

RECEIVED  
MAR 30 1983

District of Coquitlam  
Administration

**PUBLICATIONS  
UPDATE**

Operating System/3 (OS/3)

Job Control

Programmer Reference

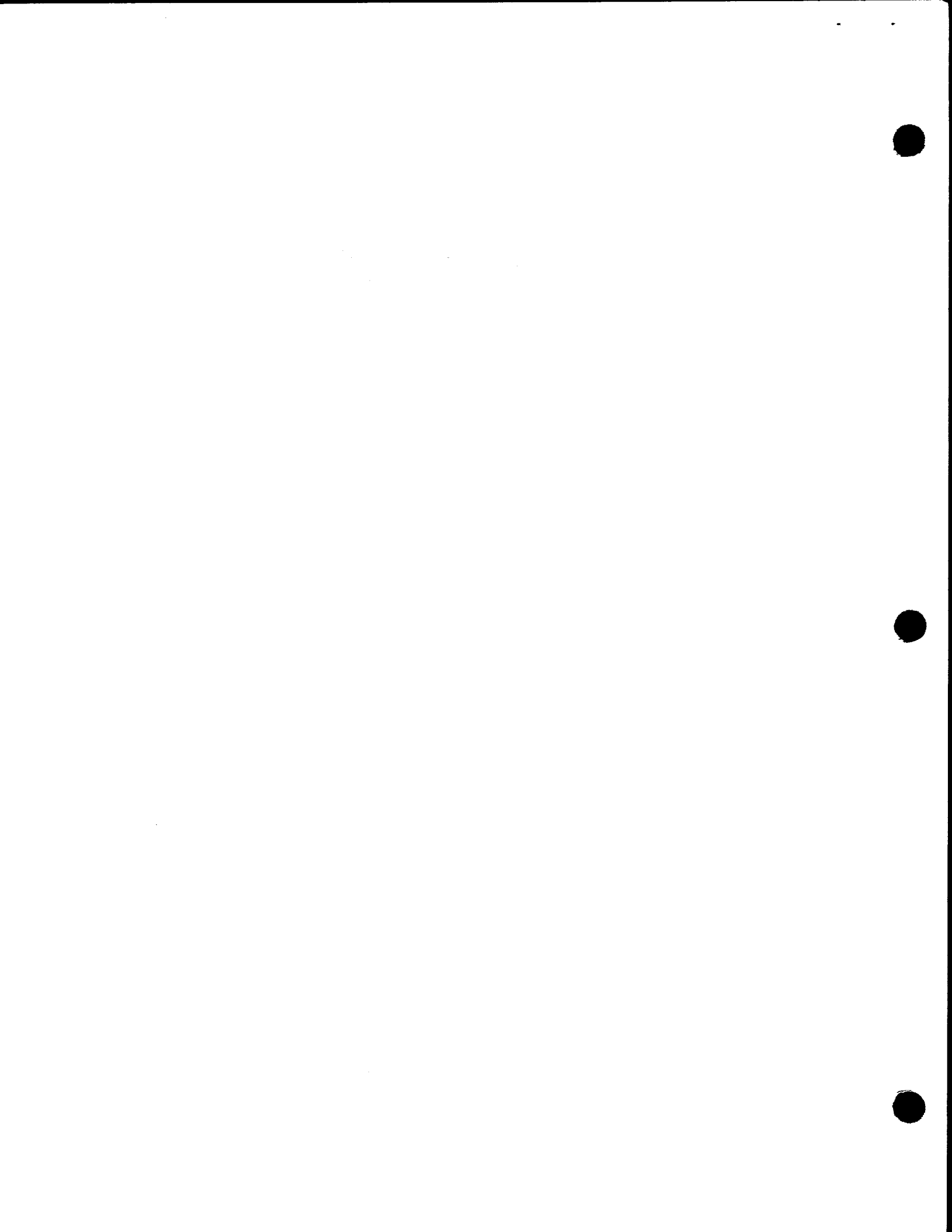
UP-8217 Rev. 8-A

This Library Memo announces the release and availability of Updating Package A to "SPERRY UNIVAC Operating System/3 (OS/3) Job Control Programmer Reference", UP-8217 Rev. 8.

This update contains corrections or clarifications applicable to features present in Job Control for Release 8.0 and prior releases.

Copies of Updating Package A are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry Univac representative. To receive only the updating package, order UP-8217 Rev. 8-A. To receive the complete manual, order UP-8217 Rev. 8.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A01, B00, B01, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package A to UP-8217 Rev. 8, 19 pages plus Memo)	Library Memo for UP-8217 Rev. 8-A  RELEASE DATE:  February, 1983



**PUBLICATIONS  
REVISION**

Operating System/3 (OS/3)

Job Control

Programmer Reference

UP-8217 Rev. 8

This Library Memo announces the release and availability of "SPERRY UNIVAC<sup>®</sup> Operating System/3 (OS/3) Job Control Programmer Reference", UP-8217 Rev. 8.

This revision documents new job control features for the 8.0 release.

The following new job control statements have been added in support of DDP, menu services, source module access, auxiliary workstation printers, alternate methods for specifying task switching priorities, and checking job and system related variables.

```
// ROUTE          // OPTION PRI
// DVC PROG       // USE LIB
// INQ JOB        // USE MENU
// INQ SYS
// OPTION OUT
```

Changes (in the form of new parameters, altered parameters, or alternate formats) have been made to the following statements:

```
// ALTJCS        // JNOTE          // OPTION NOSCHED
// DD            // OPR            // OPTION SAVE
// DVC           // OPTION LOG     // PAUSE
// DST           // OPTION MAS     // SPL
// EXT           // OPTION ORI     // UID
```

Changes in the majority of these statements reflect DDP support, increased workstation support, the ability to specify an alternate library for saved, translated control streams, enhancements to screen format services, and enhancements for controlling spooled output. While changes to the EXT job control statement reflect enhancements (the ability to change the automatic allocation amount), most of the changes are reflected in the removal of parameters for split cylinder file allocation.

A new jproc (SPOOL) is available for controlling spooled output. It provides the same parameters as the SPL job control statement but in keyword rather than positional format for easier coding.

The DVCDKT jproc is new for diskette and parallels the function of the DVCVOL and DVCVTP jprocs.

All other changes are corrections or clarifications applicable to features present in job control for 7.1 and prior releases.

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

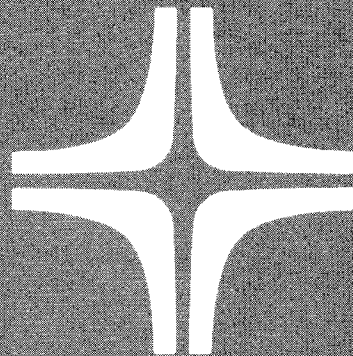
LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A01, B00, B01, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76, and 76U (Cover and 211 pages)	Library Memo for UP-8217 Rev. 8  RELEASE DATE:  September, 1982

Copies of UP-8217 Rev. 7 and UP-8217 Rev. 7—A will be available for 6 months after the release of 8.0. Should you need additional copies of this edition, you should order them within 90 days of the release of 8.0. When ordering the previous edition of a manual, be sure to identify the exact revision and update packages desired and indicate that they are needed to support an earlier release.

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

# Job Control

OS/3



Programmer Reference

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

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

Sperry Univac is a division of the Sperry Corporation.

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

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

PAGE STATUS SUMMARY

ISSUE: Update A – UP-8217 Rev. 8  
RELEASE LEVEL: 8.0 Forward

Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level
Cover/Disclaimer		Orig.	Appendix B	Title Page 1	Orig. Orig.			
PSS	1	A	Appendix C	Title Page 1, 2	Orig. Orig.			
Preface	1 thru 3	Orig.	Glossary	1 thru 9	Orig.			
Contents	1 thru 5	Orig.	User Comment Sheet					
Section 1	Title Page 1 thru 8 9 10	Orig. Orig. A Orig.						
Section 2	Title Page 1 thru 18 19 20 21 22, 23 24 25 thru 30 30a, 30b 31 32 33 thru 62 63 64 thru 75 76 77 thru 88	Orig. Orig. A Orig. A Orig. Orig. Orig. Orig. A Orig. A Orig. Orig.						
Section 3	Title Page 1 thru 12 12a thru 12g 13 thru 16 16a 17 thru 40 40a 41 42 43 thru 54 54a 55 thru 62	Orig. Orig. Orig. Orig. Orig. Orig. Orig. Orig. A Orig. Orig. Orig.						
Section 4	Title Page 1 thru 4	Orig. Orig.						
Appendix A	Title Page 1 thru 5	Orig. Orig.						

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





## Preface

This programmer reference manual is one in a series to be used as a quick-reference document for programmers familiar with the SPERRY UNIVAC Operating System/3 (OS/3). This particular manual contains descriptions of the job control statements and procs used to communicate with job control, and the proc definition statements that allow you to expand and conditionally modify the job stream when you start the job.

The information contained in this manual is presented as follows:

- Section 1. General Information

Describes the control stream concepts and coding conventions that must be adhered to when writing job control statements and procs.

- Section 2. Job Control Statements

Describes the function and parameters of each job control statement in alphabetic order by statement mnemonic.

- Section 3. Job Control Procs

Describes the function and parameters of each job control proc in alphabetic order by proc mnemonic.

- Section 4. Proc Definition Statements

Describes the function and parameters of each proc definition statement in alphabetic order by statement name.

- Appendix A. Logical Unit Number Assignment Information

Provides the information needed to assign or change the logical unit numbers of the peripheral devices in a system.

- Appendix B. Extent Specification Information

Provides the information required to establish or extend a disk or diskette file.

- Appendix C. SPERRY UNIVAC Disk Subsystem Characteristics

Lists the physical characteristics of each of the disk subsystems that may be used with OS/3.

- Glossary

Defines the terms peculiar to job control.

The term Series 90 in this manual refers to 90/25, 90/30, and 90/40 systems. System 80 refers to System 80, models 3 and 5.

The information presented in this programmer reference is limited to facts. Related information and examples of use are presented in other current OS/3 manuals:

- System 80

Job control user guide, UP-8065

Describes OS/3 job control concepts and examples of use

Workstation user guide, UP-8845

Describes use of the workstation in OS/3

Operations handbook for operators, UP-8859

Describes operator procedures in a System 80 environment

File cataloging concepts and facilities, UP-8860

Describes file cataloging concepts and facilities

Supervisor user guide, UP-8832

Describes the functions of the OS/3 supervisor

Consolidated data management macroinstruction user guide/programmer reference, UP-8826

Describes the consolidated data management macroinstructions

COBOL supplementary reference, UP-8613

Describes ANSI 74 COBOL

System installation user guide/programmer reference, UP-8839

Describes system installation procedures

Screen format services concepts and facilities, UP-8802

Describes the procedures for creating, maintaining, and using screen formats

Menu services concepts and facilities, UP-9317

Describes the procedures for creating and using menus

Dialog processor user guide/programmer reference, UP-8859

Describes the creation of user-written dialogs

Distributed data processing concepts and facilities, UP-8811

Describes OS/3 distributed data processing

■ Series 90 Systems

Job control user guide, UP-8065

Describes OS/3 job control concepts and examples of use

Workstation user guide, UP-8845

Describes use of the workstation in OS/3

Operations handbook for operators, UP-8511

Describes operator procedures in a 90/25 environment

Operations handbook for operators, UP-8072

Describes operator procedures in a 90/30 and a 90/40 environment

File cataloging concepts and facilities, UP-8860

Describes file cataloging concepts and facilities

Supervisor user guide, UP-8832

Describes the functions of the OS/3 supervisor

Data management user guide, UP-8068

Describes the data management macroinstructions

COBOL supplementary reference manuals, UP-8057 and UP-8059

Describe basic and extended COBOL

System installation user guide/programmer reference, UP-8074

Describes system installation procedures

Screen format services concepts and facilities, UP-8802

Describes the procedures for creating, maintaining, and using screen formats

Menu services concepts and facilities, UP-9317

Describes the procedures for creating and using menus

Dialog processor user guide/programmer reference, UP-8859

Describes the creation of user-written dialogs

Distributed data processing concepts and facilities, UP-8811

Describes OS/3 distributed data processing



# Contents

## PAGE STATUS SUMMARY

## PREFACE

## CONTENTS

### 1. GENERAL INFORMATION

JOB CONTROL OVERVIEW	1-1
CONTROL STREAM CONCEPTS	1-1
STATEMENT CONVENTIONS	1-1
JOB CONTROL STATEMENTS	1-4
General Format	1-4
Coding Conventions	1-5
Statement Continuation	1-7
JOB CONTROL PROCS	1-7
PROC DEFINITION STATEMENTS	1-7
Coding Conventions	1-8
Character Set	1-8
Terms	1-8
Parameters	1-9
Parameter Referencing	1-9
Control Stream Considerations	1-10
SYSTEM LIBRARY FILE NAMES	1-10

### 2. JOB CONTROL STATEMENTS

ALTER	2-1
ALTJCS	2-3

---

CAT	2-5
CC	2-6
CR	2-7
DATA FILEID	2-8
DATA STEP	2-9
DD	2-10
DECAT	2-12
DST	2-13
DVC	2-14
→ DVC PROG	2-17
EQU	2-18
EXEC	2-19
EXT	2-21
FIN	2-25
FREE	2-26
GBL	2-27
GO	2-28
IF	2-29
↓ INQ JOB	2-30
↑ INQ SYS	2-30a
JNOTE	2-30b
JOB	2-31
JSET	2-35
LBL	2-36
LBL (File Catalog)	2-38
LCB	2-41
LFD	2-45

MTC	2-47
NOP	2-48
OPR	2-49
OPTION	2-50
PARAM	2-58
PAUSE	2-59
QGBL	2-60
QUAL	2-61
REN	2-62
ROUTE	2-63
RST	2-64
RUN/RV	2-66
SCR	2-68
SET	2-69
SFT	2-71
SKIP	2-73
SPL	2-74
UID	2-77
USE DP	2-78
USE LIB	2-79
USE MENU	2-80
USE SFS	2-81
VFB	2-82
VOL	2-84
/\$	2-86
/*	2-87
/&	2-88



### 3. JOB CONTROL PROCS

	ACCESS	3-1
	ALLOC	3-3
	ASM	3-8
→	AUTO	3-12a
	COBL74	3-12g
	COBOL	3-16a
→	DVCDKT	3-20
	DVCVOL	3-21
	DVCVTP	3-22
	FORT	3-23
	LINK	3-30
	LNKUPL	3-35
	RPG II	3-36
→	SPOOL	3-40a
	UDD	3-43
	UDT	3-49
	UPLCNV	3-51
	UTD	3-52
	WORK/TEMP	3-58
	WRTBIG/WRTSML	3-61

### 4. PROC DEFINITION STATEMENTS

	END	4-1
	NAME	4-2
	PROC	4-3
	procname	4-4



**APPENDIXES****A. LOGICAL UNIT NUMBER ASSIGNMENT INFORMATION**

SPECIAL DEVICE CATEGORY	A-1
PRINTERS	A-1
CARD READER SUBSYSTEMS	A-2
CARD PUNCH SUBSYSTEMS	A-2
DISK SUBSYSTEMS	A-3
DISKETTE SUBSYSTEMS	A-3
MAGNETIC TAPE SUBSYSTEMS	A-3
WORKSTATIONS	A-4
STANDARD LOGICAL UNIT NUMBERS	A-4

**B. EXTENT SPECIFICATION INFORMATION****C. SPERRY UNIVAC DISK SUBSYSTEM CHARACTERISTICS****GLOSSARY****USER COMMENT SHEET****FIGURES**

1-1. Typical Control Stream	1-2
-----------------------------	-----

**TABLES**

A-1. Standard Logical Unit Number Assignments	A-4
---	-----



## **1. General Information**



## JOB CONTROL OVERVIEW

Job control is a component of the SPERRY UNIVAC Operating System/3 (OS/3) executive; the other component is the supervisor. Job control consists of a series of transient routines and is activated by a job control statement or an operator command input from the system console. It is by means of these transient routines that job control manages the system resources and prepares a job for execution.

Specifically, job control:

- analyzes the input control stream;
- expands job control procedure calls;
- allocates peripheral devices and main storage; and
- schedules jobs for execution.

Job control directs and controls the flow of jobs through the system. The environment of a job and its various job steps is defined and controlled by job control statements in a control stream. As the control stream is processed, job control calls on transient routines to perform the required functions. A control stream may be read into the job control stream library (\$Y\$JCS) or an alternate SAT library file for permanent storage or into a temporary job run library file (\$Y\$RUN) for scheduling and execution. Jobs are scheduled for execution based on a priority scheme and the availability of system resources. If the required resources are available for a job in the highest priority queue, the job is executed. A job is executed one job step at a time until the job is completed. Housekeeping is performed as part of the job termination process.



## CONTROL STREAM CONCEPTS

A control stream is a series of job control statements and job control procedures (jprocs) that define the job and direct its execution (Figure 1-1). It acts as an interface between the job and the operating system. It may also include source code or data as required by the job.

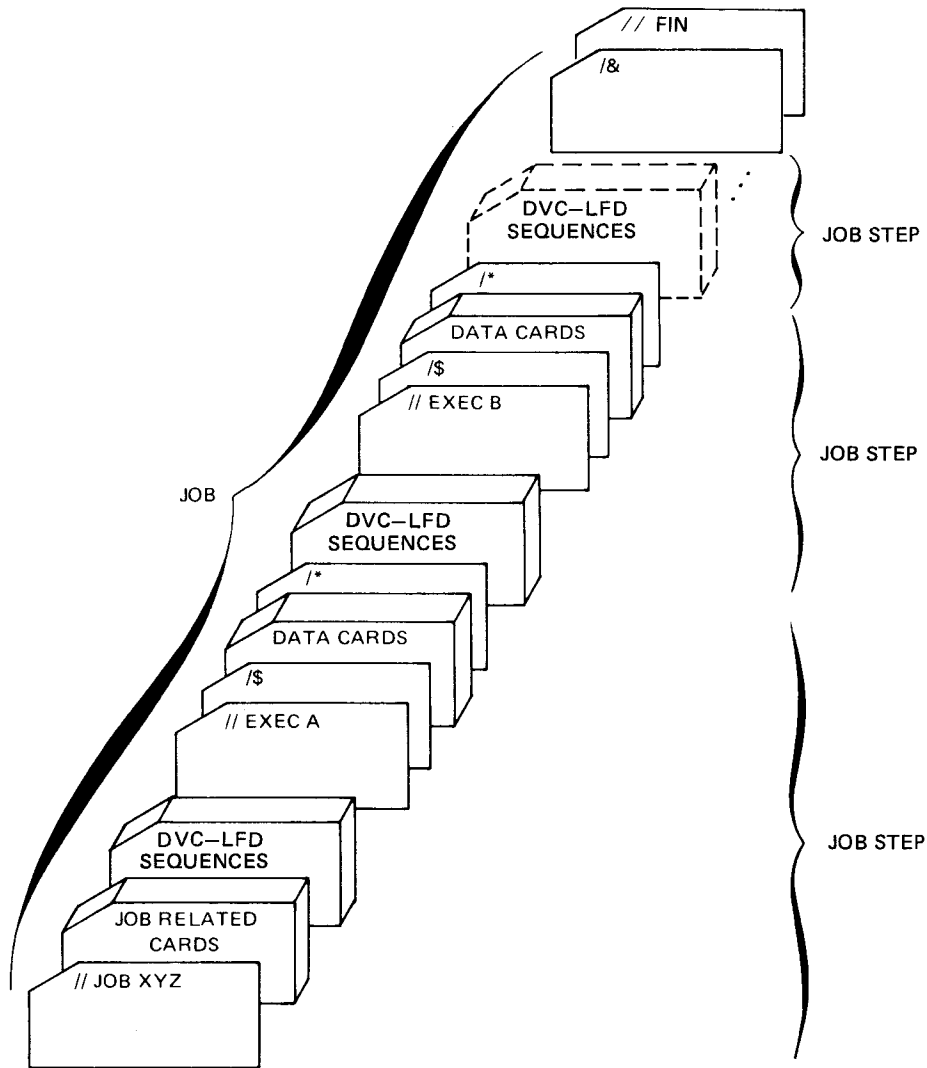
One control stream is applicable to only one job; each job is subdivided into one or more steps and a temporary \$Y\$RUN file is created for each job. Any number of jobs, however, may be stored on disk for subsequent scheduling. Up to 7 or 14 jobs may be active concurrently, depending on the availability of system resources and the configuration of your system. Each job, and therefore each control stream, stored on disk must be given a unique name.

Figure 1-1 illustrates a job control stream for a job divided into several job steps.

## STATEMENT CONVENTIONS

The conventions used to illustrate the coding formats presented in this manual are:

- Information that must be coded exactly as shown is presented in uppercase letters. Commas, equal signs, parentheses, slashes, and percent signs also must be coded as shown.
- Information that can take various forms is presented in lowercase letters.



NOTE:

Devices allocated (through DVC-LFD sequences) for one job step are available to all succeeding job steps.

