of numbers having 2 as its base; for example, the binary digits 0 and 1. (2) Involving a choice or condition of two alternatives, such as on-off or yes-no.

binary synchronous communications (BSC): A flexible form of line control that provides a set of rules for transferring data over a communications line connecting two or more devices that use a communications adapter.

binary synchronous communications equivalence link (BSCEL): The SSP-ICF subsystem that provides a BSC interface to another System/34 and many other BSC systems. All remote systems are treated identically. Any remote system considerations must be handled by the application program.

bit: A binary digit.

bits per second (bps): The rate at which a device transmits or receives binary information; or, the rate at which a recording head reads or writes data.

block: (1) A record or a collection of contiguous records recorded or processed as a unit. (2) In System/34, a 10-sector unit of disk space.

bps: Bits per second.

BSC: Binary synchronous communications.

BSCEL: Binary synchronous communications equivalence link.

byte: (1) The representation of a character. (2) A sequence of 8 adjacent bits that are operated on as a unit and that constitute the smallest addressable unit in System/34. (3) The representation of a character by 8 bits; the amount of storage required for one EBCDIC character. See *extended binary-coded decimal interchange code (EBCDIC)*.

cancel: To end the current job before the job is completed. See also *controlled cancel, immediate cancel.*

CCP: Communications Control Program.

CE: Customer engineer.

CE panel: A panel containing indicator lights and switches that the CE uses during system maintenance.

CE track: An area on disk that is used as a read/write area for CE diagnostics.

character: *A digit, letter, or other symbol that is used as part of the control, organization, or representation of data.

character generator utility: A part of the Ideographic Generator/Sort Program Product that is used to create, maintain, and display ideographic characters.

character set: A group of characters used for a specific purpose; for example, the set of characters a printer can print.

checkpoint: A reference point in a program at which information about the contents of main storage can be recorded so that, if necessary, the program can be restarted at an intermediate point.

checkpoint active file: A fixed disk file that is being used by either a task that is currently being checkpointed or a checkpointed task that failed and has not yet been restarted.

checkpoint active library: A library that is being used by either a task that is currently being checkpointed or a checkpointed task that failed and has not yet been restarted.

checkpoint record file: A disk file containing a collection of checkpoint records.

checkpoint records: Records that contain the status of a job and the system at the time the records are written by the checkpoint facility. These records provide the information necessary for restarting a job without having to return to the beginning of the job.

checkpoint restart: The process of resuming a job at a checkpoint within the job step that caused an abnormal termination.

checkpoint/restart facility: A facility for restarting the execution of a program at some point other than the beginning, after the program was terminated due to a program or system failure. The restart begins at a checkpoint and uses checkpoint records to reinitialize the job.

CICS: Customer Information Control System.

close: The SSP actions taken when processing of a file is complete and the program that processed the file terminates.

COBOL: Common Business Oriented Language. A standardized, English-like high-level computer language.

command: A request for the performance of an operation or the execution of a particular program. See also *control command, procedure command.*

command display station: A display station that was defined during system configuration as being capable of requesting and initiating jobs, as well as being acquirable by an executing program. See also *data display station.*

Communications Control Program: A feature of the IBM System/3 CCP that provides the control program services needed to operate a communications based information processing system.

compile: *To prepare a machine language program from a computer program written in another programming language by making use of the overall logic structure of the program, or generating more than one machine instruction for each symbolic statement, or both, as well as performing the function of an assembler.

compiler: (1) A program that translates a series of instructions, written in a programming language, into a program the system can execute. (2) *A program that translates a source program written in a specific programming language into an object program.

compress: (1) To use the COMPRESS procedure or the $FREE or $PACK utility program to move files together on disk to create one continuous area of unused space. (2) To use the CONDENSE procedure or the $MAINT utility program to move library members together in order to create one continuous area of free space within a library.

compression: A technique for removing strings of duplicate characters and truncating trailing blanks prior to transmitting data.

concurrent processing: A method of processing in which two or more jobs appear to be processing at the same time. The instructions of each job are processed one at a time, but alternate in such a fashion as to make the most efficient use of the system.

configuration: The group of machines, devices, and programs that make up a data processing system. See also *system configuration.*

configuration record: See *system configuration record, display station configuration record.*

configure: To communicate information (to the control program) about the devices and optional features installed on a system.

control command: A command statement used by an operator to control system or display station operation. A control command does not run a procedure and cannot be used in a procedure. See also *command, procedure command.*

control statement: A statement that provides the SSP or utility program information about the job being run. See also *utility control statement.*

controlled cancel: The system action that usually occurs when option 2 is selected in response to a formatted message display, or when option 2 is selected in conjunction with a CANCEL control command. The job step being executed is ended. Any new data already created is preserved. The job that is executing can continue with the next job step. Compare with *immediate cancel.*

copy: To read data from a source, leaving the source data unchanged, and to write the same data elsewhere in a physical form that may differ from that of the source; for example, to copy main storage to disk.

creation date: The program date at the time a file is created. The creation date is stored as an attribute in the VTOC entry for the file. See also *program date, session date, system date.*

cursor: A movable character (underscore) on a display screen that indicates where the next character keyed by the operator is to appear.

Customer Information Control System: A transaction-oriented, multiapplication data base/data communications interface between a System/370 and user-written application programs. It provides many of the facilities necessary for standard terminal applications, such as message switching, inquiry, data collection, order entry, and conversational data entry.

cylinder: All disk or diskette tracks that can be accessed without repositioning the disk drive or diskette drive access mechanism.

data display station: A display station that was defined during system configuration as only being acquirable by an executing program. A data display station cannot request or initiate jobs. See also *command display station.*

define the file (DTF): An area in the user program containing information that is used as the primary interface between data management routines and users of the data management routines.

delete: To remove a unit of data; for example, a character, field, record, or file. A deleted file is one that has been removed from the volume table of contents (VTOC).

delete-capable file: A file that can contain records that are logically deleted, through no physical compression occurred when the records were deleted.

directory: Same as *library directory.*

disk file: An organized collection of related records on disk that are treated as a unit.

diskette: A thin, flexible, magnetic disk permanently enclosed in a semi-rigid protective jacket.

diskette drive: The mechanism that reads and writes diskettes.

diskette 1: Refers to a 33FD diskette. Contrast with *diskette 2D.*

diskette 2D: Refers to a 53FD diskette. A diskette that can contain data on both sides, with two times the number of bytes being stored in the same physical space as diskette 1. See also *diskette 1.*

display screen: The part of a display station on which data, messages, or other information is displayed.

display screen format: A two-part table that defines a display presented by display station data management. Display screen formats are generated and placed in a library load member by the $SFGR utility program.

display station: An input/output device that contains a display screen on which data is displayed, and an attached keyboard from which data is entered. It can be used to request jobs and/or enter data. A display station can be designated as the system console or as a command or data display station at system configuration time.

display station configuration record: An area on disk that describes a command display station's environment. The display station configuration record contains information such as the session date, the work station ID of the printer to be used for the display station's output, and the region size for jobs submitted from the display station.

display station local data area: A 256-byte area on disk that can be used to pass information between jobs and job steps during a session. A separate local data area exists for each command display station.

DTF: Define the file.

dump: (1) To copy the contents of all or part of storage, usually to an output device. (2) Data that has been dumped.

EBCDIC: Extended binary-coded decimal interchange code.

EDF: Extendable disk file.

expiration date: The date after which a disk file or diskette file is no longer protected from automatic deletion by the system. See also *permanent file.*

extendable disk file (EDF): A feature of the SSP that provides the capability to dynamically increase the size of shared and nonshared disk files.

extended binary-coded decimal interchange code (EBCDIC): A character set of 256 eight-bit characters.

extended character file: An area on disk that contains IBM-supplied extended characters and can contain user-defined characters (see *extended ideographic character set*).

extended ideographic character set: An ideographic character set that contains 3483 IBM-supplied ideographic characters, and up to 4370 user-defined ideographic characters. The extended ideographic character set is stored on disk (see *extended character file*).

external indicators: Eight indicators that are normally set by the SWITCH OCL statement prior to processing. The indicators can be altered by the job during execution. External indicators can be used to specify which files are to be used in multifile processing. A set of external indicators exists for each command display station.

file: An organized collection of related records treated as a unit.

file name: An arbitrary symbol created by the programmer or program to identify and refer to a collection of related records. See also *label.*

first-level message member: A message member in which each record contains up to 75 bytes of text. See *message load member, message source member.*

first-level procedure: See *procedure level.*

format 1: An area in the disk VTOC that contains information about a data file; for example, the address and length of the file.

formatted diskette: A diskette on which track and sector control information has been written but which may or may not contain any data.

formatted message: A two-line display having a constant format. The first line (format line) provides information about the message, and the second line (message text line) contains the message itself.

FORTRAN: Formula translation; a high-level scientific programming language.

hexadecimal: Pertains to a number system with a base of 16; valid digits range from 0 (zero) through F (15).

history file: An area on disk where a log of specified types of system actions and operator responses is recorded.

I exchange: A file format for exchanging data on diskettes between systems or devices that support diskette exchange type I. I exchange refers to diskette files only, not entire diskettes.