Figure 1-1. Typical Control Stream

- Unless otherwise indicated, all parameters are positional parameters and must be coded in the order specified in the operand field and separated by commas. When a positional parameter is omitted, its associated comma must be retained to indicate the omission, except for the case of omitted trailing parameters. For example, the control statement:

```
//[symbol] LFD { filename } [ , { n } ] [ , { 8 } ] [ ( ACCEPT )
( EXTEND )
( INIT )
( PREP )
( RELOD ) ]
```

could be coded in any of the following ways:

```
// LFD DISKIN,1,INIT
// LFD DISKIN,,RELOD
// LFD DISKIN
```

- Keyword parameters, which take the form

**KEYWORD=variable**

can be coded in any sequence following the last positional parameter. Keyword parameters of the type

**KEYWORD**

can be coded in random sequence only when so indicated in the text accompanying the control statement; otherwise, they must be coded as positional parameters.

- A job control statement may consist of a group positional parameters followed by a keyword parameter (as the last parameter).

For example:

```
//[symbol] EXEC program-name [ , (library-name) [ , [±] switch-priority]
                               { $YSRUN
                               { SYSLOD
                               }
                               ]
                               [ , ABNORM=label ]
```

Since the last parameter is a keyword (not the last positional) parameter, this statement may be written as follows:

```
// EXEC program-name, ABNORM=label
// EXEC program-name, library-name, ABNORM=label
```

Commas for the omitted positional parameters may be retained if desired. For example:

```
// EXEC program-name,,, ABNORM=label
// EXEC program-name, library name,, ABNORM=label
```

The conventions for coding commas when a positional parameter is omitted and subsequent positional parameters are being specified still apply. When the second positional parameter is omitted, for example, the EXEC statement must be coded as follows:

```
// EXEC program-name,, switch-priority, ABNORM=label
```

- Information that may optionally be coded is enclosed in brackets:

```
[ optional
  information ]
```

- When more than one option exists for any given parameter, the options are listed within braces:

```
{
option-1
option-2
option-n
}
```

- Parameters enclosed in parentheses are called subparameters and are subject to the same rules regarding parameters. The parentheses enclosing subparameters must be coded to delimit a series of one or more subparameters.
- An ellipsis (a series of three periods) is used to indicate the omission of a variable number of similar parameters:

```
lun-2,type-code,...,lun-5,type-code
```

- Optional parameters that have a default specification are shaded:

```
{ library-name }
{ SYSRUN }
```

- When a portion of a parameter is underlined, only that portion need be specified. For example:

```
FORMNAME=symbol
```

can be coded as

```
F0=symbol
```

## JOB CONTROL STATEMENTS

Each job control statement is described in alphabetic order in Section 2. The parameters associated with a statement are described in the order of their appearance when the statement format is scanned from left to right and from top to bottom within a parameter. If a default condition for a multiple-option parameter is available, it is shaded.

### General Format

A job control statement has a maximum of five fields, which appear in the following order:

- Indication Field

Distinguishes job control statements from data. It is required and begins with //, /&, /\$, or /\*.

- Label Field

Contains an alphanumeric character symbol, of which the first character must be alphabetic. Unless this field is explained in a specific control statement, it is the target address of a SKIP control statement or the ABNORM keyword parameter of the EXEC statement. This field is not separated from the indication field by a space; it immediately follows the //.



■ Operation Field

Contains the name of the function to be performed. It is required for all job control statements having an indication field of // and must be preceded by a blank.

■ Operand Field

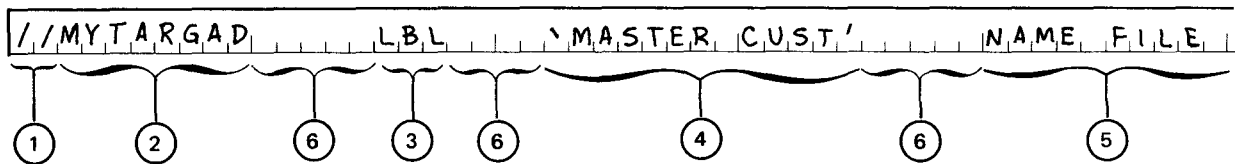
Contains the specific information concerning the items upon which a job control function is to operate or the manner in which the function is to be performed. This field must be preceded by a blank.

■ Comments Field

Contains any descriptive information desired, but not processed. The field must not contain a slash character. For those job control statements in which an operand is not permitted, such as the FIN control statement, all information beyond the operation field is treated as the comments field. This field must be preceded by a blank.

Excluding the indication and label fields, consecutive fields must be separated by one or more spaces. A space may not appear in a field except within apostrophes (hexadecimal code 7D) or parentheses in an operand field.

Example:



- 1. Indication field
- 2. Label field
- 3. Operation field
- 4. Operand field -  
Note that spaces are allowable due to the use of apostrophes.
- 5. Comments field
- 6. Field separation spaces

**Coding Conventions**

- Data cannot be contained in a job control statement.
- Embedded data is normally assumed to be 80 characters long; input from diskette can be 80 or 128 characters long.
- Comments must not contain a slash.
- Job control does not scan past position 72; however, embedded data of up to 128 bytes is passed through. Positions 72 through 80 may also be used for a continuation character and sequencing.



- For a multistatement card:
  - Each statement, except the last, is terminated by one or more blanks followed by the first slash of the next statement.
  - Statements are processed from columns 1 through 71. Column 72 is used to indicate continuation.
  - CR and procname must be the last statement on the card.
  - Any statement may be continued at any delimiter, such as the blanks following the operation or operand fields or after the comma following a parameter in the operand field. A nonblank character must be coded in column 72.
- JOB, FIN, PROC, NAME, or END must be the only statement on a card.
- /\$, /\*, or /& must be the only statement on a card.
- Numbers required for particular parameters can be expressed in decimal or hexadecimal form. Numbers preceded by D' are considered decimal. Numbers preceded by X' are considered hexadecimal. (A trailing quote may optionally be specified.) All of the following represent the same value:

X'FF

X'FF'

D'255

D'255'

Numbers not preceded by X' or D' are automatically considered decimal except in the following cases when they default to hexadecimal:

- Main storage sizes specified on the JOB statement (*min* and *max* parameters)
  - Main storage sizes specified on the OPTION MIN and OPTION MAX job control statements
  - Absolute disk addresses specified on the EXT statement (*addr* or *Tccc:hh* parameters)
  - Address on the ALTER statement (*address* parameter)
  - Expansion limit on the SFT statement's DLOAD option (*expansion-limit* parameter)
- Character strings on the ALTER, LCB, and SET job control statements must be specified as shown in their formats.

## Statement Continuation

A continuation line is not considered a statement in itself. It is a line that contains the continuation of the immediately preceding control statement. A nonblank character must appear in column 72 of each statement being continued. Continuation statements may be used to extend any job control statement for which at least the first two fields are already coded.

A continuation line must begin with either the 3-character sequence `//n`, or just a simple `//`, separated by one or more blanks from the continuation portion of the control statement. The general format of the continuation statement is:

```
//[ n] p1, . . . ,pn
```

where:

[ n]

Is a decimal number from 1 through 9. The numbers do not need to be consecutive; however, each successive number must be greater than or equal to the preceding number within a specific group of continuation statements. This is an optional field and may be left blank, or numbers may be used so the user can keep a visual record of the amount of continuation statements used.

p1, . . . ,pn

Are the parameters required to continue the immediately preceding statement.

Any statement may be continued at any delimiter. The delimiters are the blanks following the operation or operand fields, or after the comma following a parameter in the operand field. Subparameters may not be divided.

## JOB CONTROL PROCS

The job control procs supplied as part of OS/3 are described in alphabetic order in Section 3. The parameter descriptions are presented in the same order specified for the job control statements. The coding formats of these procs are illustrated in the proc descriptions, using the statement conventions previously described.

## PROC DEFINITION STATEMENTS

A job control procedure definition consists of a PROC definition statement, one or more NAME definition statements, and a series of job control statements; it ends with an END definition statement. The PROC definition statement is used to signal the beginning of the procedure, the NAME definition statement declares labels by which the procedure can be called for execution, and the END definition statement signals the end of the procedure. Each time the series of job control statements is needed, a procedure call statement is written; job control then inserts the necessary number of statements at the point of reference. The procedure definition specifies to job control the coding and statements for a particular operation, and the procedure call statement specifies the values of the variable parameters to be used when the call is executed.

## Coding Conventions

Statements and comments used in procedure definitions are generally written in columns 1 through 71. Column 72 is used to indicate continuation. Columns 73 through 80 may contain program identification and sequencing information; job control does not process these columns.

The job control statements within the body of the procedure definition follow standard job control statement conventions with respect to multistatement cards and continuation. If multistatement images are present in the procedure definition, they are expanded and passed to the run symbiont one statement at a time. If job control statements with continuation appear in the body of the procedure definition, they are expanded and passed to the run symbiont in the same form, with continuation. Job control does not attempt to verify whether continuation is allowed on the particular control statement.

## Character Set

The character set used in writing procedure definition statements consists of:

Letters	A through Z
Special letters	? \$ # @
Digits	0 through 9
Special characters	+ - * / , = ' blank ( ) . > < & ! ; ;

## Terms

The values that may be used in the operand field of a procedure definition statement may be symbol or a character string.

### ■ Symbols

A symbol consists of a group of up to eight alphanumeric characters used for parameter identification and as labels. The first, or leftmost, character must be alphabetic. Special characters or blanks may not be contained within a symbol.

The label on a NAME directive cannot exceed eight characters. The operand in a NAME directive may be obtained by referencing the symbol p(0), where p is the symbol assigned in the PROC directive used to reference any positional parameter in the definition, and (0) is the number of the positional parameter. The parameter in the operand of the NAME directive is addressed as positional parameter 0.

### ■ Character String

A character string can represent up to 252 valid characters, all of which must be printable. Character strings containing embedded blanks or commas must be enclosed in apostrophes or parentheses, which are considered part of the character string. A character string may not contain an embedded apostrophe.

A null character string is represented by two consecutive apostrophes.

All parameter values are evaluated as character strings.

## Parameters

Parameters pass information from the procedure call statement, through the procedure definition, to the body of the procedure. Parameters may be equated to values, symbols, or character strings and may be used to specify label information, file names, volume serial numbers, etc. There are two types of parameters: positional and keyword. Positional parameters are identified by their position within the operand field of the procedure call statement; keyword parameters are referenced or identified by the symbols assigned to them in the directive.

Both positional and keyword parameters may be sublisted. Thus, each operand of the procedure call statement may represent one value or a series of values that may be referenced independently. When a parameter is sublisted, the subparameters must be separated by commas and the entire list enclosed in parentheses.

An omitted positional parameter has the value of a null character string. An omitted keyword parameter that had a preset value specified in the PROC directive is given that preset value. An omitted keyword subparameter that had a preset value specified in the PROC directive is not given that preset value. If no preset value was given, the value of the keyword parameter is a null character string.

## Parameter Referencing

Job control statements coded in the body of a procedure definition follow the same rules and conventions as all other job control statements, with one exception. The parameters in the operand field of the job control statements that require substitute values at job execution time must begin with an ampersand (&). The positional parameters in the operand field of the job control statements, whose values will be changed when the procedure definition is called, must have some sort of symbol included in the parameter specification. For example, in the body of a procedure definition, if you had a DVC statement in which you want to vary the logical unit number, it would be coded:

```
// DVC &P(2)
```

The ampersand (&) is the indicator; the P is the symbol you assigned in the PROC directive; the (2) indicates that the logical unit number to be inserted in the control statement is coded as positional parameter 2 in the procedure call statement. The parentheses are required.

Parameter substitution is performed at job execution time by the run symbiont when it expands the procedure definition.

Substitution is attempted only on your embedded procedure call statements. PROC or NAME directives are not candidates for substitution.

For each valid character string following a single ampersand, a substitution is made. If a valid substitution exists, it is used; if no valid substitution exists, the null character string is substituted.

The characters ampersand, period, parentheses, apostrophe, blank, comma, plus, minus, asterisk, and slash (& . ( ) ' Δ , + - \* /) may not be embedded in the dummy arguments on the PROC directive.

Any statement can be continued only between operands or between the command and the first operand. This limits the length of operands on the input source (primary control stream or procedure file) to 65 characters.

The length of a single operand may not exceed 252 characters.

The length of a single parameter is limited to 242 characters. For positional parameters, this is the value field; for keyword parameters, this is the value plus the key length. If a parameter is sublisted, the maximum length is decreased by 2 for each element of the sublist.

## Control Stream Considerations

Procedure definitions are filed by the file symbiont in either `$$JCS` or an alternate library file. The new values to be used are not substituted for the preset values until the procedure definition is called. Substitution takes place then and the procedure definition is expanded. A procedure definition may be called as often as necessary or until the procedure definition is deleted from `$$JCS` or the alternate library file.

A job input directly from the card reader may include procedure definitions in its control stream. Procedure definitions must appear in the control stream before any references are made to them. They may not be embedded in a data sequence (`/$ - /*` statements).

A procedure definition filed in `$$JCS` or an alternate library file may be overridden by reading in a new version from the card reader, a diskette, or the system spool file.

## SYSTEM LIBRARY FILE NAMES

The system library files for OS/3 are called by their file identifiers. These allow a program to access a file via its symbolic name. In this way, a file is not device dependent, and the programmer is not hampered by physical device restrictions in his program specifications. The file identifiers used for the system library files are:

- Load library file                    `$$LOD`
- Object library file                  `$$OBJ`
- Source library file                  `$$SRC`
- Macro library file                  `$$MAC`
- Job control stream library file    `$$JCS`

A temporary job run library (`$$RUN jobname`) is also established for each job input to the system. These files exist only for the duration of the job, unless the job's `$$RUN` file is saved through the `OPTION SAVE` or `OPTION NOSCHED` job control statement.

## **2. Job Control Statements**





## ALTER

### Function:

Alters load modules at execution time. ALTER control statements must precede the EXEC control statement in the job step. ALTER control statements are processed by the supervisor prior to the transfer of control to the user program.

### Format:

```
//[ symbol] ALTER [ phase-name][ ,address][ ,change] [ , {RESET }
                                     {ORG } ]
```

### Parameters:

#### phase-name

One to eight alphanumeric characters specifying the name of the phase to be altered. If less than eight characters are specified, the phase being altered is assumed to be an alias phase and the name is padded on the right with EBCDIC blanks to make eight characters. The names of all other phases must be specified by using the full eight characters.

If omitted, the previously named phase is altered.

#### address

A 1- to 5-digit address that specifies the main storage location into which the bytes specified by positional parameter 3 are to be stored; considered hexadecimal if coded *X'number* or *number*, and decimal if coded *D'number*. If positional parameter 4 is *RESET*, address may be omitted. ←

If omitted and an address is required, zero is used.

#### change

Specifies the contents of the bytes to be stored at the address specified by positional parameter 2. Changes are not stored if the address is invalid. The change must be specified in hexadecimal or EBCDIC.

- Hexadecimal

X'cccccccc...' or cccccccc...

The number of characters must be even, and the maximum number of hexadecimal characters allowed is 16 (eight bytes).

- EBCDIC

C'cccccccc'

The maximum number of EBCDIC characters allowed is 8 (eight bytes).

If positional parameters 2 and 3 are omitted, this is a null ALTER control statement, and control is returned to the user program.

If positional parameter 4 is used, *change* may be omitted.

## ALTER

### RESET

Resets the alter mode indicator. If all other parameters are omitted, control returns to the user program with the alter mode indicator reset.

### ORG

Adds the address specified in positional parameter 2 to all addresses on succeeding ALTER control statements until the next ALTER control statement with a RESET parameter is encountered or the final ALTER control statement for the phase is processed.

If *RESET* or *ORG* is not specified, the control stream is read until an ALTER control statement with the *RESET* or *ORG* parameter is detected or the final ALTER control statement for the phase is processed.

**ALTJCS**

## Function:

Specifies an alternate SAT library file (one other than `$$JCS`) to be searched for jprocs. In System 80, the alternate library file can be located on format-label diskette as well as disk. ←

## Format:

```
//[symbol] ALTJCS [file-label-id] [ , { RES
                                RUN
                                vol-ser-no } ] [ , rpw ] [ , { FREE
                                                                ONLY
                                                                OFF
                                                                ON } ] [ , LUN=nnn] ←
```

## Parameters:

**file-label-id**

A 1- to 44-alphanumeric-character label name of the file to be searched for jprocs; is optional only if you're not activating the search of a new library but changing the last parameter (*FREE*, *ONLY*, *OFF*, or *ON*) for an alternate library already defined in a previous ALTJCS statement. If the *file-label-id* is not specified, then *vol-ser-no* and *rpw* cannot be specified.

**vol-ser-no**

Specifies the volume serial number of the file to be searched for jprocs. If a *vol-ser-no* is not specified, the cataloged *vol-ser-no* is used; if it is not cataloged, RES is used.

**rpw**

Specifies the read password of the specified cataloged file that contains the jprocs needed for the job.

**FREE**

Specifies that only `$$JCS` is to be searched and frees the alternate device (from the run processor) after the search is completed.

**ONLY**

Specifies that only the identified alternate library file is to be searched.

**OFF**

Specifies that only `$$JCS` is to be searched. You specify this option if you no longer want an alternate library file searched for jprocs. The alternate library file remains open to the run processor and can be searched again by use of the *ON* or *ONLY* options.

**ON**

Specifies that the identified alternate library file is to be searched first and then `$$JCS` is to be searched; ON is the default order-of-search option.

↓

**LUN=nnn**

Supplies a logical unit number indicating the device type and characteristics for the alternate library; never specified unless a volume serial number is also specified; helps (in System 80) to determine whether a disk or format-label diskette is required (since volume serial numbers for disk and format-label diskette are syntactically the same); can be specified with Series 90 although the required device for the alternate library is always a disk; used only to determine device type and characteristics and has no relationship to logical unit numbers used elsewhere in the control stream.

**NOTE:**

↑

*If your job control stream is in an alternate library, you cannot use the // ALTJCS statement to specify a different library. You can use it only to specify the options FREE, ONLY, OFF, or ON. (If the job stream is in \$\$JCS, the // ALTJCS statement can reference any alternate library.)*

**CAT****Function:**

Catalogs the file defined in a previously encountered device assignment set in the job. The file is identified by the lfdname parameter in this statement and in the LFD job control statement. The read/write passwords are specified in the LBL job control statement.

**Format:**

```
//[ symbol] CAT lfdname[ ,catpw][ ,SCR] [ , {GEN=nn}
                                     {MEM} ]
```

**Parameters:****lfdname**

File name; must agree with the file name of the associated LFD job control statement.

**catpw**

Password of one to six alphanumeric characters for the catalog itself; must be specified if the catalog itself is password protected.

**NOTE:**

*Even if all the files in a file catalog have been decataloged (using // DECAT), the catalog password remains in effect. Catalog passwords are established, removed, and changed through the JC\$CAT routine, which is explained in the file cataloging concepts and facilities manual. When you use // CAT or // DECAT, you must specify catpw if a password has been established through the JC\$CAT routine.*

**SCR**

Indicates the file should be scratched when it is removed from the catalog. This parameter is effective only for specific references. (If several files are removed from the catalog as a result of specification of a nonterminal node point, this parameter is not effective.)

**GEN=nn**

Number of generations to maintain for the file.

**MEM**

Used to fill a gap between members of a generation file. A full device assignment set must be used with the CAT statement if MEM is specified.

**NOTE:**

*A CAT statement with MEM specified and a DECAT statement cannot be used against the same member of a generation file in the same job.*

## CC

### Function:

Allows you to issue OS/3 system commands from within a job control stream. Parallels the functions of OS/3 system console and workstation commands.

### Format:

```
//[ symbol] CC {command  
                'command and parameters' }
```

### Parameters:

#### command

A single system console or workstation command.

#### 'command and parameters'

System console or workstation command and associated parameters; total number of characters enclosed in quotes cannot exceed 60.

### NOTES:

1. When the command string contains no blanks (other than the blank separating the command from its first parameter), you can precede the first parameter with a comma instead of enclosing the command and its parameters in quotes. For example:

```
// CC BE,JOB1 instead of // CC 'BE JOB1'
```

2. The following system console commands may not be specified in the CC job control statement: MIX, SWITCH, AVR, REBUILD, SHUTDOWN, SYSDUMP, and all SET commands.
3. Unsolicited input messages (see the interactive services commands and facilities user guide/programmer reference) and // PAUSE responses cannot be specified in the // CC statement.

See your operations handbook for a description of system console commands. See the interactive services commands and facilities user guide/programmer reference for a description of workstation commands.

**CR**

## Function:

Temporarily inserts embedded data or other job control statements from cards, data-set-label diskette, or spool file into a stored control stream. This statement can be used in control streams or procedures filed in \$Y\$JCS or an alternate library. The CR control statement is placed in the control stream at the point where data is to be inserted. The data from the input reader is merged with the filed control stream, and the combined control stream is examined by the job control routines. The data read from the card reader must be terminated by a FIN control statement.