ID: Identification.

ideographic: Consisting of both pictograms and graphics and often other types of symbols.

ideographic character: A pictogram or graphic that requires 2 bytes of storage. Contrast with *alphameric character.*

ideographic character set: A character set that contains pictograms or graphics that can be used to represent ideas.

ideographic field: In a record, one or more ideographic characters of related information bracketed by S/O and S/I control characters.

Ideographic Generator/Sort Program Product: The program product (program number 5726-IG1) that consists of the character generator utility and the ideographic sort utility.

ideographic sort utility: A part of the Ideographic Generator/Sort Program Product that is used to (1) arrange records (or their relative record numbers) in a predetermined sequence according to data contained in one or more specific fields within the records, and (2) rebuild the tables that contain the predetermined sequence if the user wants to alter the predetermined sequence.

ideographic support: The combinations of hardware and software elements that allow the use of ideographic data on System/34.

IF expression: Expressions within a procedure that are used to test for a true or false condition. The action performed by the procedure depends upon the results of the test (true or false). IF and IFT test for true, IFF tests for false.

IFILE characteristic: A disk file characteristic that provides shared index sequential access to records added randomly by key.

immediate cancel: The system action that occurs when option 3 is selected in response to a formatted message, or when option 3 is selected in conjunction with a CANCEL control command. When option 3 is specified, the job being executed is terminated. Any new data created by a previous step in this job is preserved; however, any new data created by this job step is lost. Compare with *controlled cancel.*

IMS/VS: Information Management System/Virtual Storage.

Information Management System/Virtual Storage: A System/370 program product that assists the computer user in implementing teleprocessing and batch-type data processing applications. Its features facilitate implementation, change, and expansion of applications that examine and maintain large centralized information files.

initial program load (IPL): A sequence of events that loads the system programs and prepares the system for execution of jobs.

initialize: To use the $INIT utility program to prepare (format) a diskette for initial use.

input job queue: A list of jobs waiting to be processed by the system. The list is maintained on the disk. Each entry in the list references a procedure stored in a library on the disk.

inquiry: (1) A request (entered from a display station) for information in storage. (2) A request for information that puts the system into inquiry mode. (The operator initiates an inquiry by pressing the Attn key.)

inquiry mode: The mode of operation when the system is responding to an inquiry request. (The operator puts the system in inquiry mode by pressing the Attn key.)

Interactive Communications Feature (SSP-ICF): A feature of the SSP that includes interactive support for BSC and SNA communications as well as communications between programs within the system.

intra: A SSP-ICF subsystem that enables user programs to communicate with other programs on the same System/34 without the use of communications lines.

IPL: Initial program load.

job: One or more related procedures or programs grouped into a first-level procedure. See *procedure level.*

job file: A disk file created with a retain parameter of J. A job file can be used by all the job steps in a job. The job file is defined only within the job and does not exist after the job ends.

job region: The amount of main storage ensured by the SSP for use by a job. The job region is specified by the REGION OCL statement, the SET procedure, or the $SETCF utility program.

job step: A unit of work represented by a single program. The LOAD OCL statement, the RUN OCL statement, and other OCL and utility control statements define the job step within a procedure.

job step date: Same as *program date.*

Katakana: A native Japanese character set that is used to represent the different Japanese sounds.

keyword: (1) A symbol that identifies a parameter. (2) A nonvariable character string (such as NAME) in a statement.

keyword parameter: A parameter that consists of a keyword, followed by one or more values.

label: The name in the disk or diskette volume table of contents that identifies a file.

library: An area on disk that can contain load members, procedure members, source members, and subroutine members. See also *system library*.

library directory: A variable-sized area that contains information about each member in the library; for example, the member name and the location.

library member: A named collection of records or statements in a library. See *load member, procedure member, source member, subroutine member*.

load member: A collection of instructions that the system can execute to perform a particular function, regardless of whether the function is requested by the operator or specified in an OCL statement. Load members can also contain display screen formats and message members. Load members are stored in a library.

main storage: General-purpose storage of a computer.

MCS: Multinational character set.

MCSCU: Multinational character set conversion utility.

member: Same as *library member*.

menu: A displayed list of items (usually jobs) from which the operator makes a selection.

message identification code (MIC): A 4-digit decimal number that identifies a record in a message member.

message load member: A special type of library load member from which the SSP retrieves the text associated with a specified message identification code.

message source member: A type of library source member that contains control and message-text statements (MIC and text) required to create a message load member.

message-text statements: Statements in a message source member that specify the message identification code and the text associated with that code.

MIC: Message identification code.

MLCA: Multiline communication adapter.

modulus 10 checking/modulus 11 checking: Formulas that calculate the self-check digit for a self-check field. See *self-check field*.

MRJE: MULTI-LEAVING remote job entry.

MRT: Multiple requestor terminal.

MULTI-LEAVING Remote Job Entry (MRJE): An SSP function that allows the user to communicate with a system over a communications line using BSC.

multiple-program mode: A mode of operation during which more than one job is processing concurrently.

multiple requestor terminal (MRT) program: A program that can process requests from more than one requesting display station concurrently. Compare with *single requestor terminal (SRT) program*.

multipoint: In data communications, pertains to a network configuration in which interconnected stations communicate with each other on a time-shared basis.

multivolume file: A diskette file that resides on more than one disk.

NEP: Never-ending program.

nested procedure: A procedure that is called by another procedure. See also *procedure level*.

network: In data communications, a configuration by which two or more stations can communicate with a computer.

never-ending program (NEP): A program that will own system resources for a long period of time, and defined as an NEP (NEP-YES) on the COMPILE OCL statement or the ATTR OCL statement.

NLV: National language version.

OCL: Operation control language.

offline multivolume file: A multivolume file that a system processes in segments. Each segment is processed before the next segment is copied to or from the disk.

operation control language (OCL): A programming language used to identify a job and its processing requirements to the System Support Program Product (SSP).

parameter: (1) A variable that is assigned a particular value for a specific purpose or process. (2) A value that is specified in a command statement or a control statement.

permanent file: A file that remains in existence until deleted by using the $DELET utility program. A permanent file is created with a retain parameter of P for disk or 999 for diskette.

positional parameters: Parameters that must appear in a designated sequence.

print belt: A belt containing the characters that the IBM 5211/3262 Printer can print.

print image: The character set loaded into storage that corresponds to the characters on the print belt being used.

print intercept: The routine that causes printer output to be placed in a spool file in disk storage rather than going to the printer. See also *spool writer* and *spooling.*

procedure: A set of related OCL statements, and possibly utility control statements, that cause a specific function or set of functions to be performed. A procedure in a library is called a procedure member.

procedure command: A command statement that runs a procedure. A procedure command is a special form of the INCLUDE OCL statement. See also *command, control command.*

procedure level: An indication of the order in which nested procedures are called. For example, if procedure A calls procedure B, and procedure B in turn calls procedure C, then procedure C is a third-level procedure. See *nested procedure.*

procedure member: A procedure that is stored in a library.

profile: A set of data that portrays the significant features of a user, program, or device.

program: A sequence of instructions to a computer that are written in a special form the computer can interpret. A program tells the system where to get input, how to process it, and where to put the results.

program date: The date associated with a program (job step). The program date is specified by a DATE OCL statement or the DATE procedure used between the LOAD and RUN OCL statements for the program. If a program date is not specified, the program date is the same as the session date. See also *creation date, session date, system date.*

program product: An IBM-written, licensed program for which a monthly charge is made. A program product performs functions related to processing user data.

program temporary fix (PTF): A temporary solution or bypass of a problem diagnosed by IBM as the result of a defect in a current release of a program. See *authorized program analysis report (APAR).*

PTF: Program temporary fix.

record: (1) A collection of related data that is treated as a unit. For example, one line of an invoice could constitute a record. A complete set of records could form a file. (2) To store data on a reusable input/output medium, such as a disk, diskette, or punched cards.

record mode: A method of operation in which data is transferred by the SSP one record at a time.

region: The amount of main storage available for a task. See also *job region, step region.*

relative record number: In a file, the location of a record in relation to the beginning of the file.

RPG II: A commercially oriented programming language specifically designed for writing application programs that meet common business data processing requirements.

scratch file: A scratch file does not exist after the job step that uses it ends. A scratch file is created with a RETAIN parameter of S for disk or 0 for diskette. You can change a temporary file to a scratch file by specifying a RETAIN parameter of S for disk or 0 for diskette.

SDLC: Synchronous data link control.

sector: (1) An area on a disk or diskette track reserved to record a unit of data. (2) The smallest amount of data that can be transferred to or from a disk or diskette by a single data transfer operation.

sector mode: A method of operation in which data is transferred by the SSP in sector multiples.

self-check digit: The rightmost digit of a self-check field. See *self-check field.*

self-check field: A field, such as an account number, that consists of a base number and a self-check digit. For data entry applications, the operator-entered self-check digit is compared to the self-check digit computed by the system. If the operator makes a mistake when entering (keying) a self-check field, an error message is displayed and the entire field can then be rekeyed.

session: The period of time during which programs or devices can communicate with each other; the elapsed time that starts when an operator signs on the system and ends when the operator signs off the system.

session date: The date associated with a session. The session date is specified by a DATE OCL statement or DATE procedure used before the first program is run from the display station, or by the SET procedure or the $SETCF utility program. If the session date is not specified, the session date is the same as the system date. See also *creation date, program date, system date.*

SEU: Source entry utility.

shift-in (S/I) control character: A character that indicates the end of a string of ideographic characters. The shift-in control character is represented by hex 0F. Contrast with *shift-out (S/O) control character.*

shift-out (S/O) control character: A character that indicates the start of a string of ideographic characters. The shift-out control character is represented by hex 0E. Contrast with *shift-in (S/I) control character.*

single-program mode: A mode of operation during which one job is completely processed before another job begins. Contrast with *multiple-program mode.*

single requestor terminal (SRT) program: A program that can have only one requesting display station at a time. Contrast with *multiple-requestor terminal (MRT) program.*

SNA: System network architecture.

SNA Upline Facility: An SSP-Interactive Communications Feature subsystem that provides the application programmer with the capability of using the interactive communications functions in an SNA environment.

SNUF: SNA Upline Facility.

sort program: A portion of the Utilities Program Product that arranges records in a predetermined sequence, according to data contained in one or more specific fields within the records.

source entry utility (SEU): A portion of the Utilities Program Product that the operator uses to enter and update procedures and source programs in a library.

source member: A collection of records (such as RPG II specifications or sort sequence specifications) that are used as input for a program. Source members are stored in a library.

spool file: An area on disk where spooled printer output is stored while waiting to be printed.

spool writer: A program that causes printer output, which has been stored in the spool file, to be printed. See also *print intercept* and *spooling.*

spooling: A part of the SSP that provides temporary storage of print data on disk. See also *print intercept* and *spool writer.*

SRT: Single requestor terminal.

SSCP: System services control point.

SSP: System Support Program Product.

SSP utility program: An SSP control program used by programmers in their daily system operations. For example, SSP utility programs can be used to copy files or initialize diskettes.

SSP-ICF: System Support Program Interactive Communications Feature.

SSP-ICF queue space: An area used by the SSP-ICF BSC subsystems to keep control blocks and buffers.

SSP-ICF session: A logical information route between a System/34 application and a remote subsystem.

SSP-Interactive Communications Feature (SSP-ICF): See *Interactive Communications Feature.*

standby mode: A mode of operation in which a display station is waiting to be acquired and used by a program running on the system.

step region: The maximum amount of main storage that can be used by a program that dynamically requests storage beyond its load member size.

subroutine member: A subroutine that needs to be link edited (joined) before being loaded for execution. Subroutine members are stored in a library.

subsystem: An IBM supplied SSP-Interactive Communications Feature module that runs as a System/34 task and provides specific data management, and, if applicable, link-level functions. All functions provided by the SSP-ICF for the System/34 user require one of the defined subsystems.

swapping: Temporarily removing an active job from main storage, saving it on disk, and processing another job in the area formally occupied by the first job.

system configuration: A process that specifies the various components and devices that form a particular operating system. System configuration combines user-specified options and parameters with IBM programs to produce a system having the desired form and capacity.

system configuration record: Information stored on disk that describes system characteristics and programming support; for example, system date format, disk capacity, and main storage capacity.

system console: A display station designated to activate certain system functions, and to control and monitor system operation, in addition to functioning as a display station. Contrast with *display station.*

system date: The date assigned by the system operator during the initial program load procedure. Generally, the system date is the same as the actual date. See also *creation date, program date, session date.*

system library: The library that contains the members that are part of the SSP. The system library is labeled #LIBRARY and cannot be deleted from disk. See also *library, user library.*

system log device: A device or devices designated by the LOG OCL statement to record messages and OCL statements.

system printer: The printer, designated at system configuration time, that is used for system and display station printed output, unless the output is specifically directed to another printer.

task: A unit of work for the main storage processor; therefore, the basic multiprogramming unit under the control program.

temporary file: A file that cannot be automatically deleted until after its expiration date. A temporary file is created with a retain parameter of T for disk or 001 through 998 for diskette.

transparent text mode: In data communications, a mode of binary synchronous transmission in which only transmission control characters preceded by the DLE control character are processed as line control characters. All other characters having the same bit pattern as transmission control characters are transmitted as data.

user area: (1) The area of main storage that is not used by the SSP. (2) The area on disk that is available to the user.

user library: A library created by the user. A user library is in addition to the system library and may contain any type of library member.

Utilities Program Product: A multipurpose program product for creating, maintaining, listing, and sorting files, and for creating, maintaining, and listing source and procedure members in a library.

utility control statement: A control statement that provides a utility program with information concerning the way the program is to perform its function or the output it is to produce.

utility program: See *SSP utility program.*

virtual telecommunications access method (VTAM): A set of programs that control communications between work stations and application programs operating under DOS/VS, OS/VS1, and OS/VS2.

volume table of contents (VTOC): An area on a disk or diskette that describes the location, size, and other characteristics of each file on the disk or diskette.

VTAM: Virtual telecommunications access method.

VTOC: Volume table of contents.

work station: A device that lets a person transmit information to or receive information from a computer, or both, as needed to perform his job; for example, a display station or printer.

X.21: A Consultative Committee on International Telephone and Telegraph (CCITT) recommendation that defines the physical, electrical and logical connection that provides digital circuit-switched and leased-circuit services between data terminal equipment and synchronous public data networks.

128-byte format: A format for diskette 1 diskettes with 128 bytes per sector and 26 sectors per track.

256-byte format: A format for diskette 2D diskettes with 256 bytes per sector and 26 sectors per track.

512-byte format: A format for diskette 1 diskettes with 512 bytes per sector and 8 sectors per track.

1,024-byte format: A format for diskette 2D diskettes with 1,024 bytes per sector and 8 sectors per track.

Index

//* (message) statement 1-77
//** (message) statement 1-78
/* (end of data) statement 1-76
$ARSP (auto response utility program) 4-4
$BACK (backup library utility program) 4-9
$BICR (basic data exchange utility program) 4-10
$BMENU (build menu utility program) 4-13
  input to 4-13
  output from 4-13
$BUILD (alternative sector rebuild utility program) 4-18
$COPY (disk copy/display utility program) 4-20
$DDST (keysort utility program) 4-37
$DELET (file delete utility program) 4-38
$DUPRD (diskette copy utility program) 4-48
  work space for 4-48
$FBLD (file build utility program) 4-52
$FREE (disk reorganization utility program) 4-55
  errors during 4-55
$HELP (help utility program) 4-57
$HIST (history file display utility program) 4-57
$HSML (history file scroll utility program) 4-62
$IDSET (Switched Line Remote ID Specification Utility Program 4-70
$IEDS (Subsystem Termination Utility Program) 4-72
$IENBL (Subsystem Initialization Utility Program) 4-73
$INIT (diskette labeling and initialization utility program) 4-75
$LABEL (VTOC display utility program) 4-81
  messages from J-2
$LOADI (reload library utility program) 4-93
$MAINT (library maintenance utility program) 4-94
$MGBLD (create message member utility program) 4-119
$MMSP (stop monitoring line utility program) 4-125
$MMST (start monitoring line utility program) 4-126
$PACK (disk reorganization utility program) 4-127
$PDSR (primary SDLC retry count reset utility program) 4-128
$PNLM (phone list utility) 4-130
$POST (data exchange utility program) 4-133
$PRES (resource security program) 4-136
$PRLT (security report utility) 4-136
$PRMN (menu security utility program) 4-137
$PROF (password security utility program) 4-138
$PRON (resource owner utilty program) 4-139
$PRST (security file restore utility program) 4-140
$PRSV (security file save utility program) 4-141
$RSTRT (program restart utility program) 4-143
$SETCF (set configuration utility program) 4-144
$SFGR (screen format generator utility program) 4-157
  messages from J-3
$SLFL (SETFILE utility program) 4-192
$UASC (CRT display of spool file entries) 4-193
$UASF 4-194
$XNLM (X.21 phone number list utility program) 4-197
$XSAVE (extended character file save utility) 4-202
* (comment) statement 1-76
#LIBRARY (see system library)

acquiring display stations 1-74
active library
  during inquiry request 1-48
  specifying with $SETCF utility 4-146
  specifying with LIBRARY statement 1-46
  specifying with SET procedure 2-171
  substituting 5-18
active message member
  definition J-1
  for a procedure 1-51
  for a session 1-51
  types of 1-51
  using $SFGR to display messages from 4-183

adding library members to a file
using $MAINT utility 4-95
using FROMLIBR procedure 2-81
addressing characters
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
aligning forms 1-60
ALTERBSC procedure 2-3
alternative sector rebuild utility program ($BUILD) 4-18
alternative sectors
definition N-1
ALTERSDL procedure 2-6
answer tone
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
APAR file
listing with LISTFILE procedure 2-119
APAR service procedure D-3
APPLYPTF service procedure D-5
ASCII codes H-2
ASSIGN command
system operator 3-38
ATTR statement 1-6
attributes of library members 4-98
auto response utility program ($ARSP) 4-4
autocall (see COMM, DEFINEPN, SESSION)