Any number of CR control statements may appear in a control stream, provided the FIN control statement for a previous CR statement is encountered before the current CR control statement.

## Format:

```
//[ symbol] CR
```

The CR statement contains no parameters.

## DATA FILEID

### Function:

A control statement used by the input reader. It identifies and must precede any input card data that's going to be spooled as the result of the IN operator command.

### Format:

```
// DATA FILEID=file-identifier[ ,RETAIN][ ,IGNORE]
```

### Parameters:

#### FILEID=file-identifier

File identifier. Agrees with either the *file identifier* of the LBL control statement for the file or a concatenation of the *jobname* from the JOB control statement and the *filename* from the LFD control statement.

#### RETAIN

Used to maintain the spool file after the job has processed the file. The only means of deleting the reader file after RETAIN has been specified is by issuing the operator console command DE SPL,RDR.

#### IGNORE

Specifies that // RUN/RV statements in the card deck to be spooled are to be processed as data cards.

### NOTE:

→ If the LBL control statement is used in the device assignment set for a spooled card file, then the file-identifier of the DATA control statement must be the same as the file identifier of the LBL control statement.



## DATA STEP

### Function:

A control statement that identifies replacement embedded data for a saved translated job control stream. It must precede the replacement data submitted from a card reader, data-set-label diskette, or input spool file.

### Format:

```
// DATA STEP=nnn
```

### Parameters:

nnn

A decimal number in the range 1 to 255 that specifies the number of the job step within the job for which you are submitting new embedded data.

### NOTES:

1. *A DATA STEP statement must be submitted for each job step that contains embedded data you want to replace. You must replace the old data sets in the specified job step with an equal number of new data sets.*
2. *Because the replacement embedded data is submitted to the saved translated stream from an input device, you must use the SI command or the // CC SI job control statement to initiate the job stream run.*
3. *The embedded data is replaced for that job run only.*

**DD**

## Function:

Provides a means of changing certain data definitions at run time. The DD statement must appear within the DVC and LFD sequence. If the file is a cataloged file, it must follow the LBL statement.

```
//[symbol] DD [ {RCFM} = {FIXBLK } [ {BKSZ} =n ] [ {RCSZ} =n ]
                {RCFMn} {FIXUNB } [ {BKSZn} ] [ {RCSZn} ]
                {UNDEF }
                {VARBLK }
                {VARUNB }
                [ {LACE} =n ] [ {SIZE} =n ] [ {UOS} =n ]
                {LACEn} {SIZE n } {AUTO} ] [ {UOS } =n ]
                [ {KLEN} =n ] [ {KLOC} =n ] [ , INDS=n ]
                {KLEn} ] [ {KLOCn} ]
                [ , ACCESS= {EXC } [ , REWIND= {NORWD }
                {EXCR } [ {UNLOAD} ]
                {SRDO }
                {SRD }
                {SADD} ]
                [ , OPRW=NORWD ] [ , CLRW= {NORWD} ] [ , FILABL= {NO }
                {RWD} ] [ {NSTD }
                {STD} ]
                [ , TPMARK=NO ] [ , RECV= {YES } ] [ , VSEC= {YES } ] [ , VMNT= {ONE }
                {FCE} ] [ {n } ] [ {NO } ]
                [ , RCB= {NO } ] [ , OFFSET=1 ]
                {YES} ]
```

→

→

**DD****Parameters:**

The n that is suffixed to some keywords (e.g., RCFMn) refers either to partition n of a multipartition SAT or NI disk file or to KEYn of a multikey MIRAM disk file. The definitions for keyword parameters associated with each file type are found in the data management user guide. The definitions for the keyword parameters associated with SAT files are found in the supervisor user guide. Tables in the job control user guide provide a summary of the allowable keywords for each file type.

**NOTES:**

1. *Extreme care must be used in specifying the BKSZ/BKSZn or RCSZ/RCSZn keyword parameters; see data management user guide for a complete description of these keywords.*
2. *OFFSET=1 is used under consolidated data management for conversion purposes (IBM System/34 to OS/3). It indicates that a data-set-label diskette was created by an IBM utility and has a file header as the first physical record. // DD OFFSET=1 allows processing of the diskette to begin at the first logical record (the second physical record).*



## DECAT

### Function:

Removes a file from the catalog. The file is defined by the `lfname` parameter in this statement and in the LFD job control statement.

### Format:

```
//[symbol] DECAT lfname[,catpw][,SCR] [ , { GEN }  
                                     { ROL } ]
```

### Parameters:

#### `lfname`

File name; must agree with the file name of the associated LFD job control statement.

#### `catpw`

Password of one to six alphanumeric characters for the catalog itself; must be specified if the catalog itself is password protected.

#### *NOTE:*

*Even if all the files in a file catalog have been decataloged (using // DECAT), the catalog password remains in effect. Catalog passwords are established, removed, and changed through the JC\$CAT routine, which is explained in the file cataloging concepts and facilities manual. When you use // CAT or // DECAT, you must specify catpw if a password has been established through the JC\$CAT routine.*

#### `SCR`

Scratches the file when it's removed from the catalog. (See the CAT job control statement.)

#### `GEN`

Removes all generations of the file from the catalog.

#### `ROL`

Used to reset a generation file's status to the condition that existed before the current generation was added. *ROL* (rollback) removes the current member of a generation file, making the previous member current, and inserts the rolled-back file as the oldest member of the generation file.

#### *NOTE:*

*ROL can only be used against the current member of a generation file, and can only apply once to the current member within a single job.*

**DST**

## Function:

Sends spooled out (print or punch) to RBP (remote batch processing) terminals. This statement must appear in the device assignment set for the print or punch file.

## Format:

```
//[ symbol] DST dest-1[ ,dest-2, ...,dest-16]
```

## Parameters:

**dest**

Destination identifier of one to six alphanumeric characters; defined by RBP. Specify OS3CTR or CENTRAL to specify the local site's control printer.

**NOTE:**

*Although the DST statement can be used in DDP-initiated jobs, DDP or auxiliary printer output (specified by // ROUTE) and RBP output (specified by // DST) cannot be mixed for any one job. For any job, all output must be of one type or the other.*

# DVC

## Function:

Requests the assignment of a peripheral device to a job. The device must be available before the job can be scheduled and is allocated by job control at job step time. Specific devices can be requested and assigned.

This statement must be the first in a device assignment set, and one set is required for each file. A device assignment set consists of the DVC, VOL, EXT, LBL, and LFD statements. At least one DVC statement and one LFD statement are required for each file accessed by the user program. The EXT and LBL statements are optional. The VOL statement is required for tape and disk files. For catalog files, the EXT statement must follow the LBL statement.

→ The maximum number of unique devices allowed in a job is 255. The maximum number of unit record devices (e.g., card readers, data-set-label diskettes, printers) allowed in one job is 42.

## Format:

→ `//[ symbol] DVC { nnn[ (n)]  
RES  
RUN } [ , addr  
ALT  
OPT  
IGNORE  
I  
O  
REQ[ (n)]  
REAL ] [ , HOST=host-id]`

## Parameters:

### nnn[ (n)]

Decimal logical unit number specifying a device type and features from the system's logical unit table (Table A-1). When a job is scheduled, each logical unit number is assigned to a specific device. The device type and features for a logical unit number may be changed with the EQU job control statement.

→ When the device is a workstation, (n) is used to specify the number of workstations that can access a single workstation file. A maximum of 255 workstations of the type specified by nnn can be associated with one file.

### NOTES:

1. If more than one volume is assigned the same logical unit number, the operator may be required to demount the first volume and then mount the next volume on the same device. All volumes but the last are nonshareable with other jobs because they must be demounted.
2. If a single volume is assigned more than one logical unit number in different job steps, the operator may be required to demount the volume from the device assigned to the first logical unit number and then mount the volume on the device assigned as the next logical unit number. The volume is not shareable with other jobs until it is mounted on the last device.

**DVC**

3. *Before a mount is requested, the system tests to determine if the required volumes are mounted on other devices.*
4. *You cannot use logical unit numbers 50 through 59 on the DVC statement you use to catalog a disk file. These logical unit numbers specify "any type" of disk. To ensure that cataloged files obtain a valid device when a job is scheduled, you must use logical unit numbers that specify a particular kind of device. For example, you could specify // DVC 60 to indicate that an 8416 disk is to be used.*

**RES**

Use system-resident device. The VOL statement is not required when RES is specified and will be ignored if present.

**RUN**

Use device containing the job's \$Y\$RUN file. The VOL statement is not required when RUN is specified and will be ignored if present.

**addr**

Physical address of the device, in hexadecimal. This represents the channel number, the control unit address, and the device number. The use of this parameter inhibits dynamic allocation of devices.

**NOTE:**

*The addr parameter may be used to specify a real device in a spooling system. Real devices and virtual devices (of the same type) may not be assigned in the same job.*

**ALT**

Alternate devices of the same type and features may be assigned (within a single job step). If an alternate device is not available, the job will execute with a single device.

**OPT**

Requested device is not essential to the running of the job.

**IGNORE**

More than one logical file may be assigned to a physical device. Use this parameter when different files are used on one device in a serial manner by a job step.

**I**

Indicates input spooling was configured for the diskettes at SYSGEN time.

**O**

Indicates output spooling was configured for the diskettes at SYSGEN time.

**HOST=host - id**

Indicates that a disk file is located at a remote DDP host. The host-id is one to four alphanumeric characters long and identical to the label-id of the LOCAP macroinstruction in your ICAM network. You can specify \$HOST instead of a host-id to indicate a remote originator (the host that initiated the job). The specified host-id or \$HOST must always be for a remote host; otherwise, a data management error will result.

## DVC

### REQ[ (n) ]

Used for workstations. When a number is specified for (n), this parameter indicates how many workstations specified in *nnn[(n)]* must be connected before the job is run. If (n) is omitted in this parameter, all workstations specified in *nnn[(n)]* are required.

### REAL

Used to assign a real device to a job, without having to specify the device's physical address. This parameter can be used to bypass spooling.



**DVC PROG** ↓

## Function:

Allows a BAL program (using consolidated data management macros) to communicate with another BAL program via DDP's program-to-program facility; can be specified only once in any single job step. Used in place of // DVC when the device assignment set is for a program-to-program type of file; the device assignment set may include an LBL statement but must include an LFD statement.

## Format:

```
//[symbol] DVC PROG [,jobname][,HOST=host-id]
```

## Parameters:

**jobname**

Any 1- to 6-character job name. Identifies the other participant in the program-to-program communication. (When specified in // DVC PROG for the primary, for example, it identifies the surrogate. When specified in // DVC PROG for the surrogate, it identifies the primary.) This parameter is required in the // DVC PROG statement for the primary but is optional for the surrogate.

**HOST=host-id**

Specifies a particular host in a DDP network. The host-id is one to four alphanumeric characters long and identical to the label-id of the LOCAP macroinstruction in your ICAM network. Use \$HOST to indicate the originator (the host that initiated the job).

 ↑

## EQU

### Function:

Equates a logical unit number specified in the control stream with a device type in your system. This enables you to run on a system other than the one the control stream was designed and generated for. The EQU statement must be used to assign additional logical unit numbers to virtual readers, printers, and punches.

The EQU statement must precede the device assignment set in the control stream that references the devices and is effective for the entire job.

### Format:

```
//[ symbol] EQU lun-1,type-1[ ,lun-2,type-2,...,lun-n,type-n]
```

### Parameters:

#### lun-n

Logical unit number specified in the control stream. The valid logical unit numbers are shown in Appendix A.

#### type-n

Code of four to eight hexadecimal characters providing the exact device characteristics (Appendix A).

**EXEC****Function:**

Identifies program to be executed. The EXEC control statement is a job step delimiter and is the last control statement processed by job control before execution of the program named in the statement. Any PARAM control statements included in the control stream must immediately follow an EXEC statement.

The EXEC control statement loads the specific program from either `$$LOD`, the job's `$$RUN` file, or from an alternate load library previously defined by a device assignment set.

Before actual program loading, job control completes any pending tape and disk mounting requests. The specified program is then loaded, overlaying job control, and receives control.

**Format:**

```
//[symbol] EXEC program-name [ , { library-name } ] [ , [ ± ] switch-priority ] [ , ABNORM=label ]
```

{
}

\$Y\$RUN  
\$Y\$LOD

**Parameters:****program-name**

1- to 6-character name identifying the program to be executed. It must be the same as the load module name specified to, or generated by, the linkage editor. The name is left-justified in the field and zero filled to the right, if necessary.

**library-name**

LFD name of the previously defined library file (on disk) containing the load module. Use this parameter when the program must be loaded from an alternate load library. If the program cannot be found, `$$LOD` and the job's `$$RUN` file, in that order, are searched.

**\$\$RUN**

Load user program from the job's `$$RUN` file. If the program cannot be found, `$$LOD` is searched for the program.

If positional parameter 2 is omitted, `$$LOD` is searched first and if the program cannot be found, the job's `$$RUN` file is searched.

**[ ± ]switch-priority**

Decimal number specifying the programs task switching priority; used by the supervisor for dispatching control during program execution. The number specified can be an absolute value ranging from 1 to 60 or a relative value such as +3 or -3. With absolute values, the lower number represents the higher priority. (The highest priority is 1.) Relative values increment or decrement the overall absolute task switching priority set for a job via the SWITCH operator command or // OPTION PRI. A positive value (+) *decrements* the overall priority resulting in a *higher* task switching priority for that program. A negative priority *increments* the overall priority resulting in a *lower* task switching priority for the program. If *switch-priority* is omitted, the lowest priority in the system is used. Task switching is explained in the supervisor user guide.



## EXEC

### ABNORM=label

Used to bypass job control statements in case the program named terminates abnormally. The *label* corresponds to the label specified in the optional label (symbol) field of a subsequent job control statement. Should abnormal termination occur, job control skips forward to this statement. Since ABNORM=label is a keyword parameter it may be coded in any position. For example:

```
// EXEC MYPROG, ABNORM=ERR
```

# EXT

**Function:**

Obtains disk or diskette space. Provides information needed to create new files or extend existing files on disks or diskettes.

Any number of extents may be specified on a single EXT statement or on several EXT statements. The first EXT statement in a DVC-LFD sequence applies to the first volume specified on the immediately preceding VOL statement. If multiple extents are specified for disk or format-label diskette files, the specifications given for the first four positional parameters apply to all extents specified on that statement. When allocating catalog files that require an EXT statement, // EXT must follow // LBL in the device assignment set.

See Appendix B for further information regarding extent specification.

**Format 1 (Series 90 Disk):**

```
//[ symbol] EXT { DA } [ { C } ] [ { inc } ]
                { IS } [ { CF } ] [ { 0 } ]
                { IR } [ { F } ] [ { 1 } ]
                { NI }
                { MI }
                { SQ }
                { ST }
                [ { addr } ] [ { mi } ] [ { mj } ] , ... [ , OLD ]
                { Tccc:hh } [ { (bi, ai) } ] [ { (bj, aj) } ]
                { BLK }
                { TBLK }
                { CYL }
                { TRK }
                { OLD }
```

**Format 2 (System 80 Disk):**

```
//[ symbol] EXT { MI } [ { C } ] [ { inc } ] [ { addr } ] [ { mi } ]
                { ST } [ { CF } ] [ { 0 } ] [ { Tccc:hh } ] [ { (bi, ai) } ]
                [ { F } ] [ { 1 } ] [ { BLK } ]
                [ { TBLK } ]
                [ { CYL } ]
                [ { TRK } ]
                [ { OLD } ]
                [ { mj } ] , ... [ , OLD ] [ , FIX ]
                [ { (bj, aj) } ]
```

## EXT

Format 3 (Series 90 Data-Set-Label Diskette):

```
//[ symbol] EXT SQ,C,Ø,BLK,(bi,ai)
```

Format 4 (System 80 Data-Set-Label Diskette):

```
//[ symbol] EXT MI,C,Ø,BLK,(bi,ai)[,NDI]
```

**NOTE:**

*In System 80, all the parameters (except FIX) that apply to disk also apply to format-label diskette.*

**Parameters:**

**DA**

Series 90 only. Direct access file.

**IS**

Series 90 only. ISAM file.

**IR**

Series 90 only. IRAM file.

**MI**

MIRAM file; only valid specification for System 80 data-set-label diskette.

**NOTE:**

*In System 80, data files must be allocated as MIRAM files.*

**NI**

Series 90 only. Nonindexed (DA or SQ) file.

**SQ**

Series 90 only. Sequential file; only valid specification for a data-set-label diskette file.

**ST**

System access technique (SAT) file.

**C**

Allocates contiguous space; only valid specification for a data-set-label diskette file.

**F**

Format the file at allocation time. Positional parameter 4 must be *BLK*.

**CF**

Both of the preceding.

**EXT****inc**

Specifies the secondary increment in cylinders by which the file is to be extended if automatic extension is required. If no EXT statement is in this file, the value of the most recently specified secondary increment for this file is used, which could be the very first time this file was used.

**0**

File cannot be dynamically extended; must be specified for a data-set-label diskette file.

**addr**

Absolute cylinder address where the file is to begin; considered hexadecimal if you code *X'number* or *number*; considered decimal if you code *D'number*. The allocation is in terms of cylinders.

**Tccc:hh**

Absolute track address in cylinder/head format where the file is to begin; considered hexadecimal if you code *X'number* or *number*; considered decimal if you code *D'number*. Allocation is in terms of tracks.

**BLK**

Allocates space in blocks; actual allocation is in terms of cylinders. This is a request for a new extent; only valid specification for data-set-label diskette files.

**TBLK**

Allocates space in blocks; actual allocation is in terms of tracks.

**CYL**

Allocates space in cylinders.

**TRK**

Allocates space in tracks.

**OLD**

Indicates parameter 3 is being used to change the secondary increment (the automatic allocation amount for dynamic extension) of the extent for a previously allocated file.

Cannot be followed by any other parameters if specified.

**NOTE:**

You cannot allocate a disk or format-label diskette file by track (using TRK, TBLK, or Tccc:hh) when creating any of the following: ISAM files, indexed IRAM files, or indexed IRAM characteristic MIRAM files.

**mi**

Number of cylinders or tracks allocated for this file; positional parameter 4 must be *CYL*, *addr*, *TRK*, or *TCCC:hh*.

## EXT

(bi,ai)

Used when allocation is in terms of blocks (by cylinder or track) for disk and diskette files; positional parameter 4 must be *BLK* or *TBLK*.

bi

Is the average block length.

ai

Is the number of blocks.

### NOTE:

*The mj parameter means the same as the mi parameter and the (bj,aj) parameter means the same as the (bi,ai) parameter. The only difference is that mj and (bj,aj) are used for additional extents in the file and are never specified for data-set-label diskette files.*

### OLD

Indicates the file's previously allocated extent is to be increased by the allocation amount specified (*mi*, (*bi,ai*), etc). If *OLD* is omitted, the request is for a new extent.

### FIX

System 80 only. Indicates you're allocating the extent in the fixed-head area of an 8417 disk. Note that you can specify *FIX* and *OLD* in any order so that if you omit *OLD* there's no need to retain the comma.

### NDI

System 80 only. *NDI* (nondata interchange) must be specified to allocate space for all data-set-label diskettes that are not basic data exchange (BDE) diskettes. If omitted, a BDE diskette (a single-sided, single-density diskette having 128-byte sectors, 26 sectors per track, and 73 tracks) is assumed.



**FIN****Function:**

Terminates card reader operations. Used as a sentinel card to signal the end of card input to the run symbiont. If // FIN is used to signal the termination of card reader operations when a job is being read, it must be preceded by a /& control statement.

This statement is also used to terminate card reader operation initiated by a FILE operator command when complete control streams or procedure definitions are filed or by an IN operator command when card input is being spooled.

**Format:**

//[ symbol] FIN

There are no parameters associated with this statement.

## FREE

### Function:

Releases volumes and peripheral devices assigned to a job. Released volumes and devices, including related alternate devices, are returned to the pool of unallocated system resources and are available to fulfill requirements later in the job or for subsequent assignment to other jobs unless required by later job steps.

### Format:

```
//[ symbol] FREE lfdname-1[ (DEV)] ,... ,lfdname-n[ (DEV)]
```

### Parameters:

#### lfdname

File name previously defined in a DVC-LFD control statement sequence.

#### (DEV)

The device and volume containing the file are released.

### NOTE:

*You should always specify the (DEV) parameter even though it's shown as optional. Additionally, you must specify (DEV) to free unit record devices such as card readers and punches, printers, and workstations.*

**GBL**

## Function:

Global status is given to the named set symbols. This statement may appear anywhere in the job stream, and the symbols are global from the point of encounter forward. When defining a set symbol, an ampersand is not to be included on the global declaration. In the run processor, an ampersand always means substitution and is not part of the symbol name.



## Format:

```
//[symbol] GBL set-id-1[=init-1] [,set-id-2[=init-2],...,set-id-n[=init-n]]
```



## Parameters:

**set-id-1 through set-id-n**

Name of the set symbol that is to have global status.

**=init-1 through =init-n**

Value given to *set-id* if the variable has not been previously defined as being global. If previously defined, *init* is ignored.

*NOTE:*

*Whenever a quoted value is assigned to a set symbol (using // GBL), the quotes are considered part of the value. (For example, if // GBL X='ABC', then &X is 'ABC'.)*



## GO

### Function:

Causes an unconditional branch, in a forward direction, to another control statement. GO is acted upon during run processor time (prior to the job's execution), unlike the SKIP job control statement, which is acted upon at execution time.

### Format:

```
//[symbol] GO destination
```

### Label

#### symbol

Identifier that is used if this control statement is to be the target of another GO or IF control statement.

### Parameter:

#### destination

Identifier of the target control statement that is to receive control; must agree with the name in the label field of the receiving control statement.

**IF**

## Function:

Causes a conditional branch, in a forward direction, to another control statement. IF is acted upon during run processor time (prior to the job's execution), unlike the SKIP job control statement, which is acted upon at execution time.

## Format:

```
//[symbol] IF (a op b)destination
```

## Label:

**symbol**

Identifier that is used if this control statement is to be the target of a preceding GO or IF control statement.

## Parameters:

**a op b**

Conditional branch test values:

**a and b**

Are the operands to be compared; can be either alphabetic or numeric; (a run processor error results if one is alphabetic and the other numeric). If a numeric comparison is done and neither operand is numeric, both the greater than and less than conditions are set, resulting in all conditions except equal being allowed to branch. If a character comparison is being done and the two operands are not the same length, the comparison is made on the number of characters, rather than on the contents of the operands.

**op**

Relational operator:

<b>EQ</b>	<i>a equal to b</i>
<b>NE</b>	<i>a not equal to b</i>
<b>GT</b>	<i>a greater than b</i>
<b>LT</b>	<i>a less than b</i>
<b>GE</b>	<i>a greater than or equal to b</i>
<b>LE</b>	<i>a less than or equal to b</i>

Comparands and the relational operator are separated by spaces.

**destination**

Identifier of the target control statement that is to receive control if the transfer condition is true. This must agree with the name in the label field of the receiving control statement.

**NOTES:**

1. *If the operands to be compared by an IF statement contain embedded blanks, you must use //JSET to replace the values before you can compare the operands.*
2. *Whenever you enclose an operand in quotes, the quotes are considered part of the operand. For example, the two operands in ('a' EQ a) are not considered equal.*

  
**INQ JOB****Function:**

Through the use of symbols, allows you to examine job related values and to determine the availability of certain facilities.

**Format:**


```
//symbol INQ JOB,keyword
```

**Label:**

```
symbol  
Symbol name
```

**Parameter:****keyword**

One of the following keywords that assigns the specified value to *symbol*:

- |      |  |
|------|--|
| NAME | Assigns the job name.  |
| ORI  | Assigns the user-id of the originator.   |
| HOST | Assigns the host-id of the originator (null value if none).  |
| ORID | Assigns the device-id of the originator (if a local workstation).  |
| WKS  | Assigns 0 if a workstation is not initiated for the job.<br>Assigns 1 if a workstation is initiated for the job. |
| DDP  | Assigns 0 if remote DDP is not initiated.<br>Assigns 1 if remote DDP is initiated.                               |
- 



## JNOTE

↓  
Function:

Displays a message at the system console or specific workstations. JNOTE is acted upon by the run processor, before a job is put into execution.

Format:

```
//[symbol] JNOTE comment-line [,destination-1,...,destination-n]
```

Parameter:

**comment-line**

Comment or message to be displayed; may contain up to 60 characters. If the message contains embedded blanks, the slash character, or commas, the line must be enclosed in single quotes.

**destination (where destination=[host-id:]user-id)**

**host-id**

In a DDP environment, directs the message to a particular host; is one to four alphanumeric characters and is identical to the label-id of the LOCAP macroinstruction in your ICAM network. You can specify \$HOST to indicate the originator/master host. The host-id is optional but, if specified, must be followed by a user-id. If host-id is omitted, the local host is assumed.

**user-id**

Directs the message to a particular workstation; can be any 1- to 6-alphanumeric-character workstation user-id, the keyword OPERATOR or \$Y\$CON (denoting the console workstation), or \$Y\$ORI (denoting the job's originator workstation). \$Y\$ORI is the default.

If you omit a destination, the message goes to the originator workstation. A message sent to an originator workstation that is not logged on is rerouted to the console. No other messages are rerouted.

↑  
If the system does not have workstations or DDP, messages go to the system console.



**JOB**

## Function:

Identifies the job and indicates the beginning of control information for the job. The associated job's \$Y\$RUN file is given the same name.

## Format:

```
//[symbol] JOB jobname [ , { P } ] [ , min ] [ , max ] [ , { tasks } ] [ , { max-time } ]
                               [ , { H } ]
                               [ , { 1 } ]
                               [ , { SUP } ]
                               [ , print-option-list ] [ , acct-no ] [ , nXm ]
                               [ , { ACT } ] [ , { NOHDR } ]
                               [ , { LOG } ] [ , { HDR } ]
                               [ , { NOACT } ]
                               [ , { NOLOG } ]
                               [ , { NONE } ]
                               [ , { BOTH } ]
```

## Parameters:

## jobname

One to eight alphanumeric characters specifying the job identifier. This name must be used to reference the control stream after it has been filed in \$Y\$JCS.

## P

Preemptive job selection priority. If specified on a system that does not support rollin/rollout, the priority is converted to high (H).

## H

High job selection priority.

## N

Normal job selection priority.

## min

Minimum number of main storage bytes required to execute the largest job step of the job; may be specified in hexadecimal (by coding the number or *X'number*) or in decimal (by coding *D'number*). If the number is not preceded by an X' or a D', it is considered hexadecimal. 2000 hexadecimal (8K decimal) is the minimum amount that can be assigned; it does not include the job prologue. Required when the programs being executed are not in \$Y\$LOD, in an alternate load library on SYSRES, or on the volume containing the job's \$Y\$RUN file.

**NOTE:**

*If you are compiling (or assembling), link editing, and executing your program in a single run, job control cannot determine how much main storage is required for the execution of your program. If your generated load module requires more main storage than the compiler (the largest known job step), insufficient main storage will be allocated. If possible, you should indicate how much main storage is needed for your load module.*

## JOB

### max

Maximum number of main storage bytes requested, but not required, to execute the largest job step on the job. May be specified in hexadecimal (by coding the number or *X'number*) or in decimal (by coding *D'number*). If the number is not preceded by an X' or a D', it is considered hexadecimal.

If either *min* or *max* is omitted, the value specified for one is assumed for the other. Only that space actually needed by the largest program in the job is specified. Job control adds additional storage for the prologue, as required.

If both *min* and *max* are omitted, the length of the largest load module needed by the job is used. In most cases, omitting *min* and *max* gives the best use of main storage. Exceptions are programs that use additional main storage to improve performance (such as sort/merge), those cases where the user loads phases to locations other than addresses assigned by the linkage editor, or when programs to be executed are not on the SYSRES volume or the volume containing the job's \$Y\$RUN file at run time.

### tasks

Maximum number of tasks that can be active simultaneously in any job step. Each task specified requires 256 bytes in the job prologue so that the maximum number of tasks you can specify is limited by the size of the prologue (65,535 bytes).

### max-time

Maximum number of minutes this job should take for execution.

If *max-time* is omitted, the default value specified at system generation (SYSGEN) time is used. If your job executes beyond this limit (specified or default), the operator is queried as to whether the job should continue or be terminated.

If a timer service is not specified at SYSGEN time, the job continues processing until normal or abnormal termination occurs, whether or not *max-time* is specified in job control.

The specified *max-time* limit is adjusted to allow for the following conditions: checkpoint/restart, PAUSE job control statements, SET CLOCK commands, and rollout/rollin.

### SUP

Suppresses the *max-time* function completely for a particular job.

### print-option-list

A list of one or more options used to control the printing of job control statements on the job log (in a spooling environment) or their display on the system console (in a nonspooling environment).

<b>BASIC</b>	Basic job control statements with substitution (This is the default in a spooling system.)
PROC	Expanded procs
EMB	Embedded data
DEBUG	Job control statements before substitution

**JOB**

SKIP	Statements skipped as a result of an IF or GO directive
WARNING	Job control warning messages are not displayed on the console or workstation
ALL	All options in effect except WARNING
<b>NONE</b>	No options in effect (This is the default in a nonspooling system.)

Only the first character need be specified for each option. BASIC (B) is automatically in effect whenever P, E, D, or S is chosen.

If more than one option is given, they must be separated by commas and enclosed in parentheses.

**acct-no**

Job account number of one to four alphanumeric characters.

**nXm**

Buffer pool size. Job log and spooled files not having reserved buffers use this pool.

n

Number of buffers.

X

Constant.

m

The number of 256-byte blocks per buffer. The only values accepted for m are 1, 2, 4, 8, 16, and 32. Numbers larger than 32 default to 32; numbers outside the acceptable range are changed to the lower acceptable constant (e.g., 6 is changed to 4).

If omitted, one 256-byte buffer is assumed if only job log is used. If other files are also sharing the pool, two buffers (2x2) are assumed.

**ACT**

Forces the printing of accounting records, regardless of what system options are in effect.

**LOG**

Forces the printing of job log records, regardless of what system options are in effect.

**NOACT**

Suppresses the printing of accounting records from the job log file.

**NOLOG**

Suppresses the printing of log information from the job log file (including main storage dumps).

**NONE**

Suppresses the printing of both accounting records and log information from the job log file.

## JOB

**BOTH**

Allows the printing of accounting records and job log information.

**NOHDR**

Suppresses the printing of page headers in burst mode.

**HDR**

Allows page headers to be printed.

**NOTE:**

*The user should consult the parameters used for system generation to ensure that a conflict does not arise when submitting these parameters.*

## JSET

### Function:

Used to declare a local set symbol (local set symbols can be declared and referenced only within a jproc) or to change the value of a global set symbol without changing its status to local. ←

### Format:

//symbol JSET value ←

### Label:

symbol  
Set symbol name.

### Parameter:

#### value

Value assigned to the symbol. May be a character string enclosed within apostrophes if it contains embedded blanks, may be a 2-term expression that is evaluated and converted to a character string before it is assigned, or may be used to establish a null character (blank). The operators allowed are +, -, \*, /, ++, --, \*\*, and //. Leading zeros are not maintained for multiple-digit numeric values in a JSET job control statement. If a leading zero is required when a symbol is used, it must be created by another JSET statement.

#### NOTES:

1. When using a JSET statement to perform an operation, the operands to be acted upon must be numeric.
2. When specifying a quoted value with // JSET, one level of quotes is always removed from the value. (For example, if //X JSET 'ABC', then &X is ABC.) ←

## LBL

### Function:

Supplies label information for files on disk, diskette, or tape volumes for use by data management. There may be only one LBL statement in a device assignment set.

### Format:

```

//[symbol] LBL {file-identifier} [,{file-serial-number}] [,expiration-date]
              {'file-identifier'} [VCHECK
              [,creation-date] [,{file-sequence-number}] [,{generation-number}]
              [,{version-number}]
  
```

### Parameters:

#### file-identifier

Name that identifies the file; maximum of 17 characters for tape files and 44 for disk files. This name corresponds to the file label. If prefixed by \$SCR, the file is temporary and deleted at the completion of the job step. If prefixed by \$JOB, the file is temporary and deleted at the completion of the job. If prefixed by \$LOKnn (for disk files), the file is lockable.

#### 'file-identifier'

Name that contains embedded blanks.

#### file-serial-number

One to six alphanumeric characters specifying the file serial number. Identical to the volume serial number of the first volume of the file.

#### VCHECK

A file serial number is to be created on output and checked on input.

If positional parameter 2 is omitted, the tape does not have a VOL1 label, or there is no volume-serial-number/file-serial-number relationship for disk.

#### expiration-date

File expiration date; may take either of two forms:

- yyddd

yy

Is the year.

ddd

Is the day of the year.

**LBL**

## ■ Rddd

R

Indicates that the file is to be retained.

ddd

Is the number of days (1-9999) that the output file is to be saved beyond the creation date.

If you omit the *expiration date* and allocate then write to the file (in the same job step), the current system date is used. If omitted and file is only being allocated, no date is specified and zeros are inserted.

**creation-date**

File creation date; written in form yyddd:

yy

Is the year.

ddd

Is the day of the year.

If *creation-date* is omitted for a tape file, the date stored in either the job prologue or the system information block is used. For a disk output file, the date stored in the job prologue is used; for a disk input file, the field is ignored.

**file-sequence-number**

At file creation, assigns a sequence number to the file indicating its position with respect to the first file within a multifile set; should be specified in subsequent // LBL statements to indicate the file's position. If omitted on output, data management assigns 1 as the sequence number regardless of the file's position. If omitted on input, data management does not check for sequence numbers but expects to use the first file encountered. This parameter applies only to standard labeled tape files.

**generation-number**

Unique edition of a file; applies only to tape files.

**version-number**

Version of a generation of a file; applies only to tape files.

**LBL**

## File Catalog

## Function:

Provides information for cataloging files. Allows the maintaining of multiple generations of a file and protects against unauthorized access or update by using read/write passwords. Only disk and tape files may be cataloged.

## Format:

```

//[ symbol] LBL { [ qual/ ] level-id-1 [ .level-id-2... [ .level-id-n ] [ (nn) [ (rpw/wpw) ] ] ] }
                { ' [ qual/ ] level-id-1 [ .level-id-2... [ .level-id-n ] [ (nn) [ (rpw/wpw) ] ] ] ' }
                [ , { file-serial-number } ] [ , expiration-date ] [ , creation-date ]
                [ , { VCHECK } ]
                [ , { file-sequence-number } ] [ , { generation-number } ]
                [ , { 1 } ]
                [ , { version-number } ]
                [ , { 1 } ]

```

## Parameters:

```

[ qual/ ] level-id-1 [ .level-id-2... [ .level-id-n ] [ (nn) [ (rpw/wpw) ] ] ]
                { (nn)
                  +n
                  -n
                }

```

Specifies the file catalog information; limited to 17 characters for tape files and 44 characters for disk files. If the level names include blanks, the parameter must have leading and trailing apostrophes.

**qual/**

File qualifier of one to eight characters. This removes a file from general use and places it in the category of restricted use. The slash is coded as part of qualifier.

**.level-id-1 through .level-id-n**

Level names for the file; each level must be separated by a period.

**nn**

Absolute reference number of a file generation.

**+n**

Creation number of a new file generation; the plus sign must be coded.

**-n**

Relative reference number of an old member of a generation in relation to the current member. The minus sign must be coded.



**LBL**  
File Catalog**(rpw/wpw)**

Read and write passwords of one to six alphanumeric characters; must be separated by a slash and enclosed by parentheses.

**file-serial-number**

One to six alphanumeric characters specifying the file serial number. Identical to the volume serial number of the first volume of the file.

**VCHECK**

The volume-serial-number/file-serial-number relationship is to be checked on input and created on output by data management.

If positional parameter 2 is omitted, the tape does not have a VOL1 label, or there is no volume-serial-number/file-serial-number relationship for disk.

**expiration-date**

File expiration date; may take either of two forms:

■ **yyddd****yy**

Is the year.

**ddd**

Is the day of the year.

■ **Rddd****R**

Indicates that the file is to be retained.

**ddd**

Is the number of days (1-9999) that the output file is to be saved beyond the creation date.

If you omit the *expiration-date* and allocate then write to the file (in the same job step), the current system date is used. If omitted and file is only being allocated, no date is specified and zeros are inserted. ←

**creation-date**

File creation date; written in form yyddd:

**yy**

Is the year.

**ddd**

Is the day of the year.

If *creation-date* is omitted for a tape file, the date stored in either the job prologue or the system information block is used. For a disk output file, the date stored in the job prologue is used; for a disk input file, the field is ignored.

**LBL**

## File Catalog

↓  
**file-sequence-number**

At file creation time, assigns a sequence number to the file indicating its position with respect to the first file within a multifile set; should be specified in subsequent // LBL statements to indicate the file's position. If omitted on output, data management assigns 1 as the sequence number regardless of the file's position. If omitted on input, data management does not check for sequence numbers but expects to use the first file encountered. This parameter applies only to standard labeled tape files.

↑  
**generation-number**

Unique edition of a file; applies only to tape files.

**version-number**

Version of a generation of a file; applies only to tape files.

**NOTES:**

1. *The logical unit number specified on the DVC control statement when a file is cataloged must not be of the general type (50-59). This ensures that cataloged files will obtain a valid device whether or not the required volume is mounted when the job is scheduled.*
2. *The plus and minus are only used to delimit relative generation references. Use elsewhere results in errors.*
3. *When cataloging a new member to a generation that will cause the maximum number of generations to be exceeded, the device type and volume serial number will be taken from the catalog entry for the oldest member of the generation. The oldest member is also automatically decataloged. This allows for the cycling of three tapes for the three current generations.*

## LCB

## Function:

Overrides the system default load code buffer for a print file; becomes effective when the file is opened. During interjob-step processing, the SYSGEN default value is used. If used, the LCB control statement must be within the DVC-LFD sequence for the print file that uses it. ↓

## Format 1 (Series 90):

```
//[ symbol] LCB [ {X'hex-string-1'} ] [ {X'hex-string-2'} , ..., {X'hex-string-n'} ]
                [ {C'char-string-1'} ] [ {C'char-string-2'} , ..., {C'char-string-n'} ]
                [ , CARTNAME=symbol ] [ , NAME={ 48-BUS
                                                48-SCI
                                                63-STD
                                                OWNLC1
                                                OWNLC2 } ] [ , CARTID= { X'aa' } ] [ , NUMBCHAR=n ]
                [ , TYPE={ 0773
                           0768
                           0770
                           0776
                           0778 } ] [ , SPACE={ X'aa'
                                                  X'40'
                                                  C'c' } ]
                [ , MISM= { IGNORE
                           REPORT } ] [ , DUAL= { X'xyyxyyxyyxyyxyy'
                                                  C'ababab'
                                                  C'bbbb'
                                                  X'yyyyyyyy' } ] [ , MISMCHAR= { X'aa'
                                                                 C'c'
                                                                 X'40' } ]
```

## Format 2 (System 80):

```
//[ symbol] LCB [, CARTNAME=symbol ] [ , NAME={ 48-BUS
                                                48-SCI
                                                63-STD
                                                OWNLC1
                                                OWNLC2 } ]
                [ , TYPE={ 0776
                           0789 } ] [ , MISM= { IGNORE
                                                REPORT } ]
```

## Label:

## symbol

Specifies a default cartridge name when *CARTNAME* is omitted or specifies the name of a filed load code buffer (48-BUS, 48-SCI, 63-STD, OWNLC1, OWNLC2) you're changing via the job SG\$PRB. ↑

## LCB

Parameters:

$\left\{ \begin{array}{l} \text{X'hex-string'} \\ \text{C'char-string'} \end{array} \right\}$

Series 90 only. Actual load code buffer on the cartridge for which this load code is being constructed. You need two hexadecimal characters or one EBCDIC character for every graphic symbol on the print band. The position of each in the string of positional parameters corresponds to its position on the print band. As many positional parameters as are needed to specify the entire print band may be used by mixing the character and hexadecimal types, as required.

### NOTES:

1. Statement continuation may be used only between parameters. A particular parameter may not be partially coded on one statement line and continued on another. In hexadecimal representation, the number of digits must be even.
2. When describing print band characteristics for an 0773 or 0778 printer, you must define a load code buffer 256 characters in length if your print band is greater than 64 characters. You may repeat the character array until 256 bytes are defined.

**CARTNAME=**symbol

1- to 8-character identifier of the print cartridge that the operator must mount.

If specified, the system issues a mount message to the operator. If omitted, the user must take steps to stop the execution of his program, if necessary, to allow the operator to mount a required cartridge. You may use *symbol* instead of *CARTNAME* to specify a cartridge name. If you use both *symbol* and *CARTNAME* to specify a cartridge name, *CARTNAME* takes precedence.

**NAME=**
 $\left\{ \begin{array}{l} \text{48-BUS} \\ \text{48-SCI} \\ \text{63-STD} \\ \text{OWNLC1} \\ \text{OWNLC2} \end{array} \right\}$

Name of a filed load code buffer established at SYSGEN time or via the job SG\$PRB. If used, *CARTNAME*, *TYPE*, and *MISM* are the only other parameters you can specify. (In System 80, *NAME* specifies the name of a filed load code buffer, which in turn specifies a cartridge name. So when *NAME* is specified, *CARTNAME* is unnecessary.) When the job SG\$PRB is being used to change a filed buffer, specify the buffer name using *symbol* instead of *NAME*.

**CARTID=**
 $\left\{ \begin{array}{l} \text{X'aa'} \\ \text{C'c'} \end{array} \right\}$

Series 90 only. Load code buffer cartridge identification. May be two hexadecimal digits (X'aa') or one EBCDIC character (C'c').