backing up a user library 2-170
backing up the system library 2-8, 4-9
backup file
listing with LISTFILE procedure 2-119
backup library utility program ($BACK) 4-9
BACKUP procedure 2-8
basic data exchange files
copying to/from disk
using $BICR utility 4-10
using JOBSTR procedure 2-110
using TRANSFER procedure 2-185
listing with LISTFILE procedure 2-119
basic data exchange utility program ($BICR) 4-10
belt image
changing with $SETCF utility 4-146
changing with IMAGE statement 1-37
changing with SET procedure 2-171
binary synchronous communications (see BSC)
blank compression
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
BLDFILE procedure 2-11
BLDLIBR procedure 2-15
BLDMENU procedure 2-19
block, record conversion A-1
block, sector conversion A-1
braces in statement formats xi
brackets in statement formats xi
BSC
altering configuration
with $SETCF utility 4-144
with ALTERBSC procedure 2-3
BSC CCP subsystem
specifying with SESSION statement 1-69, 1-69
BSC CICS subsystem
specifying with SESSION statement 1-69
BSC IMS subsystem
specifying with SESSION statement 1-69
BSCEL subsystem
specifying with SESSION statement 1-69
finance subsystem
specifying with SESSION statement 1-69
intra subsystem
specifying with SESSION statement 1-69
line monitoring
start using STARTM procedure 2-178
stop using STOPM procedure 2-179
line type
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
overriding configuration values
with $SETCF utility 4-144
with OVERRIDE procedure 2-132
peer subsystem
specifying with SESSION statement 1-69
SNA Upline Facility subsystem
specifying with SESSION statement 1-69
trace D-16
build menu utility program ($BMENU) 4-13
BUILD procedure 2-34
building a disk file
using $FBLD utility 4-52
using BLDFILE procedure 2-11
building a library
using $MAINT utility 4-94
using BLDLIBR procedure 2-15
building a menu
using $BMENU utility 4-13
example 4-18
using BLDMENU procedure 2-19
example 2-30

CANCEL command
display station operator 3-3
subconsole operator 3-21
system operator 3-40
CANCEL keyword 5-36
CANCEL statement 5-40
canceling jobs
on input job queue 3-40
on print queue 3-40
on system 3-40
CATALOG procedure 2-35
messages from J-2
CHANGE command
subconsole operator 3-23
system operator 3-42
characters per inch (horizontal)
assigning
using FORMS statement 1-35
using LINES procedure 2-117
using PRINTER statement 1-61
characters, System/34 F-1
checkpoint restart
$RSTRT utility 4-143
CRESTART procedure 2-54
clocking
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
COMM statement 1-10
command keys 4-163
codes returned from PROMPT screen 5-16
commas in statement formats xii
comment statement
in message source members 4-120
comments
on OCL statements 1-4
on utility control statements 4-9
common queue space (see SSP Interactive Communications Feature)
communicating between programs
using the local data area 1-48
communications lines
assigning 1-10
displaying status of 3-17, 3-57
dropping/holding the connection 3-14
monitoring
start using STARTM procedure 2-178
stop using STOPM procedure 2-179
COMPILE statement 1-12
compiling a program 1-12
COMPRESS procedure 2-37
compressing a library
using $MAINT utility 4-101
using CONDENSE procedure 2-38
compressing the disk
using $FREE utility 4-55
using $PACK utility 4-127
using COMPRESS procedure 2-37
compression of data
definition N-3
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
computing self-check digits 6-13
CONDENSE procedure 2-38
conditional expressions 5-23
CONSOLE command
display station operator 3-6
continuation
of OCL statements 1-3
of utility control statements 4-2
control commands
display station operator
CANCEL 3-3
CHANGE 3-4
CONSOLE 3-6
HOLD 3-7
IDELETE 3-8
JOBQ 3-9
MENU 3-10
MODE 3-11
MSG 3-12
OFF 3-14
PRTY 3-15
RELEASE 3-16
STATUS 3-17
TIME 3-20
subconsole operator
CANCEL 3-21
CHANGE 3-23
HOLD 3-27
MSG 3-28
RELEASE 3-29
REPLY 3-30
RESTART 3-31
START 3-32
STATUS 3-33
STOP 3-36
TIME 3-37
system operator
ASSIGN 3-38
CANCEL 3-40
CHANGE 3-42
HOLD 3-46
IDELETE 3-47
MSG 3-48
PRTY 3-49
RELEASE 3-50
REPLY 3-52
RESTART 3-53
START 3-54
STATUS 3-57
STOP 3-60
TIME 3-63
VARY 3-64
conversion
disk block address to first sector in block A-2

conversion (continued)
disk sector address to block address A-2
records to blocks
direct file A-1
indexed file A-1
sequential file A-1
copies
changing with CHANGE command 3-23, 3-42
specifying with PRINTER statement 1-59
COPYIN FILE statement for $COPY 4-31
copying basic data exchange diskette files
using $POST utility program 4-133
using POST procedure 2-136
copying disk files
using $COPY utility 4-20
using $POST utility program 4-133
using POST procedure 2-136
using RESTORE procedure 2-160
using SAVE procedure 2-165
copying diskette files to another diskette
using $DUPRD utility 4-48
using COPYI1 procedure 2-39
copying library members to a file
using $MAINT utility 4-95
using FROMLIBR procedure 2-81
copying library members to another library
using $MAINT utility 4-95
using LIBRLIBR procedure 2-115
copying spool file entries to a disk file
example 4-114
using $UASF utility 4-194
using COPYPRT procedure 2-43
COPYI1 procedure 2-39
COPYO FILE statement for $COPY 4-31
COPYPRT procedure 2-43
create message member utility program ($MGBLD) 4-119
CREATE procedure 2-52
creating a disk file
DISP parameter for 1-24
IFILE characteristic 1-27
using $FBLD utility 4-52
using BLDFILE procedure 2-11
creating a library
using $MAINT utility 4-94
using BLDLIBR procedure 2-15
creating a message load member
using $MGBLD utility 4-119
using CREATE procedure 2-52
creating a source or procedure member from the keyboard 4-95
creation date
definition N-4
on disk files 1-24
on diskette files 1-32
CRESTART procedure 2-54
CRT (see display screen)
current library substituting 5-18
cursor positioning 4-177
data exchange utility program ($POST) 4-133
data passing on procedure command 1-42, 4-111
DATE procedure 2-55
DATE statement 1-15
date substitution 5-18
decimal to hexadecimal conversion B-1
default region size
setting with $SETCF utility 4-146
setting with SET procedure 2-171
defer status of spooled output
assigning
using CHANGE command 3-4, 3-23, 3-42
using PRINTER statement 1-60
changing 3-4, 3-23, 3-42
DEFINEID procedure 2-57
DEFINEPN procedure 2-59
DEFINX21 procedure 2-66
DELETE procedure 2-70
deleting a disk or diskette file
using $DELET utility 4-38
using DELETE procedure 2-70
deleting library members
using $MAINT utility 4-101
using REMOVE procedure 2-153
deleting user libraries
using $DELET utility 4-38
using DELETE procedure 2-70
DFA procedure D-7
directory
listing contents of
using $MAINT utility 4-96
using LISTLIBR procedure 2-124
DISABLE procedure 2-72
disabling command keys 4-163
disabling function keys 4-162
disk
accumulating free space
using $FREE utility 4-55
using $PACK utility 4-127
using COMPRESS procedure 2-37
disk copy/display utility program ($COPY) 4-20
disk files
building an empty disk file
using $FBLD utility 4-52
using BLDFILE procedure 2-11
BYPASS parameter for 1-27
changing a label
using $RENAM utility 4-142
using RENAME procedure 2-155
changing space allocated for
using RESTORE procedure 2-160
copying from a library
using $MAINT utility 4-95
using FROMLIBR procedure 2-81
copying from diskette
using $COPY utility 4-20

disk files (continued)
copying from diskette (continued)
using $POST utility 4-133
using POST procedure 2-136
using RESTORE procedure 2-160
copying to a library
using $MAINT utility 4-95
using TOLIBR procedure 2-182
copying to another area on disk
using $COPY utility 4-20
using ORGANIZE procedure 2-128
copying to diskette
using $COPY utility 4-20
using $POST utility 4-133
using ORGANIZE procedure 2-128
using POST procedure 2-136
using SAVE procedure 2-165
creating
using $FBLD utility 4-52
using BLDFILE procedure 2-11
creation date for 1-24
DEFINEID procedure 2-57
DEFINEPN procedure 2-59
definition N-4
DEFINX21 procedure 2-66
delete-capable file
specifying FILE statement 1-27
specifying with $FBLD utility 4-52
specifying with BLDFILE procedure 2-11
deleting
using $DELET utility 4-38
using DELETE procedure 2-70
deleting records from
using $COPY utility 4-20
using $FBLD utility 4-52
using BLDFILE procedure 2-11
using FILE statement 1-27
using ORGANIZE procedure 2-128
DFA procedure D-7
DISABLE procedure 2-72
DISP parameter for 1-24
displaying
using $COPY utility 4-20
using DISPLAY procedure 2-73
using LISTFILE procedure 2-119
duplicate keys, ignoring 1-27
ENABLE procedure 2-75
FILE statement for 1-19
IFILE characteristic 1-27
label 1-20
listing contents of
using $COPY utility 4-20
using LISTFILE procedure 2-119
location of 1-21
maximum number of 1-20
renaming
using $RENAM utility 4-142
using RENAME procedure 2-155
disk files (continued)
reorganizing
using $COPY utility 4-20
using ORGANIZE procedure 2-128
retention types 1-22
saving on a diskette
using $COPY utility 4-20
using ORGANIZE procedure 2-128
using SAVE procedure 2-165
sorting index keys
using $DDST utility 4-37
using KEYSORT procedure 2-114
disk reorganization utility programs
$FREE 4-55
$PACK 4-127
disk VTOC
listing 2-35
rebuilding
using $LOADI utility 4-93
using RELOAD procedure 2-150
diskette copy utility program ($DUPRD) 4-48
diskette drives
placing online/offline 3-64
diskette files
basic data exchange for C-3
changing retention value 1-31
copying from a disk
using $COPY utility 4-20
using $POST utility program 4-133
using POST procedure 2-136
using SAVE procedure 2-165
copying from a library
using $MAINT utility 4-95
using FROMLIBR procedure 2-81
copying to a library
using $MAINT utility 4-95
using TOLIBR procedure 2-182
copying to another diskette
using $DUPRD utility 4-48
using COPYI1 procedure 2-39
copying to disk
using $COPY utility 4-20
using $POST utility program 4-133
using POST procedure 2-136
using RESTORE procedure 2-160
deleting
using $DELET utility 4-38
using DELETE procedure 2-70
FILE statement for 1-29
list of utility programs that use diskette files 1-29
listing contents of
using $COPY utility 4-20
using LISTFILE procedure 2-119
retention types 1-31
types of C-3
diskette labeling and initialization utility program ($INIT) 4-75

diskette magazine drive C-1
diskette volume ID 2-108
diskette VTOC 2-35
diskettes
  deleting active files from
    using $INIT utility 4-75
    using INIT procedure 2-107
  formats C-1
  initializing
    using $INIT utility 4-75
    using INIT procedure 2-107
  renaming
    using $INIT utility 4-75
    using INIT procedure 2-107
  types C-1
display control specifications 4-158
DISPLAY procedure 2-73
display screen format
  specifications 4-157
display screen formats
  creating
    using $SFGR utility 4-157
    using FORMAT procedure 2-77
  definition N-4
  deleting from load member
    using $SFGR utility 4-157
    using FORMAT procedure 2-77
  displaying with PROMPT statement 1-62
  example 4-189
  field placement 6-10
  maximum number of input fields 6-10
  multiple formats 6-15
  programming considerations 6-7
  replacing
    using $SFGR utility 4-157
    using FORMAT procedure 2-77
  specifications 4-157
    summary K-1
  steps in creating 6-7
  using 6-7
display station configuration
 record 2-171, 4-144
display station local data area
  at beginning of session 1-48
  changing contents 1-48
  definition N-4
  substituting from 5-14
display station operator
  control commands for 3-3
    CANCEL 3-3
    CHANGE 3-4
    CONSOLE 3-6
    HOLD 3-7
    IDELETE 3-8
    JOBQ 3-9
    MENU 3-10
    MODE 3-11
    MSG 3-12
    OFF 3-14
display station operator (continued)
  control commands for (continued)
    PRTY 3-15
    RELEASE 3-16
    STATUS 3-17
    TIME 3-20
display station session ending 3-14
display stations
  acquiring 1-74
  displaying status 3-57
  displaying status of 3-17
  exchanging work station IDs 3-38
  placing online/offline 3-64
  releasing 1-8
displaying copied spool file entries
  using $UASC utility 4-193
  using COPYPRT procedure 2-43
displaying messages
  on screen format 4-182
  using //* statement 1-77
  using //** statement 1-78
DUMP service procedure D-8

E format diskettes
  copied to disk
    using $POST utility 4-133
    using POST procedure 2-136
EBCDIC H-1
ELSE expressions 5-38
ENABLE procedure 2-75
enabling command keys 4-163
enabling function keys 4-162
end of data statement 1-76
end of data, special 2-140
END utility control statement 4-2
ending a session 3-14
ERAP service procedure D-11
error retry count
  setting with $PDSR utility 4-128
  setting with $SETCF utility 4-147
  setting with ALTERBSC procedure 2-3
  setting with SETRETRY procedure 2-175
EVOKE statement 1-17
exchanging work station IDs 3-38
exiting a field 4-174
extended character file, restore 2-193
extended character file, save 2-195
extending files 1-26
external indicators
  definition N-5
  setting with SWITCH statement 1-70
  testing 5-36
  within SSP procedures 1-70

field definition specifications 4-169
file build utility program ($FBLD) 4-52
file delete utility program ($DELET) 4-38
  copying to diskette
    using $COPY utility 4-20
    using SAVE procedure 2-165
  deleting with $DELET utility 4-38
  field exit keys 4-174
file group
  deleting with $DELET utility 4-38
  naming 1-20
  saving
    using $COPY utility 4-20
    using SAVE procedure 2-165
FILE OCL statements
  for disk files 1-19
  for diskette files 1-29
file rename utility program ($RENAM) 4-142
file size substitution 5-19
files (see disk files; diskette files)
finance subsystem 1-69
first-level message members
  active 1-51
  creating 2-52, 4-119
  for building menus 2-19, 4-13
  substituting from 5-15
fixed disk (see disk)
fixed-format menus 2-22
floppy disk (see diskette)
FORMAT procedure 2-77
  messages from J-3
format 1 4-33
  definition N-5
formats (see display screen formats; statement formats)
forms alignment when changing lines per page 6-20
forms number
  assigning
    using $SETCF utility 4-146
    using CHANGE command 3-4, 3-42
    using FORMS statement 1-35
    using PRINTER statement 1-59
    using SET procedure 2-171
FORMS statement 1-35
free-format menus 2-25
FROMLIBR procedure 2-81
function keys 4-162
  codes returned from PROMPT screen 5-16
GOTO statements 5-41
HELP procedure 2-86
hexadecimal to decimal conversion B-1
HISTCRT procedure 2-93
history file
  changing size
    using $LOADI utility 4-93
    using RELOAD procedure 2-150
  contents of 4-57, 4-57
  definition N-5
  displaying or listing
    using $HIST utility 4-57
    using $HSML utility 4-62
    using HISTCRT procedure 2-93
    using HISTORY procedure 2-102
history file display utility program ($HIST) 4-57
history file scroll utility program ($HSML) 4-62
HISTORY procedure 2-102
HOLD command 3-27
  display station operator 3-7
  subconsole operator 3-27
  system operator 3-46
holding jobs on spool file 3-7, 3-27, 3-46
I exchange
  definition N-5
  description C-3
  using $BICR utility 4-10
  using TRANSFER procedure 2-185
IDELETE command
  display station operator 3-8
  system operator 3-47
IF (IF, IFT, and IFF) expressions 5-23
  definition N-6
IFILE characteristic
  definition N-6
  specifying using $SLFL utility 4-192
  specifying with BLDFILE procedure 2-11
  specifying with FILE statement 1-27
  specifying with SETFILE procedure 2-174
IFORMAT, see I exchange
IMAGE statement 1-37
INCLUDE statement 1-42
index keys
  sorting
    using $DDST utility 4-37
    using KEYSORT procedure 2-114

indexed files
building using BLDFILE procedure 2-11
creating from a basic data exchange diskette file
using $BICR utility 4-10
using TRANSFER procedure 2-185
sorting
using $DDST utility 4-37
using KEYSORT procedure 2-114
informational messages
suppressing at a display station 3-8
suppressing at the system console 3-47
INIT procedure 2-107
initial program load (IPL) 6-18
definition N-6
using $LOADI utility 4-93
using RELOAD procedure 2-150
initializing a subsystem
using $IENBL utility 4-73
using ENABLE procedure 2-75
initializing diskettes
using $INIT utility 4-75
using INIT procedure 2-107
input fields, maximum number of 6-10
input job queue
canceling a job 3-40
canceling all jobs 3-40
changing positions of jobs 3-42
definition N-6
displaying status 3-57
displaying status of jobs 3-57
jobs
active library for 1-46
messages from 1-77
placing a job on the queue 3-9
using JOBQ command 3-9
using JOBQ statement 1-45
priority 1-45, 3-9
starting a job 3-54
stopping a job 3-60
input job queue control commands
CANCEL
display station operator 3-3
subconsole operator 3-21
system operator 3-42
CHANGE
subconsole operator 3-23
system operator 3-42
JOBQ
display station operator 3-9
START
subconsole operator 3-32
system operator 3-54
STATUS
display station operator 3-17
subconsole operator 3-33
system operator 3-57
STOP
subconsole operator 3-36
system operator 3-60
inquiry
definition N-6
intra subsystem 1-69
IPL 6-18
definition N-6
using $LOADI utility 4-93
using RELOAD procedure 2-150
job files 1-23
definition N-6
reserving disk space for 1-68
job queue (see input job queue)
job region 1-66, 6-2
definition N-6
job step
definition N-6
region for 1-66, 6-2
releasing 1-8
JOBQ command
display station operator 3-9
JOBQ statement 1-45
jobs
canceling an executing job 3-40
definition M-5
displaying status of 3-17, 3-57
enabling initiation of jobs 3-54
placing on input job queue 3-9
region 1-66, 6-2
resuming 3-53
stopping execution of a job or all jobs 3-60
stopping initiation of jobs 3-60
JOBSTR procedure 2-110
key mask, $SFGR 4-168
keyboard
using to create library members 4-95
KEYSORT procedure 2-114
keysort utility program ($DDST) 4-37
labels for disk files 1-20
definition N-7
layout sheet for $SFGR utility 4-159
libraries
assigning as active
using $SETCF utility 4-146
using SET procedure 2-171