This parameter is required for the SPERRY UNIVAC 0770 and 0776 Printer Subsystems and is optional for the SPERRY UNIVAC 0773 and 0778 Printer Subsystems. Not applicable for the SPERRY UNIVAC 0768 Printer Subsystem.

# LCB



## NUMBCHAR=n

Series 90 only. Total number of graphic symbols expected on the print band. Should coincide with the number of characters specified in the positional parameters as a safety check.

If omitted, the number of characters specified by the positional parameters is assumed correct.

LCB control statements directed to a SPERRY UNIVAC 0773 Printer Subsystem and containing 63 characters must specify 64 characters; the 36th character on the band will print as the 64th character also.

TYPE= { 0773  
0768  
0770  
0776  
0778  
0789 }

Type of SPERRY UNIVAC print device. The LCB control statement must be specified for a particular device and band character sequence. There is no interchangeability of the LCB control statements between devices.

## NOTE:

*OS/3 also supports the 9300 printer; however, LCB does not apply to the 9300 printer.*

SPACE= { X'aa'  
X'40'  
C'c' }

Series 90 only. Blank character representation. May be two hexadecimal digits (X'aa') or one EBCDIC character (C'c').

MISM= { IGNORE  
REPORT }

Specifies whether character mismatch error conditions are to be reported on the system log device. A mismatch occurs when you try to print a character that has not been placed in the printer's load code buffer or specified as a dual character.

DUAL= { X'xyxyxyxyxyxyxy' }  
C'ababab'  
C'bbbb'  
X'yyyyyyyy'

Series 90 only. Allows you to equate up to four nonprintable character codes with up to four printable characters. This enables you to print data with up to 52 character codes using a 48-character print band.



## LCB

For the SPERRY UNIVAC 0773 and 0778 Printer Subsystems, the DUAL parameter can be used for only the 48-character cartridges. In this case, either  $DUAL=C'bbbb'$  or  $DUAL=x'yyyyyyyy'$  applies, where 'bbbb' or 'yyyyyyyy' corresponds to the 41st, 42nd, 46th, and 48th symbols, respectively, that are physically on the band. These numerals are equated with the load code buffer positions 51, 52, 53, and 54, respectively. The characters you code as 'bbbb' or 'yyyyyyyy' are printed as the symbols that are in these positions on the print band. You have no choice as to the replacement symbol; you should be aware of what is in these positions so that you can interpret your print output. If less than four characters are coded, the uncoded positions assume the space code.

For the SPERRY UNIVAC 0770 or 0776 (Series 90) Printer Subsystems, you have a choice as to the replacement symbol. If you specify in hexadecimal, you would choose the  $DUAL=X,xyxyxyxyxyxyxy'$  parameter option;  $xx$  is a specified character in the load code buffer and  $yy$  is a code that is not in the buffer;  $xx$  and  $yy$  print as the same graphic symbol.

If you specify in EBCDIC, you would use the  $DUAL=C'abababab'$  parameter option,  $a$  being a specified character representing a specific graphic symbol on the print band and  $b$  being a character that is printed as the same graphic symbol as  $a$ . You may specify from one to four character substitutions.

The SPERRY UNIVAC 0768 Printer Subsystem does not provide dualing capability.

$$MISMCHAR = \left\{ \begin{array}{l} X'aa' \\ C'c' \\ X'40' \end{array} \right.$$

Series 90 only. Applies to the SPERRY UNIVAC 0770 and 0776 Printer Subsystems only. The character specified through this parameter is printed for any character detected as a mismatch (a character that is not in the load code buffer and is not dualized). The character specified through the *MISMCHAR* parameter must be specified as a graphic symbol through positional parameter 1.

### NOTES:

1. Only the first five characters of the LCB control statement's keywords need be specified, i.e.,  $CARTIN=symbol$ , rather than  $CARTNAME=symbol$ . If the keyword symbol has less than five characters, you have to specify all.
2. Since the single quote (apostrophe) and ampersand (&) symbols have special meaning to job control, they must always be coded in hexadecimal.

**LFD****Function:**

Links the file information in the control stream with the data management file definition. For each LFD statement in the control stream, a file control block is generated that contains information determined as a result of processing device assignment sets.

A device assignment set consists of at least one DVC statement and one LFD statement for each file to be accessed in the user program, and the LFD statement must follow the DVC statement. Depending on the file definition requirements, a device assignment set may also include VOL, EXT, and LBL statements inserted, in that order, between the DVC and LFD statements.

**Format:**

```

//[symbol] LFD {filename} [, {n}] [ , {ACCEPT
                *filename} [ , {8} ] [ , {EXTEND
                                INIT
                                RELOD
                                PREP } ] ]

```

**Parameters:****filename**

One to eight alphanumeric characters specifying the logical file name; must be the same as the name specified in the label field of the DTF macroinstruction for data management or the PIOC macroinstruction for the physical I/O users.

**\*filename**

Input-only file. The operator should verify that the write enable ring has been removed from the tape reel or that the file protect ON/OFF switch has been pressed to the ON position for the disk pack.

**n**

Maximum number of extents in the file; used to reserve extent table storage for use by data access methods. Space acquired by using this parameter increases total storage requirement for the job and must be taken into consideration when allocating main storage, but not when specifying main storage space on the JOB control statement. This number is the sum of the physical extents on the disk plus the number of partitions in the file.

**ACCEPT**

Data management specifications should be obtained from the format 1 and format 2 labels in the VTOC. Used for files previously opened and closed by data management.

**EXTEND**

Sequential file (tape or disk) is to be extended. The information is appended to the present end of the file, provided your program makes the proper specifications.



## LFD

### INIT

Specified file is to be initialized starting at the first record the first time the file is opened. Previous control information in the format labels is ignored at the file open time and is overwritten by specifications contained in this DVC-LFD sequence at file close time. This parameter should not be used for a checkpoint record file.

### RELOD

Series 90 only. Specified ISAM file is not to be reformatted at file open time when the file is being reloaded.

### PREP

Specifies that a cataloged tape file be prepped before it is used as an output file.



**MTC****Function:**

Positions tape volumes prior to the execution of a job step. Can be used to position a data file or pre-position a multifile tape volume.

Must be specified after the device assignment set for that magnetic tape unit.

**Format:**

```
//[ symbol] MTC lfdname, {
                        BB,nn
                        BM,nn
                        FB,nn
                        FM,nn
                        WM,nn
                        RL
                        RU
                        }
```

**Parameters:****lfdname**

The name of a tape file that is identified in a preceding DVC-LFD control statement sequence.

**BB**

Space volume backward the specified number of blocks.

**BM**

Space volume backward the specified number of tape marks.

**FB**

Space volume forward the specified number of blocks.

**FM**

Space volume forward the specified number of tape marks.

**WM**

Write number of tape marks specified.

**RL**

Rewind volume to load point.

**RU**

Rewind and unload volume.

**nn**

Number of blocks or tape marks (in decimal) to be spaced or number of tape marks to be written.

## NOP

Function:

Inserts labels to be used as targets of branch or SKIP statements.

Format:

```
//[ symbol] NOP [ QUERY]
```

Parameter:

**QUERY**

Used as the target of a dynamic skip function, which is accomplished through the // OPTION QUERY job control statement. This parameter is for workstation users and console operators.

**NOTE:**

*Comments preceded by one or more blanks and enclosed in quotes may be specified in the NOP statement in place of the QUERY parameter. When used for this purpose, NOP does not have to be the target of branch or SKIP job control statements.*

**OPR**

## Function:

Displays a message at the system console or specific workstations. The statement can appear anywhere in the control stream and is displayed at interstep processing time.



## Format:

```
//[symbol] OPR comment-line [,destination-1,...,destination-n]
```

## Parameters:

**comment-line**

Comment or message to be displayed; may contain up to 60 characters and must be enclosed in single quotes if it contains embedded blanks, the slash character, or commas.

**destination** (where destination=[host-id:]user-id)

**user-id**

Directs the message to a particular workstation; can be any 1- to 6-alphanumeric-character workstation user-id, the keyword OPERATOR or \$\$CON (denoting the console), or \$\$MAS (denoting the job's master workstation). \$\$MAS is the default.

**host-id**

In a DDP environment, directs the message to a particular host; is one to four alphanumeric characters and is identical to the label-id of the LOCAP macroinstruction in your ICAM network. You can specify \$HOST to indicate the originator/master host. The host-id is optional but, if specified, must be followed by a user-id. If a host-id is omitted, the local host is assumed.

If you omit a destination, the message goes to the master workstation. A message sent to a master workstation that is not logged on is rerouted to the console. No other messages are rerouted.

If the system does not have workstations or DDP, the message goes to the system console.



## OPTION

### Function:

Specifies certain optional features and controls the operating environment. Some options are effective only in the job step in which they are specified; others are effective from the time the option is encountered until end of job; while others are in effect for the entire job. The options may be written in any order and must be separated by commas with no intervening spaces. Only the first three characters of each option need be coded.

### Format:

```
//[symbol] OPTION p1,...,pn
```

### Parameters:

p1,...,pn

Represents the options to be specified:

ACN=account-number

Overrides the *acct-no* specified in the JOB control statement.

ABRDUMP

Provides abbreviated dump-only printing of selected areas in the macro storage vicinity relative to the current TCB PSW and OPEN DTFs.

BOF

User program is given control with binary overflow interrupt-enabled.

BUF=nXm

Overrides the *nXm* parameter specified in the JOB control statement.

DOF

User program is given control with decimal overflow interrupt-enabled.

DUMP

Job region dump is provided in hexadecimal at job execution time if job step termination is requested, or snapshot dumps are given in response to a SNAP macroinstruction execution.

EOD=xx

Supplies substitute characters for the /\* (end-of-embedded-data) job control statement. Used when embedded data is dialog specification language source code. The first character specified must be a slash (/). The second character can be anything but a slash, an asterisk, an ampersand, or a currency symbol (/, \*, &, \$).

GABRDUMP

Specifies that OPTION ABRDUMP is in effect for every job step from the time GABRDUMP is encountered to end of job.

GDUMP

Specifies that OPTION DUMP is in effect for every job step from the time GDUMP is encountered to end of job.

## OPTION

### GJOBDDUMP

Specifies that OPTION JOBDUMP is in effect for every job step from the time GJOBDDUMP is encountered to end of job.

### GO

Root phase of the last load module generated during the linkage editor job step is automatically executed at the completion of the linkage editor without intervention from job control.

### GSYSDUMP

Specifies that OPTION SYSDUMP is in effect for every job step from the time GSYSDUMP is encountered to end of job.

HDR= { NOHDR }  
      { HDR }

NOHDR suppresses the printing of page separators. HDR allows page separators to be printed. OPTION HDR overrides page separator specification in the JOB control statement.

### HOLD

Places a job containing it in hold status while the job is in the job queue table. A job containing this option is not released until a BEGIN operator command is issued or until a CC job control statement with BE specified is encountered in a subsequent control stream.  
// CC BE cannot be used to release a HOLD within the same job.

### JOBDUMP

Specially edited version of the dump should be provided, if a termination dump is requested.

### LINK

Linkage editor is automatically executed following the termination of the program named on the EXEC statement. This allows you to compile and link job steps with no intervention from job control.

LOG= { logical-unit-number }  
      { ORIGINATOR }  
      { CENTRAL }

Directs the job log to a specific printer or magnetic tape. The keywords ORIGINATOR and CENTRAL apply only to printers. ORIGINATOR directs the log to the printer at the job's originator. (This includes an auxiliary workstation printer if the originator is a workstation.) CENTRAL directs the log to the local site's central printer. Only LOG=CENTRAL can be specified in RBP-initiated jobs. The default log destination for RBP is the originator. If the DVC statement for the output file indicates a specific printer device (e.g., DVC 26) OPTION LOG=26 should be specified so that the job log and the output file are directed to the same device.



## OPTION

↓

MASTER=destination (where destination=[host-id:]user-id)

Assigns the specified workstation (at the specified host) as the master – the workstation that has control of the job during execution. By default, the originator (see OPTION ORI) has control of the job unless you use this option. OPTION MAS becomes effective when the job name is entered in the job queue and remains in effect throughout job processing. Specify OPERATOR to designate the console as master. Specify a host-id to designate (a workstation at) a particular host. If host-id is omitted, the local host is assumed. When you specify a host-id, you must follow it with a user-id. This option in a saved translated control stream is effective when the stream is restored.

↑

MASTER=destination (EXEC) (where destination=[host-id:]user-id)

Functions the same as MASTER=destination but takes effect only when the job is in execution. The originator has control when the job is in the job queue.

→

MAX=maximum-main-storage-size

Overrides the *max* parameter specification in the JOB control statement. The *max* value is expressed in hexadecimal (MAX=number or MAX=X'number). To indicate that the value of *max* is to be interpreted as a decimal value, specify *MAX=D'number*. If a maximum value is specified more than once (via the // JOB statement or multiple // OPTION statements), the largest value is used.

MERGE=NO

Used to create a separate identifier for a job's log in the spool LOG file. By including this option, you can determine if your job log is present in the accumulated LOG file.

→

MIN=minimum-main-storage-size

Overrides the *min* parameter specification in the JOB control statement. The *min* value is expressed in hexadecimal (MAX=number or MAX=X'number). To indicate that the value of *min* is to be interpreted as a decimal value, specify *MIN=D'number*. If a minimum value is specified more than once (via the // JOB statement or multiple // OPTION statements), the largest value is used.

MXT=maximum-time

Overrides the *max-time* parameter specified in the JOB control statement. The maximum time can be specified in minutes, or you can specify SUP or DEF. MXT=SUP suppresses the max-time function. MXT=DEF specifies that the system default is to be used for the *max-time* value.

NODUMP

No dumps are taken in behalf of this job step. This option turns off snap dumps, end-of-job-step-dumps, and abnormal termination dumps. This is defined for compatibility purposes and should not be specified. Job control assumes this condition by default.

→

NOSCHED

Saves a job control stream in its expanded state in \$\$\$SAVE but does not schedule the job. See description of SAVE option for information about subsequent runs of the job.

## OPTION

NOSCHED:(alt-filename [ { RES } ] [,write-password])

Saves the job in its expanded state (in an alternate MIRAM library) but does not schedule the job.

alt-filename

Used to specify a 1- to 44-character file identifier.

[ { RES }  
{ RUN }  
{ vsn } ]

Used to specify the volume containing the alternate file. RES identifies the SYSRES volume, RUN identifies the RUN pack, vsn identifies the volume serial number of a disk pack or format-label diskette. If the file is cataloged, RES, RUN, or a vsn overrides the volume indicated in the catalog. If the file is cataloged and RES, RUN, or a vsn is omitted, the volume indicated in the catalog is used. If the file is not cataloged and you omit RES, RUN, or a vsn, SYSRES is assumed. (Simply code // OPTION NOSCHED:alt-filename in this case.)

[,write-password]

Used to specify a 1- to 6-character write-password when the file is cataloged with a write-password; ignored if the file has no password.

See the SAVE option for information about subsequent runs of the job.

## NSCAN

Resets the SCAN facility. It should be used only within the embedded data of a job step for which SCAN has previously been specified. Subsequent job control statements normally removed by SCAN are not removed. The OPTION NSCAN statement itself is removed. When NSCAN is specified, SCAN cannot be used again in the same job step.

## NSRCH

Specifies that only the library indicated by the EXEC control statement is to be searched.

## NSUB

Resets the SUB facility. It should be used only within the embedded data of a job step for which both SUB and SCAN have previously been specified. Set symbols in embedded data are not substituted until another SUB is encountered.

## NULL

Specifies a no-operation for the OPTION statement.

## OFT = +n

Tells the run processor to reserve space for an additional number (*n*) of files in the open file table. The *n* parameter must be in the range 1 through 16 and must be preceded by a plus sign. For IMS 90 users, *n* is the number of terminal classes used to dynamically create files.

## OPTION

OPL=print-option-list

Overrides *print-option-list* specifications in the JOB control statement. Any of the options available through the *print-option-list* parameter of the JOB control statement may be specified via OPTION OPL.

ORIGINATOR=destination (where destination=[host-id:]user-id)

Assigns a workstation (at the specified host) as the originator. The workstation that physically initiates the job is considered the originator and subsequently has control of the job during processing unless this option is used. OPTION ORI takes effect when it's encountered in the control stream and can be specified several times in one stream. Specify OPERATOR to designate the console as master. Specify a host-id to designate (a workstation at) a particular host. If host-id is omitted, the local host is assumed. When you specify a host-id, you must follow it with a user-id. This option in a saved translated control stream is effective when the stream is restored.

OUT={  
ORIGINATOR  
CENTRAL  
[host-id:]user-id

Directs all job output (print files, punch files, and job logs) to the specified destination as follows:

- ORIGINATOR

Directs all printed output to the printer at the job's originator. Directs all punch output to the central punch at the job's originator.

- CENTRAL

Directs all print or punch output to the local site's central printer/punch.

- [host-id:]user-id

Directs all printed output to the specified destination and all punch output to the central punch at the specified host.

The host-id is one to four alphanumeric characters long and identical to the label-id of the LOCAP macroinstruction in your ICAM network. Use \$HOST to indicate the job's originator. The local host is assumed if the host-id is omitted. The host-id is optional but, if specified, must be followed by a user-id.

A 1- to 6-alphanumeric-character workstation user-id identifies an auxiliary workstation printer. The keyword CENTRAL in place of a user-id identifies the central printer.

This option is effective for all of the job's print and punch output but can be changed for individual print or punch files by specifying // ROUTE or // DST in the device assignment set for that file.



**OPTION****PRI=switch-priority**

Establishes an overall task switching priority that can be changed for particular job steps by specifying a relative priority (e.g., +3 or -3) or an absolute priority (e.g., 3) on the EXEC statement.

PRT =  $\left. \begin{array}{l} \text{ACT} \\ \text{LOG} \\ \text{NOACT} \\ \text{NOLOG} \\ \text{NONE} \\ \text{BOTH} \end{array} \right\}$

Overrides the print option specified in the JOB control statement.

- ACT forces the printing of accounting records.
- LOG forces the printing of job log records.
- NOACT suppresses the printing of log information from the job log file.
- NOLOG suppresses the printing of both accounting records and log information from the log file.
- NONE suppresses the printing of both accounting records and log information from the log file.
- BOTH allows the printing of accounting records and job log information.

**QUERY**

Allows the interactive user to change control stream execution from a workstation. The QUERY option is explained in the job control user guide for your system.

**REPEAT**

Currently executing program is automatically reinitialized upon termination until all embedded data files are exhausted. This gives you the ability to execute stacked assemblies or compilations without job control intervention.

**SAVE**

Saves a job control stream in its expanded state in \$\$\$SAVE and schedules the job to be run. A copy of the control stream as it appears in \$\$\$RUN is stored in the system file \$\$\$SAVE. Subsequent runs of the job are initiated through the SC/SI workstation or system console command, or through the CC {SC/SI} job control statement. Saving an expanded job control stream that has a number of jprocs eliminates the time-consuming chore of jproc expansion by the run processor on subsequent runs. Information about the SC/SI workstation/console command is found in the workstation user guide and the operations handbook, respectively, for your system. This option is available only if your system is configured with consolidated data management.

## OPTION

↓

SAVE: (alt-filename [ { RES }  
                          { RUN }  
                          { vsn } ] [,write-password] )

Functions the same as SAVE but saves the control stream in an alternate MIRAM library.

alt-filename

Used to specify a 1- to 44-character file identifier.

[ { RES }  
  { RUN }  
  { vsn } ]

Used to specify the volume containing the alternate file. RES identifies the SYSRES volume, RUN identifies the RUN pack, vsn identifies the volume serial number of a disk pack or format-label diskette. If the file is cataloged, RES, RUN, or a vsn overrides the volume indicated in the catalog. If the file is cataloged and RES, RUN, or a vsn is omitted, the volume indicated in the catalog is used. If the file is not cataloged and you omit RES, RUN, or a vsn, SYSRES is assumed. (Simply code // OPTION NOSCHED:alt-filename in this case.)

[,write-password]

Used to specify a 1- to 6-character write-password when the file is cataloged with a write-password; ignored if the file has no pasword.

↑

### SEVERE

Specifies that the run processor is to be terminated (the job is not to be scheduled) if warning errors are encountered. Normally, warning errors would not terminate the job.

### SCAN

Acts upon and then removes selected control statements (CR, OPTION, GBL, GO, IF, JSET, and NOP) from the embedded data files.

If omitted, only the terminators (FIN, END, /\$, and /\*) are detected.

### SIG

User program is given control with floating-point significant exception interrupt-enabled.

### SUB

Scans embedded data for possible parameter or set symbol substitution.

### SYSDUMP

Full edited system dump is provided if job step termination is requested.

### TEST

Specifies that the job is not to be queued or run.

### TRACE

Brings in the monitor routine to record the effect of each executed instruction. Option monitor tasks are selected as described in the supervisor user guide.

**OPTION****TSK=number-of-tasks**

Overrides the tasks parameter specified in the JOB control statement. From 1 to 255 tasks can be active within any job step.