libraries (continued)
building a new library
using $MAINT utility 4-94
using BLDLIBR procedure 2-15
changing size of library or directory
using $MAINT utility 4-94, 4-94
changing size of system library or directory
using $LOADI utility 4-93
changing size of user library or directory
compressing
using $MAINT utility 4-101
using CONDENSE procedure 2-38
deleting members
using $MAINT utility 4-101
using REMOVE procedure 2-153
deleting user libraries
using $DELET utility 4-38
using DELETE procedure 2-70
label 2-16
listing contents of
using $MAINT utility 4-96
using LISTLIBR procedure 2-124
making space available within
using $MAINT utility 4-101
using CONDENSE procedure 2-38
names 2-16, 6-3
placing members in
using $MAINT utility 4-95
using TOLIBR procedure 2-182
renaming
using $RENAM utility 4-142
using RENAME procedure 2-155
rules for naming 2-16
saving on diskette 2-170
substituting 5-18
library directory
definition N-7
sample printout 4-97
library maintenance utility program ($MAINT) 4-94
library member names 6-3
library members
adding to a file
using $MAINT utility 4-95
using FROMLIBR procedure 2-81
attributes 4-98
copying from a file
using $MAINT utility 4-95
using TOLIBR procedure 2-182
copying from a library to a library
using $MAINT utility 4-95
using LIBRLIBR procedure 2-115
deleting
using $MAINT utility 4-101
using REMOVE procedure 2-153
entering from keyboard
using $MAINT utility 4-95
library members (continued)
printing types and names of members in a file
using $MAINT utility 4-96
using LISTFILE procedure 2-119
library names 2-16
LIBRARY statement 1-46
library status
sample printout 4-97
LIBRLIBR procedure 2-115
line ID
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
line printer (5211 Printer)
setting print belt image
with $SETCF utility 4-146
with IMAGE statement 1-37
with SET procedure 2-171
line type
specifying with $SETCF utility 4-148, 4-149
specifying with OVERRIDE procedure 2-132
specifying with SPECIFY procedure 2-176
lines per page
forms alignment when changing 6-20
overriding in application program 1-60
specifying with $SETCF utility 4-146
specifying with FORMS statement 1-35
specifying with LINES procedure 2-117
specifying with PRINTER statement 1-59
specifying with SET procedure 2-171
LINES procedure 2-117
LISTFILE procedure 2-119
listing
a disk file
using $COPY utility 4-20
using LISTFILE procedure 2-119
a diskette file
using $COPY utility 4-20
using LISTFILE procedure 2-119
library contents
using $MAINT utility 4-96
using LISTLIBR procedure 2-124
the history file
using HISTORY procedure 2-102
using the $HIST utility 4-57
LISTLIBR procedure 2-124
LOAD statement 1-47
loading a program 1-47
local data area
at beginning of session 1-48
changing contents 1-48
definition N-4
substituting from 5-14
LOCAL statement 1-48
local station switched line ID
specifying with $SETCF utility 4-148

local station switched line ID (continued)
specifying with OVERRIDE procedure 2-132
log device
assigning with LOG procedure 2-127
assigning with LOG statement 1-50
LOG procedure 2-127
LOG statement 1-50

magazine drive C-1
MCS (see multinational character set)
member names, library 6-3
MEMBER statement 1-51
members (see library members)
MENU command
display station operator 3-10
MENU statement 1-53
menus
activating with MENU command 3-10
activating with MENU statement 1-53
building
using $BMENU utility 4-13
using BLDMENU procedure 2-19
definition N-7
displaying 1-53, 3-10
message control statement 4-119
message identification code (MIC) 4-120
definition N-7
message load members
active 1-51
creating
using $MGBLD utility 4-119
using CREATE procedure 2-52
definition N-7
substituting from 5-15
types 1-51
message source members 4-119
message text statements 4-120
messages
//* statement 1-77
//** statement 1-78
from an evoked job 1-17
from system console 3-48
MSG command 3-12, 3-48
MSG statement 1-54
replying to at system console 3-52
sending from keyboard 3-12, 3-48
using screen format to display 4-183
MIC (message identification code) 4-120
definition N-7
MODE command
display station operator 3-11
modems
clocking
specifying with $SETCF utility 4-147, 4-149
modems (continued)
clocking (continued)
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
rate
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
testing
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
modes
normal 3-11
standby 3-11
MRT procedures
creating 4-111
passing data to 5-6, 6-16
MRT programs
communicating with 5-6
definition N-7
MRTMAX attribute
overriding with ATTR statement 1-6
specifying with COMPILE statement 1-12
MSG command
display station operator 3-12
subconsole operator 3-28
system operator 3-48
MSG statement 1-54
Multinational Character Set Feature
batch interface M-23
changed character table M-26
conversion
data file M-18
initiating conversion utility M-2
library member M-3
data file modify M-23
explanation M-1
hardware support M-2
library member considerations M-17
library member copy back M-17
library member modification M-7
library statement count M-13
library statement list M-12
library statement modify M-13
RPG M-13
multiple files
specifying with OVERRIDE procedure 2-132
multiple requestor terminal procedures (see MRT procedures)
multiple requestor terminal programs (see MRT programs)

NEP attribute
  overriding with ATTR statement 1-8
  specifying with COMPILE statement 1-12
NEPs (never-ending programs)
  definition N-7
nested substitution expressions 5-21
number of printed copies
  changing with CHANGE command 3-42
  specifying with PRINTER statement 1-59

OCL
  definition N-7
OCL statements
  //* (message) 1-77
  //** (message) 1-78
  /* (end of data) 1-76
  * (comment) 1-76
  ATTR 1-6
  COMM 1-10
  COMPILE 1-12
  DATE 1-15
  EVOKE 1-17
  FILE
    for disk files 1-19
    for diskette files 1-29
  FORMS 1-35
  IMAGE 1-37
  INCLUDE 1-42
  JOBQ 1-45
  LIBRARY 1-46
  LOAD 1-47
  LOCAL 1-48
  LOG 1-50
  MEMBER 1-51
  MENU 1-53
  MSG 1-54
  OFF 1-55
  PAUSE 1-56
  PRINTER 1-57
  PROMPT 1-62
  REGION 1-66
  RESERVE 1-68
  RUN 1-69
  SESSION 1-69
  SWITCH 1-70
  SYSLIST 1-72
  WORKSTN 1-74
OFF command
  display station operator 3-14
OFF statement 1-55
operation control language statements (see OCL statements)
operator control commands (see control commands)
ORGANIZE procedure 2-128
OVERRIDE procedure 2-132
overriding information on a display 4-165
overriding the BSC configuration
  using $SETCF utility 4-144
  using OVERRIDE procedure 2-132

packed keys 4-29
parameters
  definition N-8
  for procedures 5-6
  on OCL statements 1-2
  on procedure commands 1-45
  on utility control statements 4-2
parentheses in statement formats xii
password security utility program ($PROF) 4-138
PATCH service procedure D-12
PAUSE statement 1-56
peer subsystem 1-69
permanent files
  definition N-8
  on disk 1-23
  on diskette 1-32
phone list
  creating, updating
    using $PNLM utility 4-130
    using $XNLM utility 4-197
    using DEFINEPN procedure 2-59
    using DEFINX21 procedure 2-66
  specifying with COMM statement 1-11
phone list utility, $PNLM 4-130
phone number list 2-59
positioning the cursor 4-177
POST procedures 2-136
preparing diskettes (see initializing diskettes)
PRESTOR procedure 2-141
primary SDLC retry count reset utility program ($PDSR) 4-128
print image
  changing with $SETCF utility 4-146
  changing with IMAGE statement 1-37
  changing with SET procedure 2-171
  specifying with $SETCF utility 4-149
Print key
  assigning printer for 1-74
print queue (see spool file)
print spooling commands
  CANCEL
    display station operator 3-3
    subconsole operator 3-21
    system operator 3-42
  CHANGE
    display station operator 3-4
    subconsole operator 3-23
    system operator 3-42

print spooling commands (continued)
HOLD
display station operator 3-7
subconsole operator 3-27
system operator 3-46
RELEASE
display station operator 3-16
subconsole operator 3-29
system operator 3-50
RESTART
subconsole operator 3-31
system operator 3-53
START
subconsole operator 3-32
system operator 3-54
STATUS
display station operator 3-17
subconsole operator 3-33
system operator 3-57
STOP
subconsole operator 3-36
system operator 3-60
PRINTER statement 1-57
printers
assigning for display station output
using $SETCF utility 4-146
using SET procedure 2-171
belt image for
changing with $SETCF utility 4-146
changing with IMAGE statement 1-37
changing with SET procedure 2-171
characters per inch (horizontal)
specifying with FORMS statement 1-36
specifying with LINES procedure 2-117
specifying with PRINTER statement 1-61
controlling output 1-51
exchanging work station IDs for 3-38
forms number
assigning with $SETCF utility 4-146
assigning with CHANGE command 3-42
assigning with FORMS statement 1-35
assigning with PRINTER statement 1-59
assigning with SET procedure 2-171
lines per inch (vertical)
specifying with FORMS statement 1-36
specifying with LINES procedure 2-117
specifying with PRINTER statement 1-61
lines per page
overriding in RPG II program 2-132
specifying with $SETCF utility 4-146
specifying with FORMS statement 1-35
specifying with LINES procedure 2-117
specifying with PRINTER statement 1-59
specifying with SET procedure 2-171
number of copies
changing with CHANGE command 3-42
printers (continued)
number of copies (continued)
specifying with PRINTER statement 1-59
placing online/offline 3-64
specifying a printer 1-57
spooling 1-59
translation table L-1, 1-39
printing files and libraries (see listing)
priority
assigning from keyboard 3-15, 3-49
assigning to a job 1-6
for a spooled job 1-6
for an evoked job 1-17
for an executing job 3-49
input job queue 1-45
of a spool writer 3-17, 3-33, 3-57
PRLIST procedure 2-142
PRMENU procedure 2-143
procedure command, passing data on 1-42, 4-111
procedures
(see also SSP procedures; service procedures)
attributes 5-7
calling 5-2
CANCEL keyword in 5-36, 5-36
CANCEL statement in 5-40
conditional expressions in 5-23
creating 5-1
creating from keyboard 4-95
example 5-43
execution of 5-5
naming 1-42, 5-1
nested 5-3
parameters 1-43, 5-6
passing data on procedure cammand 4-111
passing data on procedure command 1-42
RESET statement in 5-40
RETURN keyword in 5-36
RETURN statement in 5-41
substitution expressions in 5-8
PROF procedure 2-144
profile (see system security file)
program date 1-15, 2-55
programs
compiling 1-12
loading 1-47
passing data from a procedure command 4-111
passing data from a procedure command 1-42
releasing 1-8
running 1-69
PROMPT statement 1-62
PRSAVE procedure 2-145
PRSRC procedure 2-146
PRSRCID procedure 2-147

PRTY command
display station operator 3-15
system operator 3-49
PTF log D-5
public data network (see X.21 public data network)
put override (see overriding information on a display)

read under format 6-17
record mode
definition N-8
record-mode files 6-4
copying with $MAINT utility 4-108
copying with TOLIBR procedure 2-182
record separators
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
record, block conversion A-1
record, sector conversion A-1
region
definition N-8
specifying with $SETCF utility 4-146
specifying with REGION statement 1-66
specifying with SET procedure 2-171
REGION statement 1-66
RELEASE command
display station operator 3-16
subconsole operator 3-29
system operator 3-50
releasing a job step 1-8
releasing jobs on the spool file 3-50
reload library utility program ($LOADI) 4-93
RELOAD procedure 2-150
reloading the system library
using $LOADI utility 4-93
using RELOAD procedure 2-150
remote switched line IDs
specified by $IDSET utility 4-70
specified by DEFINEID procedure 2-57
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
remote work stations
displaying status of 3-17, 3-57
placing online/offline 3-64
REMOVE procedure 2-153
RENAM (file rename utility program) 4-142
RENAME procedure 2-155
renaming a disk file
using $RENAM utility 4-142
using RENAME procedure 2-155
renaming a diskette
using $INIT utility 4-75
using INIT procedure 2-107
renaming a library
using $RENAM utility 4-142
using RENAME procedure 2-155
reorganizing a disk file
using $COPY utility 4-20
using ORGANIZE procedure 2-128
REPLY command
subconsole operator 3-30
system operator 3-52
replying to a message 3-52
REQUESTX procedure 2-156
RESERVE statement 1-68
RESET statement 5-40
resident/swappable attribute of spool writer
changing using CHANGE command 3-23, 3-42
RESPONSE procedure 2-159
RESTART command
subconsole operator 3-31
system operator 3-53
restarting a job from the spool file 3-53
restarting a program (see checkpoint restart)
RESTORE procedure 2-160
restoring a disk file
using $COPY utility 4-20
using RESTORE procedure 2-160
resuming system activity 3-54
retention types
for disk files 1-20
for diskette files 1-31
summary for $COPY 4-33
return codes 5-16
RETURN keyword 5-36
RETURN statement 5-41
RUF (see read under format)
RUN statement 1-69
running a job 1-69
running a procedure 1-42, 5-2
examples 1-45

SAVE procedure 2-165
SAVELIBR procedure 2-170
saving a user library 2-170
saving disk files on diskette
using $COPY utility 4-20
using ORGANIZE procedure 2-128
using SAVE procedure 2-165
scratch files 1-22
definition N-8
screen format (see display screen formats)

screen format generator utility program ($SFGR) 4-157
screen format specifications 4-157
SDLC station test procedure D-15
SEC procedure D-13
second-level message members
  active 1-51
  creating 2-52, 4-119
  for building menus 2-19, 4-13
sector mode
  definition N-9
sector-mode files 6-6
  copying with $MAINT utility 4-108
  copying with TOLIBR procedure 2-182
  from System/32 6-6
sector, block conversion A-1
sector, record conversion A-1
security file restore utility program ($PRST) 4-140
security file save utility program ($PRSV) 4-141
security profile (see system security file)
self-check digits, computing 6-13
service procedures
  APAR D-3
  APPLYPTF D-5
  DFA D-7
  DUMP D-8
  ERAP D-11
  PATCH D-12
  SEC D-13
  SETDUMP D-14
  STATEST D-15
  TRACE D-16
session date
  definition N-9
  displaying 3-17
  specifying with $SETCF utility 4-146
  specifying with DATE procedure 2-55
  specifying with DATE statement 1-15
  specifying with SET procedure 2-171
  substitution 5-18
session library
  specifying 1-46
  subsitution expression 5-18
SESSION statement 1-69
sessions
  definition N-9
  displaying status of 3-17
  ending 3-14
set configuration utility program ($SETCF) 4-144
SET procedure 2-171
SETDUMP command D-14
SETFILE procedure 2-174
SETRETRY procedure 2-175
shutting down the system 3-60
source members
  creating from keyboard 4-95
special end of data 2-140
SPECIFY procedure 2-176
spool file
  canceling 3-40
  changing position on spool file 3-42
  copying to disk file 2-43, 4-194
  definition N-9
  displaying status 3-57
  holding 3-46
  releasing 3-50
  restarting 3-53
  starting 3-54
  stopping 3-60
spool writer
  changing number of separator pages 3-23, 3-42
  changing priority 3-23, 3-42
  changing resident/swappable attribute 3-23, 3-42
  displaying status 3-17, 3-33, 3-57
spooling
  assigning priority to jobs 1-60
  changing the printer 3-42
  copying spooled output 2-43, 4-194
  deferring spooled output
    using CHANGE command 3-4, 3-23, 3-42
    using PRINTER statement 1-60
  definition N-9
  specifying number of copies
    using CHANGE command 3-42
    using PRINTER statement 1-59
  specifying number of separator pages 3-23, 3-42
  specifying on PRINTER statement 1-59
spooling control commands
  CANCEL
    display station operator 3-3
    subconsole operator 3-21
    system operator 3-40
  CHANGE
    display station operator
    subconsole operator 3-23
    system operator 3-42
  HOLD
    display station operator 3-7
    subconsole operator 3-27
    system operator 3-46
  RELEASE
    display station operator 3-16
    subconsole operator 3-29
    system operator 3-50
  RESTART
    subconsole operator 3-31
    system operator 3-53
  START
    subconsole operator 3-32
    system operator 3-54
  STATUS
    display station operator 3-17
    subconsole operator 3-33
    system operator 3-57

spooling control commands (continued)
STOP
subconsole operator 3-36
system operator 3-60
SSP Interactive Communications Feature
BSC Support
using $SETCF utility 4-144
using ALTERBSC procedure 2-3
using OVERRIDE procedure 2-132
changing remote IDs
using $IDSET utility 4-70
using $IEDS utility 4-72
using $IENBL utility 4-73
using DEFINEID procedure 2-57
using DISABLE procedure 2-72
using ENABLE procedure 2-75
SSP Interactive Communications Feature
common queue space
allocated by $IENBL 4-73
allocated by ENABLE procedure 2-75
changed by $IDSET utility 4-70
changed by DEFINEID procedure 2-57
freed by $IEDS 4-72
freed by DISABLE procedure 2-72
SSP Interactive Communications Feature
subsystems
SSP message members
creating with $MGBLD utility 4-120
SSP procedures
ALTERBSC 2-3
ALTERSDL 2-6
BACKUP 2-8
BLDFILE 2-11
BLDLIBR 2-15
BLDMENU 2-19
BUILD 2-34
CATALOG 2-35
COMPRESS 2-37
CONDENSE 2-38
COPYI1 2-39
COPYPRT 2-43
CREATE 2-52
CRESTART 2-54
DATE 2-55
DEFINEID 2-57
DEFINEPN 2-59
DEFINX21 2-66
DELETE 2-70
DISABLE 2-72
DISPLAY 2-73
ENABLE 2-75
FORMAT 2-77
FROMLIBR 2-81
HELP 2-86
HISTCRT 2-93
HISTORY 2-102
INIT 2-107
JOBSTR 2-110
KEYSORT 2-114
LIBRLIBR 2-115
SSP procedures (continued)
LINES 2-117
LISTFILE 2-119
LISTLIBR 2-124
LOG 2-127
ORGANIZE 2-128
OVERRIDE 2-132
POST 2-136
PRESTOR 2-141
PRLIST 2-142
PRMENU 2-143
PROF 2-144
PRSAVE 2-145
PRSRC 2-146
PRSRCID 2-147
RELOAD 2-150
REMOVE 2-153
RENAME 2-155
REQUESTX 2-156
RESPONSE 2-159
RESTORE 2-160
SAVE 2-165
SAVELIBR 2-170
SET 2-171
SETFILE 2-174
SETRETRY 2-175
SPECIFY 2-176
STARTM 2-178
STOPM 2-179
SYSLIST 2-180
TOLIBR 2-182
TRANSFER 2-185
XREST 2-193
XSAVE 2-195
SSP utility programs
$ARSP (auto response utility program) 4-4
$BUILD (alternative sector rebuild utility program) 4-18
$BACK (backup library utility program) 4-9
$BICR (basic data exchange utility program) 4-10
$BMENU (build menu utility program) 4-13
input to 4-13
output from 4-13
$COPY (disk copy/display utility program) 4-20
$DDST (keysort utility program) 4-37
$DELET (file delete utility program) 4-38
$FBLD (file build utility program) 4-52
$FREE (disk reorganization utility program) 4-55
errors during 4-55
$HELP (help utility program) 4-57
$HIST (history file display utility program) 4-57