**UNDEFINED**

Specifies that from the time this option is encountered to the end of job, a warning error message is to be generated whenever an undefined SET symbol is detected.

**UNEQUAL**

Specifies that a warning error message is to be generated whenever two unequal character strings are compared.

**XUF**

User program is given control with exponent underflow exception interrupt-enabled.

**NOTE:**

*The OPTION job control statement should not be placed between these job control statements:*

*EXEC and /\$*

*EXEC and PARAM*

*PARAM and PARAM*

*/\* and /\$ (where they delimit two separate embedded data sets)*

## PARAM

### Function:

Used to submit information to the program during its execution. The information can be in the form of keyword parameters, control statements (other than job control statements), or any other type of information the program wants it to be. The changes are coded in the operand field. Job control verifies the format.

PARAM statements must follow the EXEC control statement. Therefore, control belongs to the program named in the EXEC control statement, not to job control.

There is no limit to the number of PARAM control statements allowed in a control stream.

### Format:

```
//[ symbol ] PARAM operand-1[ , . . . , operand-n ]
```

### Parameter:

#### operand

Contains the information that is passed to the requesting program. If the information contains embedded blanks, it must be enclosed with single quotation marks.

# PAUSE

## Function:

Displays a message at the system console or specific workstations but also causes the processing of the job to come to a halt until the message is acknowledged. The statement can be placed anywhere in the job step, but is not displayed at the system console or workstation until immediately before a program is executed.

## Format:

```
//[symbol] PAUSE comment-line [,destination-1,...,destination-n]
```

## Parameters:

### comment-line

Comment or message to be displayed; may contain up to 60 characters and must be enclosed in single quotes if it contains embedded blanks, the slash character, or commas.

### destination (where destination=[ host-id:]user-id)

#### host-id

In a DDP environment, directs the message to a particular host; is one to four alphanumeric characters and is identical to the label-id of the LOCAP macroinstruction in your ICAM network. You can specify \$HOST to indicate the originator/master host. The host-id is optional but, if specified, must be followed by a user-id. If host-id is omitted, the local host is assumed.

#### user-id

Directs the message to a particular workstation; can be any 1- to 6-alphanumeric-character workstation user-id, the keyword OPERATOR or \$\$CON (denoting the console), or \$\$MAS (denoting the job's master workstation). \$\$MAS is the default.

If you omit a destination, the message goes to the master workstation. A message sent to a master workstation that is not logged on is rerouted to the console. No other messages are rerouted.

If the system does not have workstations or DDP, the message goes to the system console.

## NOTE:

*The PAUSE statement suspends all job control activity for the control string in which it appears until the message is acknowledged, but the processing of other jobs in the system continues.*

## QGBL

### Function:

Allows you to change the value of global set symbols at run time from the workstation.

### Format:

```
//[symbol] QGBL set-id-1[=init-1] [,set-id-2[=init-2] ,...,set-id-n[=init-n]]
```

### Parameters:

set-id-1 through set-id-n

Name of the set symbols; can be a maximum of eight characters in length.

=init-1 through =init-n

Initial value of set symbols; can be a maximum of 60 characters in length.

↓  
**NOTE:**

Whenever a quoted value is assigned to a set symbol (using // QGBL), the quotes are considered part of the value. (For example, if // QGBL X='ABC', then &X is 'ABC' unless changed.)  
↑

## QUAL

### Function:

Prefixes an automatic qualifier to all subsequent file identifiers in the job; remains in effect until the end of the job or until another QUAL job control statement is encountered.

### Format:

```
//[ symbol] QUAL [ qualname]
```

### Parameters:

#### qualname

Qualifier of one to eight alphanumeric characters.

If omitted, overrides the previous QUAL job control statement and terminates the use of a qualifier.

## REN

Function:

→ Permanently changes (renames) the label of a disk or format-label file.

Format:

```
//[symbol] REN lfdname, {new-label } [ ,NTERM]
                        { 'new-label' }
```

Parameters:

**lfdname**

Identifies the file to be renamed. It must match the *lfdname* in the LFD statement for the file.

**new-label**

Specifies the new name for the file; 1 to 44 alphanumeric characters in length. If *new-label* contains embedded blanks, it must be enclosed by apostrophes.

**NTERM**

Specifies that errors encountered during the renaming process are to be ignored and the job allowed to continue. If this parameter is specified and a renaming error occurs, the job continues, but the file is not renamed. If this parameter is omitted and a renaming error occurs, the job terminates at the point of error.

NOTES:

1. Subsequent references to a renamed disk file must specify *new-label* in the LBL statement of the device assignment set for the file.
2. If you rename a cataloged file, you must recatalog the file with the new name.
- 3. REN statements are not permitted against files on SYSRES that begin with *\$Y\$* or against files on SYSRUN that begin with *\$Y\$R*.



## ROUTE

### Function:

Routes spooled output to printers or punches at DDP sites and to auxiliary workstation printers. This includes local auxiliary workstation printers, remote auxiliary workstation printers, and local or remote printers and punches. The statement must appear in the device assignment set for the file to be routed.

### Format:

```
//[symbol] ROUTE destination-1[,...,destination-8]
```

### Parameters:

**destination** (where destination=[host-id:]user-id)

**host-id**

Identifies a particular host in a DDP network; is one to four alphanumeric characters long and identical to the label-id of the LOCAP macroinstruction in your ICAM network. Specify \$HOST to indicate the host that initiated the job (the originator/master). The host-id is optional but, if specified, must be followed by a user-id. If a host-id is omitted, the local host is assumed.

**user-id**

A 1- to 6-alphanumeric-character workstation user-id (denoting an auxiliary workstation printer); \$Y\$MAS (denoting an auxiliary workstation printer at the master workstation); or CENTRAL (denoting a central printer or punch).

Any destination that specifies a workstation user-id or \$Y\$MAS is valid only for print files.

### NOTES:

1. *DDP or auxiliary printer output (specified by // ROUTE) and RBP output (specified by // DST) cannot be mixed for any one job. For any job, all output must be of one type or the other. Also, DDP destinations and local auxiliary printer destinations cannot be used for the same print file.*
2. *When a workstation initiates a job that directs printed output to an auxiliary printer connected to another workstation (one that is not the originator), the user at the other workstation must issue an RP command to initiate printing. See the interactive services commands and facilities user guide/programmer reference for information about the RR command.*



## RST

### Function:

Restarts a BAL or COBOL program from a checkpoint when a computer malfunction causes a job to terminate. Checkpoint records are established by the CHKPT macroinstruction (in BAL) or the RERUN clause (in COBOL).

When the job is on cards, the RST statement must be the first card in the deck when the job is rerun. For a prefiled job, RST can be submitted from a card reader or added to the control stream (as the first statement) via the general editor (EDT) or the librarian. When the job is rerun, the program's execution begins at or near the checkpoint reached when the job stopped.

### Format:

```
//[ symbol] RST filename,checkpoint-id,number [,jobname( (rename))][ ,pri ]  
[ ,key-1=val-1, ...,key-n=val-n]
```

### Parameters:

#### filename

Name of the checkpoint file; must agree with the filename specified on the LFD statement for the checkpoint file.

#### checkpoint-id

Checkpoint number identifying which checkpoint is used to restart the job. This number is displayed on the system console when checkpoint records are encountered in the program.

#### number

Specifies the number of the job step that executes the user program to be restarted.

#### jobname

Names a prefiled job to be rerun when RST is submitted from a card reader. Must be specified if the *(rename)* parameter is used to assign an alternate name to the prefiled job.

#### (rename)

Alternate name of the job. Executes a previously filed control stream and enables use of different job names.

#### pr i

Scheduling priority.

#### key-n=val-n

A keyword and its value that may be referenced as would any GBL control statement. The effect of these parameters is as if a GBL control statement were inserted as the first control statement of the job. The total length of this parameter cannot exceed 44 characters.

**RST****NOTES:**

1. *All information necessary to prepare an RST control statement is displayed at the system console each time a program encounters a checkpoint.*
2. *The LFD job control statement in the device assignment set for the checkpoint file must not contain the INIT parameter. This parameter causes the file to be written from the beginning of the file, thus overwriting any existing records.*
3. *In COBOL, the RERUN clause is equivalent to the checkpoint macroinstruction.*
4. *The following must be taken into consideration when restarting a job:*
  - *Only one RST control statement may be submitted for any one job.*
  - *If the program being executed at the time the checkpoint is recorded is in the job's \$Y\$RUN file (output of the linkage editor), the job to be restarted will not run to normal completion if a program overlay is called after the job has been restarted.*
  - *If a restart is to be done after the job has terminated (normally or abnormally), the restarted job step must not have originally had requests for temporary work areas.*
  - *Tapes previously positioned via an MTC control statement are not positioned to the proper point in the restarted job.*
  - *Card files cannot be repositioned.*
  - *If a multiframe tape is to be repositioned, the file sequence number must be included on the LBL job control statement.*
  - *The disk file containing checkpoint records cannot contain user data.*
  - *Scheduling may be delayed if all the resources needed by all job steps in the job are not available, even if those needed only by the job step to be restarted are available.*
  - *Mount messages to the operator may be produced for volumes that were not needed in the original run. This is due to the fact that SKIP control statements are ignored.*
  - *Workstation files cannot be repositioned; therefore, a job cannot be restarted from a checkpoint that occurs when workstation files are opened.*

## RUN/RV

### Function:

The RUN and RV statements permit another job to be initiated. RUN and RV control statements function in the same way, except that RUN should be used to call a control stream that is on cards or that is prefiled (in \$Y\$JCS or an alternate library) and contains a CR statement because input (from a card reader) is necessary. RV is only used to call prefiled control streams that do not require a card reader. If RUN is used when a card reader is not required, the job is not initiated until the (unnecessary) reader is available. If RV is used when a card reader is required, the job is not initiated.

The RUN or RV statement may appear anywhere within the control stream, except between /\$ and /\* control statements or a critical series of control statements, such as the DVC through LFD sequence.

### Format:

```
//[symbol] { RUN [ { jobname[(new-name)] } ] }
              { (new-name) }
              { RV jobname[(new-name)] }
              { [ :alt-filename
                  { /alt-filename, { RES
                                     { RUN
                                     vsn }
                                 }
                  :alt-filename, [ { RES
                                     { RUN
                                     vsn }
                                 ], read-password )
              }
              [ , { PRE } ] [ , key-1=val-1, ..., key-n=val-n ]
              [ { HIGH } ]
              [ { NOR } ]
```

### Parameters:

#### jobname [(new-name)]

One to eight alphanumeric characters identifying the job. A job name is required with RV, but can be omitted from RUN if the control stream is to be read from a card reader. The name used on the control stream's JOB statement is assigned in this case.

Include *(new-name)* to assign a new 1- to 8-alphanumeric character name identifying the selected job control stream. The job assumes the new name, thus reducing the chance of duplicate job names. The new name alone may be used with RUN if the control stream is input from a card reader.

## RUN/RV

:alt-filename

Specifies the name of the alternate SAT library file, residing on SYSRES, that contains the job control stream. If the file name is cataloged (without a read password), the volume associated with that file name in the catalog is used. ←

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

Specifies the name of the alternate SAT library file and the volume containing the file. RES identifies SYSRES as the volume, RUN identifies the system RUN pack, and vsn identifies the volume serial number of a disk pack or format-label diskette. If the file name is cataloged (without a read password), the volume you specify (RES, RUN, or a vsn) is used instead of the volume associated with the file name in the catalog. ←

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

Specifies the name of the alternate SAT library file and the volume containing the file. The *read-password* parameter must be specified if the file name is in the file catalog with a read password. RES identifies SYSRES as the volume containing the file, RUN identifies the system RUN pack, and vsn identifies the volume serial number of a disk pack or format-label diskette. If RES, RUN, or a vsn is omitted, the volume associated with the file name in the catalog is used. ←

If no alternate file name is specified, the job control stream is read from \$Y\$JCS.

PRE

Specifies preemptive priority for the job control stream.

HIGH

Specifies high priority for the job control stream.

NOR

Specifies normal priority for the job control stream.

If omitted, the priority specified in the control stream's JOB statement is used.

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

A keyword and its value that may be referenced as would any GBL control statement. The effect of these parameters is as if a GBL control statement were inserted as the first control statement of the job.

## NOTE:

The total length of the RUN and RV statements, from the R in RUN or RV to the last character of the last specified value, cannot exceed 60 characters. ←

## SCR

### Function:

Scratches (deletes) files. Any file specified is deleted immediately upon the detection of an SCR control statement. Therefore, files may be deleted prior to the execution of a given job step. Files to be scratched should not be in use by another job. Files with a label prefix of \$\$\$ cannot be deleted from the SYSRES volume. Only files on volumes currently mounted are deleted. Only one volume serial number may be specified for any SCR job control statement. The DATE and PRE parameters are not supported for diskette files.

### Format:

```
//[symbol] SCR lfdname [ , { DATE, [ yyddd] } ]
                    [ { PRE, [ aaaa] } ]
```

### Parameters:

#### lfdname

File name to be deleted; must agree with the file name given on a previous LFD statement in a DVC-LFD sequence. If positional parameter 2 is DATE or PRE, the LBL statement may be omitted from the DVC-LFD sequence.

#### DATE

Delete all expired files from the volume. The expiration date, if different from the current date, is specified in positional parameter 3.

#### PRE

All files on the volume with the prefix specified in positional parameter 3 are to be deleted. The first three characters of the prefix cannot be \$\$\$.

If positional parameter 2 is omitted, the entire file specified by positional parameter 1 is deleted. Positional parameter 3 should also be omitted.

#### yyddd

Date used in testing for expired files:

#### yy

Is the year.

#### ddd

Is the day of the year.

Leading zeros must be specified. If the date is omitted, the current date from the job prologue is used.

#### aaaa

The 4-character prefix of files to be deleted. The first three characters of the prefix must not be \$\$\$\$. If more than four characters are specified, the prefix is truncated to four. If omitted, the first four characters of the file identifier from the associated LBL job control statement are used.

**SET**

## Function:

Sets or modifies the date fields, the user program switch indicator (UPSI), or the communication region in the job prologue. The SET statement does not alter fields in the system information block (SIB); this must be done by the SET operator command.

## Format:

```
//[ symbol] SET { DATE,yy/mm/dd[ ,yyddd] [ ,yyddd]
                  UPSI,switch-setting
                  COMREG,character-string }
```

## Parameters:

**DATE**

Stores the contents of positional parameters 2, 3, and 4 in the job prologue date fields. These dates are to be used rather than the system date.

**UPSI**

Sets the UPSI byte in the job prologue according to the bit pattern of positional parameter 2. The UPSI byte is the last byte of the 12-byte communication region in the job preamble.

**COMREG**

Stores the contents of positional parameter 2 in the job prologue communication region.

**NOTE:**

*If UPSI and COMREG are not specified, the fields in the job prologue are set to binary 0's.*

**yy/mm/dd**

Calendar date.

**switch-setting**

Switch setting (one to eight bits). The bit indicators are:

0	Off
1	On
X	Unchanged

Unspecified rightmost positions are assumed to be X. The byte is initially set to 0.

**character-string**

Stores 2 to 24 hexadecimal or 1 to 12 EBCDIC characters in the job preamble communications region. The data is left-justified, and unspecified rightmost characters remain unchanged. Hexadecimal characters are designated X'cccc...' and must be even in length. EBCDIC characters are designated C'cccc...'.

## SET

yyddd

5-digit date for a tape file:

yy

Is year.

ddd

Is day of the year.

The date is right-justified in a 6-character field in the job preamble. The leftmost character is set to an EBCDIC blank. You may, however, indicate the quarter of the year in this position.

If positional parameter 3 is omitted, the SIB date is used.

yyddd

5-digit date for a disk file:

yy

Is the year.

ddd

Is the day of the year.

The date is converted to a 4-byte discontinuous binary format, Oydd.

If positional parameter 4 is omitted, the positional parameter 3 value is used, if specified, or the SIB value is used if both positional parameters 3 and 4 are omitted.



**SFT**

## Function:

There are three applications for the SFT statement:

- Identifying user-written shared code modules that reside in a library not on SYSRES or \$Y\$RUN.
- Identifying data management shared-code modules so that those modules are loaded before job initiation and stay resident for the duration of the job. Even though you do not need to identify data management modules to load them, if you do identify them through // SFT, they are loaded *before* job initiation. This ensures that your job does not have to wait for main storage space to be allocated for the data management modules.
- Specifying the dynamic loading facility and/or overriding system generation limits for dynamic expansion of the user job region. (This feature is for ANSI 74 COBOL users.)

**NOTE:**

*There is a system generation parameter (IGNORESFT) that allows the Series 90 user to specify that // SFT job control statements be ignored. This system generation option is useful because you can then take advantage of the OS/3 dynamic shared code feature without changing existing control streams that contain // SFT job control statements. The system installation user guide/programmer reference contains more information about this system generation option.*

## Format:

$$//[\text{symbol}] \text{ SFT } \left\{ \begin{array}{l} \text{module-1} [, \dots, \text{module-n}] \left[ \text{, DLOAD} = \left[ \left( [\text{calls}] , \left[ \left\{ \text{expansion-limit} \right\} \right] \right) \right] \right] \\ \text{DLOAD} = \left[ \left( [\text{calls}] , \left[ \left\{ \text{MAX} \right\} \right] \right) \right] \end{array} \right\}$$

## Parameters:

**module**

Identifies the shared-code modules needed.

**DLOAD**

Tells the run processor that your job needs the OS/3 dynamic loading facility and/or that the job is to override the sysgen limits for expansion of the user job region. (For ANSI 74 COBOL users only.)

**calls**

Specifies the maximum number of dynamically loaded modules allowed for this job; if omitted, the system default (as set by the SYSGEN parameter DLOADTABLE) applies.

**expansion-limit**

Specifies the maximum number of bytes (total) that can be added to this job in support of the DLOAD facility; considered hexadecimal if you specify *X'number* or *number*; considered decimal if you specify *D'number*.

## SFT

### MAX

Indicates that the size of the job is limited only by the amount of main storage in the system.

→ If an expansion limit is omitted (*expansion-limit* or *MAX* are not specified), the system default (as set by the SYSGEN parameter DLOADBUFR) applies.

### NOTE:

*Data management shared-code load modules reside in the system library \$Y\$SCLOD. The SAT librarian is used to list the modules in \$Y\$SCLOD and to obtain information about them.*

## SKIP

### Function:

Bypasses desired portions of the control stream, either conditionally, based on UPSI bit settings, or unconditionally. This skip is effective at execution time, as opposed to the IF and GO control statements, where the skip occurs when the run processor scans the control stream prior to execution. The statements that are skipped are not acted on. The skip can only be in a forward direction.

### Format:

```
//[symbol] SKIP target-label[,mask]
```

### Parameters:

#### target-label

Corresponds to the label specified in the optional label field of another control statement. The label field may be on a NOP control statement.

#### mask

Is a binary number used to test the UPSI byte (one to eight bits). Thus, the SKIP statement is conditional. If fewer than eight bits are specified, the unspecified rightmost positions are each assumed to be 0.

Effectively, a test-under-mask instruction is executed. If any of the specified UPSI bits are set, the skip condition is satisfied and the statement is processed. Otherwise, the statement is ignored and the processing continues with the next statement in the control stream.

The allowable characters in the mask are:

- 0 Ignore the bit.
- 1 Test the corresponding UPSI bit to determine where it is set.

If the *mask* parameter is omitted, the SKIP control statement is unconditional.

## SPL

### Function:

Controls output spooling. When used, SPL must occur within the DVC-LFD sequence for the file to be spooled. Ignored in a nonspooling environment.

### Format:

```

//[ symbol] SPL {
  {
    HOLD
    RETAIN
    DUMP
    DISK
    TAPE
    DISKETTE
  }
} [ ,nXm] [ , {no-cop}
  {
    1
  }
]
  [ , {no-skpcode}
  {
    7
  }
] [ , {max-rec}
  {
    5120
  }
] [ , forms] [ , {NOHDR}
  {
    HDR
  }
] [ , NOTSTL]
  [ , brk-pge] [ , NOUPD] [ , NOCMP] [ , RETAIN] [ , HOLD] [ , SECURE]

```

### Parameters:

#### HOLD

Output file is to be held for later disposition. Can be released by a BEGIN SPL workstation/console command or through the CC job control statement specifying this command.

#### RETAIN

Output file is retained in the spool file after it is printed or punched; file may have to be reprinted or repunched at a later time.

#### DUMP

Redirects the spooled output to a tape volume; output can later be printed or punched; should not be used when the DEV operator command is active.

#### DISK

Redirects the spooled output to another disk volume.

#### TAPE

Redirects the spooled output to a tape volume.

#### DISKETTE

System 80 only. Redirects the spooled output to a format-label diskette.

### NOTES:

1. The DUMP and TAPE parameters have exactly the same function. DUMP is included in the SPL statement to provide compatibility with previous Series 90 releases.

**SPL**

2. *With Series 90 systems, spooled output can be redirected to disk only if the system is configured with basic dynamic buffer management. For information about basic dynamic buffer management, see the system installation user guide/programmer reference.*
3. *When using the SPL statement for a spooled data-set-label output file, only the nXm, NOUPD, NOCMP, RETAIN, and HOLD parameters are meaningful.*

**nXm**

Buffer size established for use by this file:

**n**

Number of buffers. No advantage is gained by making n greater than 2, unless system symbionts are being used.

**X**

Is a constant.

**m**

Is the size of each buffer, in 256-byte increments.

If omitted, the file shares the buffer pool with the job log and other spooled files not having their own buffers.

{no-cop}  
1

Number of times the spooled file is to be printed or punched (output), in range of 0 through 255. Zero indicates no output.

{no-skpcode}  
7

Used when a filed vertical format buffer having more than seven skip codes is requested (via // VFB) or when the system default buffer has more than seven skip codes. Indicates the number of lines in the buffer that contain skip codes, plus three (one for forms overflow and two for home paper position). Zero indicates no skip codes.

{max-rec}  
5120

Maximum number of records (lines, including spaces and skipped lines from print files, cards for punch files) that can be placed in the spool file before the operator is queried as to whether the job should be continued or terminated. The largest number the *max-rec* parameter will accept is 262144.

**forms**

A 1- to 8-alphanumeric-character name identifying a special printer form or card type to the operator.

**NOHDR**

Suppresses the printing of page headers in burst mode.

# SPL

**HDR**

Allows page headers to be printed.

**NOTSTL**

Suppresses the query to the operator questioning if a sample test pattern page is to be printed when a change of form name is detected.

**brk -pge**

Indicates the specific number of pages or cards to be spooled out before a file is breakpointed and printed or punched. The largest number that can be entered in this parameter is 32,000.

**NOUPD**

Indicates that the spool subdirectory entry is to be updated only when a file is closed. If NOUPD is specified and the program cancels, any output generated before cancellation is lost. If omitted (causing the spooler to update the subdirectory entry each time it crosses a logical track in the program file) and the program cancels, output generated before cancellation can be printed.

**NOCMP**

Prevents the system from attempting to compress data that's directed to the output spool file. Normally, NOCMP should not be specified if data in the file contains a large number of embedded blanks or if the file contains a block size larger than 120. If specified when the block size is 121 or greater, the resulting spool file contains only one line per sector and requires that nXm be at least 2X4.

**RETAIN**

Functions the same as the first *RETAIN* parameter but is independent of the other first-parameter options. If specified with *TAPE*, *DUMP*, *DISK*, or *DISKETTE*, the output is redirected to the appropriate device while a copy of the file is also retained for later use.

**HOLD**

Functions the same as the first *HOLD* parameter but is independent of the other first-parameter options. If specified with *RETAIN*; *TAPE*, *DUMP*, *DISK*, or *DISKETTE*; or *RETAIN* and *TAPE*, *DUMP*, *DISK*, or *DISKETTE*, the spool file is put on hold first. Other parameters are acted upon accordingly when the file is released.

**SECURE**

Suppresses the printing of the output file if the workstation to which the auxiliary printer is connected is not logged on. If not specified, the file is printed at the specified auxiliary printer whether or not the workstation is logged on.

## UID

## Function:

Allows you to specify a workstation, including the system master workstation, by user-id, device address, or both. The workstation specified is required and, when logged on, is automatically connected to the job. Used as part of the device assignment set for a workstation.

## Format:

$$//[\text{symbol}] \text{UID} \left\{ \begin{array}{l} \text{user-id-1} \\ (\text{addr-1}) \\ \text{user-id-1}(\text{addr-1}) \end{array} \right\} \left[ \dots, \left\{ \begin{array}{l} \text{user-id-255} \\ (\text{addr-255}) \\ \text{user-id-255}(\text{addr-255}) \end{array} \right\} \right]$$

## Parameters:

**user-id**

Specifies a workstation by user-id. You can use `$$MAS` as a user-id to identify the system master workstation.

**(addr)**

Specifies the physical address of the device in hexadecimal; represents the channel number, control unit address, and the device number.

**user-id(addr)**

Specifies a workstation by user-id and physical address.

## USE DP

### Function:

Specifies that the dialog processor is needed in support of a dialog session at the workstation. Used as part of the device assignment set for a workstation.

### Format:

```
//[ symbol]USE DP,dialog-name[ ,printer-lfd][ ,new-audit-lfd][ ,old-audit-lfd]
```

### Parameters:

#### dialog-name

A 1- to 8-alphanumeric-character name of the dialog file; must match the name specified on the LFD statement of the dialog file's device assignment set.

#### printer-lfd

A 1- to 8-alphanumeric-character name of the printer file; must match the name specified on the LFD statement of the printer's device assignment set. This parameter is specified when you want to produce a printed summary of the dialog session.

#### new-audit-lfd

A 1- to 8-alphanumeric-character name of the new audit file produced by the dialog processor; must match the name specified on the LFD statement of the new audit file's device assignment set.

#### old-audit-lfd

A 1- to 8-alphanumeric-character name of the old audit file input to the dialog processor; must match the name specified on the LFD statement of the old audit file's device assignment set.

### NOTE:

*Audit files must be previously allocated MIRAM files.*



**USE LIB** ↓

## Function:

Under consolidated data management, allows a user program to either sequentially read or write (create) a source module. Specified in the device assignment set for a source library file.

## Format:

```
//[ symbol] USE LIB,module-name
```

## Parameters:

**module-name**

A 1- to 8-alphanumeric-character name of the source module to be accessed. The first character must be alphabetic.

 ↑

## USE MENU

### Function:

Indicates that menu services are to be used at the workstation. Specified as part of the workstation device assignment set.

### Format:

```
//[symbol] USE MENU [ , ( menu-file-LFD/$SY$FMT ) ] [ , initial-menu ]
                    [ { $SY$FMT/menu-file-LFD } ]
                    [ { $SY$FMT } ]
                    [ , ( nnn ) ] [ , menu-format-1=alias-1 [ , ... , menu-format-12=alias-12 . ] ]
                    [ { 1 } ]
```

### Parameters:

```
[ , ( menu-file-LFD/$SY$FMT ) ]
  [ { $SY$FMT/menu-file-LFD } ]
  [ { $SY$FMT } ]
```

Specifies LFD names for up to two menu format files; must match LFD names used in previously defined device assignment sets for the files. The menu-file-LFD name is one to eight alphanumeric characters in length and must refer to a MIRAM file.

#### menu-file-LFD/\$SY\$FMT

Identifies two format files. The file identified by menu-file-LFD is examined first, then \$SY\$FMT.

#### \$SY\$FMT/menu-file-LFD

Identifies two format files. \$SY\$FMT is examined first, then the file identified by menu-file-LFD.

If nothing is specified, it is assumed that all menu formats reside in the system format file \$SY\$FMT.

#### initial-menu

Specifies the name of the first menu to be used. It is one to eight alphanumeric characters in length.

#### nnn

Specifies the number of menus to be resident in main storage at any one time, in the range 1 to 255. The default value is 1.

## USE SFS

## Function:

Specifies that screen format services are needed in support of a screen format session at the workstation. Used as part of the device assignment set for a workstation.

## Format:

```

//[symbol] USE SFS [ , { [format-file-LFD-1] / [format-file-LFD-2] } ] [ , initial-screen]
                    {
                    format-file-LFD
                    SYSFMT
                    }
                    [ , { nnn } ] [ , screen-format-1=alias-1 [ , ... , screen-format-12=alias-12 ] ]

```

## Parameters:

```

[ , { [format-file-LFD-1] / [format-file-LFD-2] } ]
  {
  format-file-LFD
  SYSFMT
  }

```

Specifies LFD names for up to two screen format files; must match LFD names used in previously defined device assignment sets for the screen format files. The format-file-LFD name is one to eight alphanumeric characters in length and must refer to a MIRAM file.

[ format-file-LFD-1 ] / [ format-file-LFD-2 ]

Identifies two format-file-LFD names. If *format-file-LFD-1* is coded alone, *format-file-LFD-1* is examined first then *SYSFMT*. If */format-file-LFD-2* is coded alone, *SYSFMT* is examined first then *format-file-LFD-2*.

format-file-LFD

Identifies one format-file-LFD name. This is the only file examined.

If nothing is specified, it is assumed that all screen formats reside in the system format file *SYSFMT*.

initial-screen

Specifies the name of a screen format to be used by the application program. It is one to eight alphanumeric characters in length. Use of this parameter depends on the program's language. For more information, see the screen formatting concepts and facilities manual.

nnn

Specifies the number of screens to be resident in main storage at any one time, in the range 1 to 255. The default value is 1.

screen-format=alias

Equates a screen format name specified in an application program (*alias*) to the screen format name generated by the screen format generator. A maximum of twelve alias name sets may be specified. The *screen-format* name and *alias* name may each be from one to eight alphanumeric characters in length.

## VFB

### Function:

Overrides the system default vertical format buffer for a printer; becomes effective when the file is opened. During interstep-job processing, the SYSGEN default value is used. If used, the VFB directive must be within the DVC-LFD control statement sequence for the print file that uses it.

### Format:

```
//[ symbol] VFB [ ,_FORMNAME=symbol] [ ,_USE={STAND1} [ ,_LENGTH=lines] [ ,_DENSITY={6} ]
[ ,_TYPE={0773
           0768
           0770
           0776
           0778
           9300
           0789} ] [ ,_OVF=(line-1,...,line-n)] [ ,_OVF2=(line-1,...,line-n)]
[ ,_CD1=(line-1,...,line-n),... [ ,_CD15=(line-1,...,line-n)] ]
```

### Label:

#### symbol

Specifies 1- to 8-character vertical format buffer name; used as the default form name when the FORMNAME parameter is omitted.

Specifies a default form name when *FORMNAME* is omitted or specifies the name of a filed vertical format buffer (STAND1 or OWNVF1) you're changing via the job SG\$PRB.

### Parameters:

#### FORMNAME=symbol

Specifies a 1- to 8-character printer form name. This is the name that the operator associates with the actual form. He is asked to mount this form before you access the print file. A form name specified through this parameter takes precedence over a form name specified in symbol. Form names specified through VFB statements take precedence over form names specified through the SPL job control statement or the SPOOL jproc.

USE={STAND1}  
{OWNVF1}

Name of a filed vertical format buffer established at SYSGEN time or via the job SG\$PRB. If used, *FORMNAME* and *TYPE* are the only other parameters you can specify. When the job SG\$PRB is being used to change a filed buffer, specify the buffer name in *symbol* instead of *USE*.

#### LENGTH=lines

Total length of printer form in terms of lines, in the range of 1 to 192.

VFB

DENSITY={ 6 }  
{ 8 }

Number of print lines per inch. If specified, *LENGTH* must also be specified.

TYPE={ 0773 }  
{ 0768 }  
{ 0770 }  
{ 0776 }  
{ 0778 }  
{ 9300 }  
{ 0789 }

Type of printer to be used. If the VFB is designed to be used with more than one type of printer, the TYPE parameter should be omitted. The 0789 printer is for System 80 only.

## NOTE:

*In the following keyword parameters, line is a decimal number between 1 and the number specified in the LENGTH keyword parameter. It specifies which line number is to be assigned a skip code (overflow or device control code). A particular code may be repeated if desired, or multiple lines for a given code may be specified as one parameter. If only one line is specified for a given code, the enclosing parentheses may be omitted. If multiple codes for a given line are used, the first is accepted and a diagnostic error message is given on all others.*

OVF=(line-1,...,line-n)  
Overflow line indicator.

OVF2=(line-1,...,line-n)  
Secondary overflow line indicator.

CD1=(line-1,...,line-n),...,CD15=(line-1,...,line-n)  
Device control codes for the printer. They indicate which lines control certain functions, such as skipping and line spacing. See the data management user guide for a list of the applicable codes for each printer.

## NOTE:

*In a spooling system, the number of skip codes being used must be passed to the open processors. If a VFB job control statement is present in the device assignment set for a file being spooled, the number of codes specified is automatically passed without the necessity of an SPL job control statement. However, if you request a filed vertical format buffer (STAND1 or OWNVF1) having more than seven skip codes or if you use a system default buffer having more than seven skip codes, you must specify the number of codes using the no-skpcode parameter in the // SPL statement or the SKPCODE parameter in the // SPOOL jproc.*

*When a // SPL statement or // SPOOL jproc is omitted, the default is 7 skip codes. Three skip codes are always included in this count: home position for current page, overflow for next page, and home position for next page. The four remaining are user-specified. The // SPL statement and // SPOOL jproc, therefore, specify the total count of lines on a form where a skip code is allowed, plus three.*

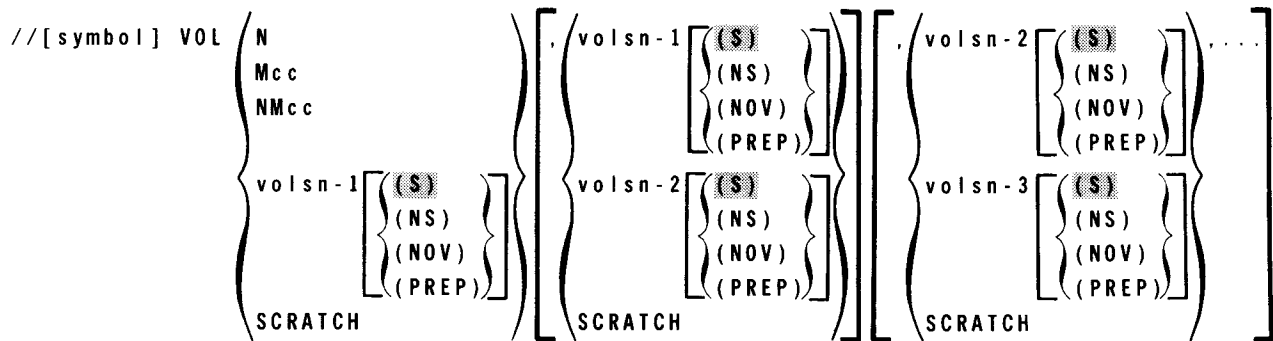
# VOL

## Function:

Supplies the volume serial numbers used to uniquely identify tape and disk volumes.

When used, the VOL statement must immediately follow the DVC statement in a device assignment set. For a disk file, the first VOL statement must precede any EXT statements for that volume.

## Format:



## Parameters:

### N

Suppress block number checking on input tape volumes or block number writing on output tape volumes.

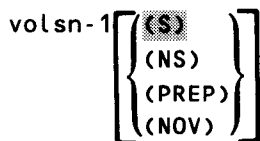
If N is not used, tape block numbering depends upon the SYSGEN specification and the BKNO parameter of the DTFMT macro. The N parameter is effective only during initialized processing.

### Mcc

Mode characteristics of the tape volumes specified in positional parameters 2 through n. The allowable values for cc are given in the data management user guide. If Mcc is not specified, the mode settings specified at system generation time are used.

### NMcc

Combines both the N and Mcc options.



One to six alphanumeric characters specifying the volume serial number of the first volume of the file. If less than six characters are used, the number is left-justified and padded with blanks. The volume may be shareable (S) or nonshareable (NS), and volume checking may be suppressed (NOV). For tape files, a request to write volume labels may be made (PREP). Any information that is currently on the volume is effectively erased if PREP is specified.

**VOL****SCRATCH**

Used to specify a multivolume file without explicitly listing the serial numbers for each volume. May appear only once in a VOL statement and is always the last parameter specified.

**NOTES:**

1. OS/3 assumes that all volume serial numbers are unique. The loading of two volumes with identical volume serial numbers at the same time will yield unpredictable results.
2. Extreme care should be taken when using the PREP option on a file to be processed by the librarian. When PREP is specified, the tape is prepped every time it is opened as output. The librarian closes output tape files whenever they are to be used as input and then reopens them as input. If this tape file is to be reused as an output file, within the same job, the librarian will close it as input and reopen it as output. This reopening will cause the file to be prepped (if PREP was specified), thereby effectively erasing all the information previously produced. The PREP option, therefore, should only be used if the file is going to be output only or output, then input. If this is not the case, use the TPREP utility routine to prep the file. The PREP option cannot be suppressed. You must redefine the tape file without specifying the PREP option on the VOL statement.

vol sn-1  $\left[ \begin{array}{l} (S) \\ (NS) \\ (PREP) \\ (NOV) \end{array} \right] \left[ \begin{array}{l} (S) \\ (NS) \\ (PREP) \\ (NOV) \end{array} \right] , \text{vol sn-2}, \dots$

Continuation of a variable amount of volume serial numbers; must appear in the same sequence required for mounting.

**NOTES:**

1. To suppress checking of volume serial numbers for multivolume files, you must specify NOV in the VOL statement for the last volume of the file.
2. For multivolume files, if PREP is specified for any of the volumes, all volumes in the file are prepped.
3. When both fixed and removable 8415 disks are specified in a control stream, logical unit numbers for the fixed disks (160/162) must appear at least once in the control stream before logical unit numbers for the removable disks (161/163).
4. When referencing multivolume files on the VOL statement, you must represent any undeclared volume serial numbers with commas. Additionally, if neither Mcc, N, or NMcc are specified for the first positional parameter, you must supply a comma.

---

**/\$**

Function:

Start of data. All statements following the /\$ statement up to and including the associated end-of-data (/\*) statement are filed in \$Y\$RUN.

The /\$ must be the only statement on the line; however, comments may follow the /\$ if preceded by one or more blanks.

Format:

**/\$**

No positional parameters are required.

*NOTE:*

*If using a SPERRY UNIVAC 0716 Card Reader Subsystem with the 96-column card feature supported, the data file can use all 96 columns. With data-set-label diskette, the data file can use up to 128 characters. Job control statements, however, can only use the first 72 columns (characters).*



/\*

## Function:

End of data; used in conjunction with the start-of-data (/ \$) statement. When the /\* statement is encountered by job control, the data file is closed and control stream processing is resumed. The /\* statement is filed with the data.

The /\* must be the only statement on the line; however, comments may follow the /\* if preceded by one or more blanks.

## Format:

/\*

No positional parameters are required.

**/&**

Function:

End of a control stream.

The /& statement must be the only statement on the line; however, comments may follow the /& if preceded by one or more blanks.

Format:

**/&**

No positional parameters are required.

### **3. Job Control Procs**



## ACCESS

### Function:

Generates the job control statements required to assign a device to a job so that the file on that device can be accessed at job execution time. The procedure call statement is included in the control stream at the point where the device assignment set is required. This statement can be used for tape files and previously allocated disk files.

### Format:

```
//lfdname ACCESS ( lblname
                   lfdname [ . {n} ] ( [ . ACCEPT
                                     . EXTEND
                                     . INIT
                                     . RELOD
                                     . PREP ] ) [ . DVC=nn, VOL= { vol sn }
                                               . VOL= { vol sn } { . } ]
```

### Label:

#### lfdname

Name of the file to be accessed. This corresponds to the 8-character name in the LFD job control statement.

### Parameters:

#### lblname

File identifier under which the file was created. This corresponds to the tape or disk file identifier in the LBL job control statement.

#### n

Number of extents in the file. Reserves extent table storage for use by the data access methods. The maximum value is 16. Space acquired by using this parameter increases the total storage requirement for the job and must be taken into consideration when allocating main storage.

#### ACCEPT

The data management specifications should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.

#### EXTEND

Adds information to the present end of a sequential (SQ) file.

#### INIT

Initializes the file, starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file close time.

## ACCESS

### RELOAD

Series 90 only. ISAM file is not to be reformatted when it is reloaded.

### PREP

Series 90 only. Specifies that a cataloged tape volume is to be prepped before it is used as an output file.

### DVC=nn

Device type, where nn is the logical unit number of the device.

If this keyword parameter is omitted, the file is assumed to be on the device containing the job's \$Y\$RUN file.

VOL= { vol sn  
      RUN  
      \* }

Identifies the volume where the file resides. The options are:

VOL=vol sn

Volume serial number.

VOL=\*

Volume is identified in the file catalog.

If this keyword parameter is omitted, the file is assumed to be in the job's \$Y\$RUN file.