SSP utility programs (continued)
$HSML (history file scroll utility program) 4-62
$IDSET (Switched Line Remote ID Specification Utility Program) 4-70
$IENBL (Subsystem Initialization Utility Program) 4-73
$INIT (diskette labeling and initialization utility 4-75
$LABEL (VTOC display utility program) 4-81
$LOADI (reload library utility program) 4-93
$MAINT (library maintenance utility program) 4-94
$MGBLD (create message member utility program) 4-119
$MMSP (stop monitoring line utility program) 4-125
$MMST (start monitoring line utility program) 4-126
$PACK (disk reorganization utility program) 4-127
$PDSR (primary SDLC retry count reset utility program) 4-128
$PNLM (phone list utility) 4-130
$POST (data exchange utility program) 4-133
$PRES (resource security utility program) 4-136
$PRLT (security report utility) 4-136
$PRMN (menu security utility program) 4-137
$PROF (password security utility program) 4-138
$PRST (security file restore utility program) 4-140
$PRSV (security file save utility program) 4-141
$RENAM (file rename utility program) 4-142
$RSTRT (program restart utility program) 4-143
$SETCF (set configuration utility program) 4-144
$SFGR (screen format generator utility program) 4-157
descriptions 4-9
$SLFL (SETFILE utility program) 4-192
$UASC (CRT display of spool file entries) 4-193
$UASF (user access to spool file utility program) 4-194
$XNLM (X.21 phone number list utility program) 4-197
$XREST (extended character file restore utility) 4-201
SSP utility programs (continued)
$XSAVE (extended character file save utility) 4-202
DUPRD (diskette copy utility program) 4-48
work space for 4-48
IEDS (Subsystem Termination Utility Program) 4-72
list of 4-1
printed output from 1-72, 2-180
PRON (resource owner utility program) 4-139
storage requirements for 4-1
that use diskette files 1-29
utility control statements for 4-1
standby line
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
standby mode
changing to command mode 3-11
valid control commands 3-11
START command
subconsole operator 3-32
system operator 3-54
STARTM procedure 2-178
statement formats, conventions for describing xi
STATEST service procedure D-15
station addresses
specifying with $SETCF utility 4-148, 4-149
specifying with OVERRIDE procedure 2-132
specifying with the SPECIFY procedure 2-176
table of G-1
status
displayed by display station operator 3-17
displayed by system operator 3-57
STATUS command
display station operator 3-17
subconsole operator 3-33
system operator 3-57
STOP command
subconsole operator 3-36
system operator 3-60
STOPM procedure 2-179
storing library members in disk or diskette files 6-4
subconsole operator commands 3-21
substitution expressions 5-8
subsystem environment
defined by $IENBL utility 4-73
defined by ENABLE procedure 2-75
SWITCH statement 1-70

switch type
specifying with $IDSET utility 4-70
specifying with $SETCF utility 4-148, 4-149
specifying with DEFINEID procedure 2-57
specifying with OVERRIDE procedure 2-132
specifying with SPECIFY procedures 2-176
switched line IDs
specifying with $SETCF utility 4-148
specifying with OVERRIDE procedure 2-132
switches (see external indicators)
symbolic work station IDs 1-74
SYSLIST procedure 2-180
SYSLIST statement 1-72
system configuration record
definition N-10
rebuilding with $LOADI utility 4-93
rebuilding with RELOAD procedures 2-150
system console
assigning an alternate 3-6
definition N-10
system date 1-15, 2-55
definition N-10
system files C-4
system library (#LIBRARY)
backing up
using $BACK utility 4-9
using BACKUP procedure 2-8
definition N-10
reloading
using $LOADI utility 4-93
using RELOAD procedure 2-150
system list device
assigning with $SETCF utility 4-157
assigning with SET procedure 2-171
assigning with SYSLIST procedure 2-180
assigning with SYSLIST statement 1-72
system log device
assigning with LOG procedure 2-127
assigning with LOG statement 1-50
definition N-10
system operator commands 3-38
system printer
assigning an alternate 3-38
definition N-10
system security files
copying from disk to diskette
using $PRSV utility 4-141
using PRSAVE procedure 2-145
copying from diskette to disk
using $PRST utility 4-140
using PRESTOR procedure 2-141
listing with LISTFILE procedure 2-119
modifying
using $PROF utility 4-138
using PROF procedure 2-144
using PRSRCID procedure 2-147
System/32
sector-mode files from 6-6

TAG statements 5-41
task work area
changing size of
using $LOADI utility 4-93
using RELOAD procedure 2-150
temporary files 1-23
definition N-10
terminals (see display stations and printers)
TIME command
display station operator 3-20
subconsole operator 3-37
system operator 3-63
TOLIBR procedure 2-182
tone
specifying with $SETCF utility 4-147, 4-149
specifying with ALTERBSC procedure 2-3
specifying with ALTERSDL procedure 2-6
TRACE service procedure D-16
TRANSFER procedure 2-185
translation table, IMAGE 1-39
tributary station address
specifying with $SETCF utility 4-148, 4-149
specifying with OVERRIDE procedure 2-132
specifying with SPECIFY procedure 2-176
table of G-1

underlining in statement formats xii
UPSI switches (see external indicators)
user access to spool file utility (see SSP utility programs)
user ID
substituting 5-13
user libraries (see libraries)
definition N-10
utility control statements
definition N-10
rules for 4-2
utility programs (see SSP utility programs)

VARY command 3-64
varying a device online or offline 3-64
volume ID
  rules for 2-108
VTOC (volume table of contents)
  definition N-10
  listing with $LABEL utility 4-81
  listing with CATALOG procedure 2-35
  sample display for disk 4-82
  sample display for diskette 4-87
VTOC display utility program
 ($LABEL) 4-81

wait time for BSC
  specifying with $SETCF utility 4-148
  specifying with OVERRIDE
   procedure 2-132
work station IDs
  exchanging 3-23
  specifying on WORKSTN statement 1-74
  symbolic 1-74
work stations (see display stations and
 printers)
wrap test
  specifying with $SETCF
   utility 4-147, 4-149
  specifying with ALTERBSC procedure 2-3
  specifying with ALTERSDL procedure 2-6

X.21 phone number list utility program
 ($XNLM) 4-197
X.21 public data network
  canceling a network facility 2-156
  creating an X.21 phone number list
    using $XNLM utility 4-197
    using DEFINX21 procedure 2-66
  requesting a network facility 2-156
XLATE parameter, IMAGE 1-39
XREST (extended character file restore
 utility) 4-201
XREST procedure 2-193
XSAVE procedure 2-195

3270 emulation subsystem 1-69
3740 multiple files
  specifying with $SETCF utility 4-149
  specifying with OVERRIDE
   procedure 2-132
5211/3262 Printer
  setting print belt image
    with $SETCF utility 4-146
    with IMAGE statement 1-37
    with SET procedure 2-171

IBM System/34
System Support
Reference Manual

READER'S COMMENT FORM

SC21-5155-8

Please use this form only to identify publication errors or to request changes in publications. Direct any requests for additional publications, technical questions about IBM systems, changes in IBM programming support, and so on, to your IBM representative or to your nearest IBM branch office.

☐ If your comment does not need a reply (for example, pointing out a typing error) check this box and do not include your name and address below. If your comment is applicable, we will include it in the next revision of the manual.

☐ If you would like a reply, check this box. Be sure to print your name and address below.

Page number(s): Comment(s):

Please contact your nearest IBM branch office to request additional publications.

Name _____

Company or Organization _____

Address _____

_____

City State Zip Code

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

No postage necessary if mailed in the U.S.A.

Cut Along Line

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N. Y.

POSTAGE WILL BE PAID BY . . .

IBM CORPORATION
General Systems Division
Development Laboratory
Publications, Dept. 245
Rochester, Minnesota 55901

Fold and tape | Please do not staple | Fold and tape

IBM

System/34 System Support Reference Manual (File No. S34-36) Printed in U.S.A. SC21-5155-8

SC21-5155-8