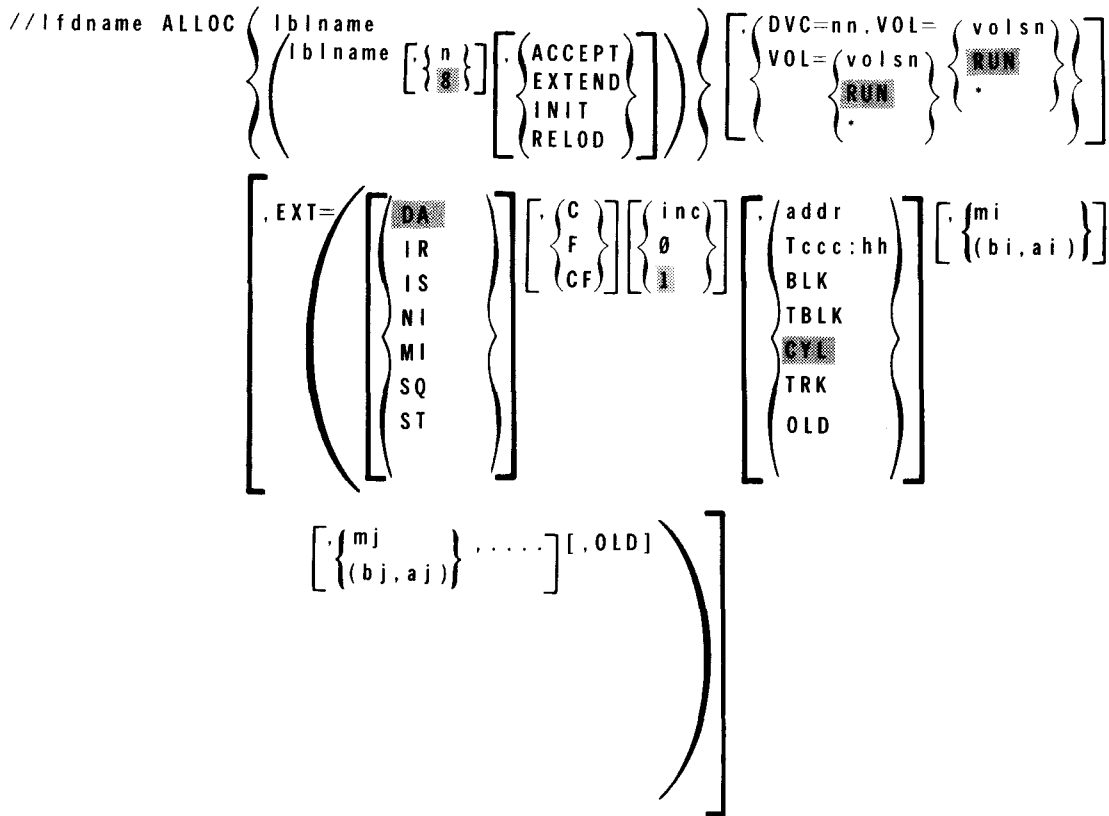
# ALLOC

**Function:**

Generates the job control statements required to assign a disk or diskette to a job step and to allocate space on that disk or diskette for the file. This statement is placed in the control stream where the device assignment set is required. Use it to create new files or extend current files.

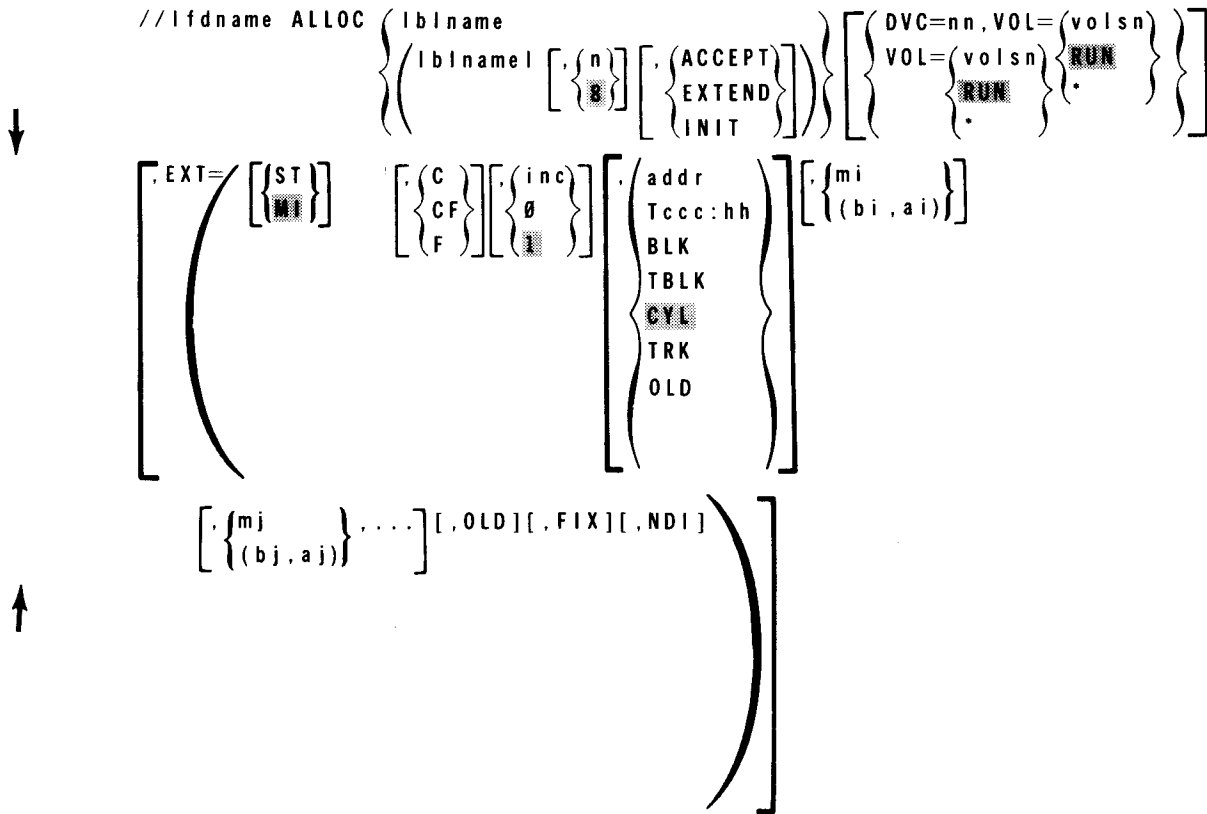
See Appendix B for further information regarding extent specification.

**Format 1 (Series 90):**



# ALLOC

Format 2 (System 80):



Label:

**lfdname**

Identifies the file; corresponds to the 8-character name in the LFD control statement.

Parameters:

**lbfname**

File identifier used when the file was created; corresponds to the disk file identifier in the LBL job control statement.

**n**

Number of extents in the file. Reserves extent table storage for use by the data access methods. The maximum value is 16. Space acquired by using this parameter increases the total storage requirement for the job and must be taken into consideration when allocating main storage.

**ACCEPT**

Indicates that the DTF specifications for data management should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.



**ALLOC****EXTEND**

Adds information to the end of a sequential (SQ) file.

**INIT**

Initializes the file, starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file close time.

**RELOAD**

Series 90 only. ISAM file is not be reformatted when it is reloaded.

**DVC=nn**

Device type, where nn is the logical unit number of the device.

If this keyword parameter is omitted, the file is assumed to be on the device containing the job's \$Y\$RUN file.

**VOL=** $\left\{ \begin{array}{l} \text{vol sn} \\ \text{RUN} \\ * \end{array} \right\}$

Identifies the volume where the file resides. The options are:

**VOL=vol sn**

Volume serial number.

**VOL=\***

Volume is identified in the file catalog.

If this keyword parameter is omitted, the file is assumed to be in the job's \$Y\$RUN file.

**EXT=**

Supplies the extent specifications to be used when reserving system resources for a particular file. The information is coded within parentheses.

With Series 90 systems, if this keyword parameter is completely omitted, job control allocates one cylinder of extent space for a direct access (DA) file. With System 80, job control allocates one cylinder of extent space for a MIRAM (MI) file. In either case, job control assumes one cylinder for dynamic extension.

**DA**

Series 90 only. Direct access file.

**IR**

Series 90 only. IRAM file.

**IS**

Series 90 only. ISAM file.

**NI**

Series 90 only. Nonindexed (DA or SQ) file.

## ALLOC

**MI**

MIRAM file; only valid specification for System 80 data-set-label diskette.

**SQ**

Series 90 only. Sequential file; only valid specification for data-set-label diskette.

**ST**

System access technique (SAT) file.

**NOTES:**

1. In System 80, data files must be allocated as MIRAM files.
2. In System 80, all parameters that apply to disk (except FIX) apply to format-label diskette.

**C**

Allocate contiguous space; must be specified for data-set-label diskette files.

**F**

Reformat the file at allocation time. Subparameter 4 must be *BLK*.

**CF**

Both of the preceding.

**inc**

Secondary increment (in cylinders or blocks) for automatic extension. An increment of zero indicates that there can be no dynamic extension of the file. If there is no *EXT* keyword parameter for this file, the value of the most recently specified secondary increment is used.

**∅**

File cannot be dynamically extended. Must be specified for data-set-label diskette.

**addr**

Absolute beginning cylinder address, in terms of cylinders.

**Tccc:hh**

Absolute track address (hexadecimal) in cylinder/head format where the file is to begin. Allocation is in terms of tracks.

**BLK**

Allocation in terms of blocks; actual allocation is in terms of cylinders; only valid specification for data-set-label diskette files.

**TBLK**

Allocates space in blocks; actual allocation is in terms of tracks.

**ALLOC****CYL**

Allocation in terms of cylinders.

**TRK**

Allocates space in tracks.

**NOTE:**

TRK, TBLK, and Tccc:hh cannot be used for ISAM, indexed IRAM, or indexed IRAM characteristic MIRAM files.

**OLD**

Secondary allocation increment change (subparameter 3) for an existing file; cannot be followed by any other subparameters if specified; cannot be specified for data-set-label diskette.

**mi**

Number of cylinders or tracks to allocate for this file; subparameter 4 must be *CYL*, *addr*, *TRK*, or *Tccc:hh*.

**(bi,ai)**

Used when allocation is in terms of blocks (by cylinder or track) for disk and diskette files; subparameter 4 must be *BLK* or *TBLK*.

**bi**

Is the average block length.

**ai**

Is the number of blocks to allocate.

**Subparameter 6:**

Same as subparameter 5 but used to describe additional extents; never specified for data-set-label diskette files.

**OLD**

Indicates that the file's previously allocated extent is to be increased by the allocation amount (*mi*, (*bi,ai*), etc) specified.

If OLD is omitted, the request is for a new extent; may not be specified for data-set-label diskette.

**NDI**

System 80 only. NDI (nondata interchange) must be specified to allocate space for all data-set-label diskette files that do not use the basic data exchange (BDE).

**FIX**

System 80 only. Indicates you're allocating this extent in the fixed-head area of an 8417 disk.

# ASM

## Function:

Generates the necessary job control statements to run an assembly language processor. Optionally, it can generate the job control statements that specify the following:

- input-source library;
- output-object library;
- procedure, copy, and alternate load libraries;
- PARAM job control statements to define the format of the assembler listing; and
- OPTION job control statements to automatically execute the linkage editor and the generated load module.

## Format:

↓

```

//[symbol] { ASM } [ PRNTR={ lun[,dest] } ] [ IN={ (vol-ser-no,label) } ]
           { ASML } [ { N[,dest] } ] [ { (RES) } ]
           { ASMLG } [ { 20[,dest] } ] [ { (RES,label) } ]
                                           [ { (RUN,label) } ]
                                           [ { (*,label) } ]

                                           [ ,OUT={ (vol-ser-no,label) } ]
                                           [ { (RES,label) } ]
                                           [ { (RUN,label) } ]
                                           [ { (*,label) } ]
                                           [ { (N) } ]
                                           [ { (RUN,$YSRUN) } ]

                                           [ ,LIN={ (vol-ser-no-1,label-1) } ] [ { (vol-ser-no-2,label-2) } ]
                                           [ { (RES,label-1) } ] [ { (RES,label-2) } ]
                                           [ { (RUN,label-1) } ] [ { (RUN,label-2) } ]
                                           [ { (*,label-1) } ] [ { (*,label-2) } ]
                                           [ N ] [ N ]
                                           [ { (RES,$YSMAC) } ] [ { (RES,$YSMAC) } ]

                                           [ ,COPY={ (vol-ser-no-1,label-1) } ] [ { (vol-ser-no-2,label-2) } ]
                                           [ { (RES,label-1) } ] [ { (RES,label-2) } ]
                                           [ { (RUN,label-1) } ] [ { (RUN,label-2) } ]
                                           [ { (*,label-1) } ] [ { (*,label-2) } ]
                                           [ N ] [ N ]
                                           [ { (RES,$YSSRC) } ] [ { (RES,$YSSRC) } ]

                                           [ ,LST={ option } ] [ ,SCR1={ vol-ser-no } ]
                                           [ { (opt-1,...opt-n) } ] [ { RES } ]

                                           [ ,SCR2={ vol-ser-no } ] [ ,ALTLOD={ (vol-ser-no,label) } ]
                                           [ { RUN } ] [ { (RES,label) } ]
                                           [ ] [ { (RUN,label) } ]
                                           [ ] [ { (*,label) } ]
                                           [ ] [ { (RES,$YSL0D) } ]
                                           [ ] [ { (RUN,$YSRUN) } ]
    
```

↑

**ASM****NOTE:**

The ASM procedure call is used to assemble a program. When the ASML procedure call is used, an *OPTION LINK* job control statement is generated, which allows for the automatic execution of the linkage editor. When the ASMLG procedure call is used, an *OPTION LINK,GO* job control statement is generated, which allows for the automatic execution of the linkage editor, followed by the automatic execution of the program.

When either the ASML or ASMLG procedure calls are used, the *OUT* parameter to define a specific output library cannot be used.

**Label:****symbol**

Source module name of one to eight alphanumeric characters; only needed when the *IN* parameter is used.

**Parameters:**

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \end{array} \right\}$$

Input file definition; also replaces the *PARAM IN* control statement. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**IN=(RES)**

Input is on SYSRES, in \$\$\$SRC.



## ASM

**IN=(RES, label)**

Input is on SYSRES, but in a file other than `$$SRC`.

**IN=(RUN, label)**

Input is on the volume containing the job's `$$RUN` file, with a file identifier specified by *label*.

**IN=(\*, label)**

Input volume is identified in the file catalog with a file identifier specified by *label*.

If omitted, source input is in the form of data cards.

**NOTE:**

*The IN parameter must be used if the source correction feature of the assembler (i.e., SKI, REC, SEQ type cards) is to be used instead of strictly source cards.*

$$\text{OUT} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \\ (\text{N}) \\ (\text{RUN, } \text{\texttt{SYSRUN}}) \end{array} \right\}$$

Output file definition; also replaces the PARAM OUT control statement. The options are:

**OUT=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**OUT=(RES, label)**

Output is on SYSRES, with a file identifier specified by *label*.

**OUT=(RUN, label)**

Output is on the volume containing the job's `$$RUN` file, with a file identifier specified by *label*.

**OUT=(\*, label)**

Output volume is identified in the file catalog with a file identifier specified by *label*.

**OUT=(N)**

No object code output.

If omitted, output is in the job's `$$RUN` file.

## ASM

$$\text{LIN} = \left( \begin{array}{l} (\text{vol-ser-no}, \text{label-1}) \\ (\text{RES}, \text{label-1}) \\ (\text{RUN}, \text{label-1}) \\ (*, \text{label-1}) \\ \text{N} \\ (\text{RES}, \text{\$YSMAC}) \end{array} \right) , \left( \begin{array}{l} (\text{vol-ser-no-2}, \text{label-2}) \\ (\text{RES}, \text{label-2}) \\ (\text{RUN}, \text{label-2}) \\ (*, \text{label-2}) \\ \text{N} \\ (\text{RES}, \text{\$YSMAC}) \end{array} \right)$$

Indicates the location of the private procedure and macro libraries. RES indicates a library on SYSRES; RUN indicates a library on the volume containing the job's  $\$Y\$RUN$  file; the \* indicates the volume is identified in the file catalog. The file identifier is specified by *label*. N indicates no libraries. If omitted, the library is on SYSRES in the file  $\$Y\$MAC$ .

$$\text{COPY} = \left( \begin{array}{l} (\text{vol-ser-no-1}, \text{label-1}) \\ (\text{RES}, \text{label-1}) \\ (\text{RUN}, \text{label-1}) \\ (*, \text{label-1}) \\ \text{N} \\ (\text{RES}, \text{\$Y\$SRC}) \end{array} \right) , \left( \begin{array}{l} (\text{vol-ser-no-2}, \text{label-2}) \\ (\text{RES}, \text{label-2}) \\ (\text{RUN}, \text{label-2}) \\ (*, \text{label-2}) \\ \text{N} \\ (\text{RES}, \text{\$Y\$SRC}) \end{array} \right)$$

Indicates the location of the copy libraries. RES indicates a library on SYSRES; RUN indicates a library on the volume containing the job's  $\$Y\$RUN$  file; the \* indicates the volume is identified in the file catalog. The file identifier is specified by *label*. N indicates no libraries. If omitted, the library is on SYSRES in the file  $\$Y\$SRC$ .

$$\text{LST} = \left\{ \begin{array}{l} \text{option} \\ (\text{opt-1}, \dots, \text{opt-n}) \end{array} \right\}$$

Specifies the format of the assembler listing. The options are:

- N  
Specifies that no assembly listing is produced.
- NC  
Specifies that no cross-reference listing is produced.
- ND  
Specifies that no diagnostic listing is produced.
- NR  
Specifies that the cross-reference listing is to contain only symbols that have at least one reference each. The NC option, if specified with NR, always overrides it.
- DBG  
Specifies a proc or macro debug mode feature within the OS/3 assembler.
- (NC, ND)  
Specifies that neither a cross-reference nor diagnostic listing is produced.
- (NR, ND)  
Specifies that no diagnostic listing is produced and the cross-reference contains only symbols with references.

## ASM

SCR1={vol-ser-no}  
RES

Specifies the volume serial number of the first work file.

SCR2={vol-ser-no}  
RUN

Specifies the volume serial number of the second work file.

ALTLOD={ (vol-ser-no, label)  
(RES, label)  
(RUN, label)  
(\* , label)  
RES, \$Y\$LOD  
RUN, \$Y\$RUN }

Specifies the alternate load library that contains the assembler language processor. The options are:

ALTLOD=(vol-ser-no, label)

Indicates the volume serial number and file identifier.

ALTLOD=(RES, label)

Alternate load library is on SYSRES, with a file identifier specified by *label*.

ALTLOD=(RUN, label)

Alternate load library is on the volume containing the job's \$Y\$RUN file, with a file identifier specified by *label*.

ALTLOD>(\* , label)

Alternate load library is on the volume identified in the file catalog, with a file identifier specified by *label*.

If omitted, the alternate load library is in \$Y\$LOD when ASM and ASML are used and \$Y\$RUN when ASMLG is used.



↓

# AUTO

## Function:

Generates the necessary job control statements to run RPG II auto report. The AUTO jproc has four forms:

## AUTO

Processes auto report source programs.

## AUTRPG

Processes auto report source programs and then compiles the RPG II generated source program.

## AUTRPGI

Processes auto report source programs, compiles the generated RPG II source program, and link-edits the generated object module.

## AUTRPGIG

Processes auto report source programs, compiles the generated RPG II source program, link-edits the generated object module, and executes the generated load module.

## Format:

```
//[symbol] {
  AUTO
  AUTRPG
  AUTRPGI
  AUTRPGIG
} [
  PRNTR={
    lun[,dest]
    N[,dest]
    20[,dest]
  } [
    IN={
      (vol-ser-no,label)
      (RES)
      (RES,label)
      (RUN,label)
      (*,label)
    }
  ]
  [
    OUT={
      (vol-ser-no,label)
      (RES,label)
      (RUN,label)
      (*,label)
      (N)
      (RUN,SYSRUN)
    } [
      OUTSRC={
        (vol-ser-no,label,lfd-name)
        (RES,label,lfd-name)
      }
    ]
  ]
  [
    LST={
      K
      M
      N
      S
    } [
      SCR1={
        vol-ser-no
        RES
      } [
      SCR2={
        vol-ser-no
        RUN
      }
    ]
  ]
  [
    ALTL0D={
      (vol-ser-no,label)
      (RES,label)
      (RUN,label)
      (*,label)
      (RES,SYSL0D)
    } [
    EMB={
      NO
      YES
    } [
    MOD={
      3
      4
      5
      IRAM
    }
  ]
  [
    SKIP=C [
    COPYn={
      (vol-ser-no,label,lfd-name)
      (RES,label,lfd-name)
      (RUN,label,lfd-name)
    }
  ]
  [
    ERRFIL=(vol-ser-no,label,module-name)
  ]
]
```

↑

# AUTO

Label:

symbol

Source module name of one to six alphanumeric characters; needed only when the IN parameter is used.

Parameters:

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \end{array} \right\}$$

Input file definition; also replaces the PARAM IN control statement. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**IN=(RES)**

Input is on SYSRES, in \$\$\$SRC.

**IN=(RES, label)**

Input is on SYSRES, but in a file other than \$\$\$SRC.

**IN=(RUN, label)**

Input is on the volume containing the job's \$\$\$RUN file, with a file identifier specified by *label*.

**IN=(\*, label)**

Input is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, source input is in the form of data cards.

↓  
**AUTO**

$$\text{OUT} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{N}) \\ (\text{RUN SYSRUN}) \end{array} \right\}$$

Output file definition; also replaces the PARAM IN control statement. The options are:

**OUT=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**OUT=(RES, label)**

Output is on SYSRES, with a file identifier specified by *label*.

**OUT=(RUN, label)**

Output is on the volume containing the job's \$\$RUN file, with a file identifier specified by *label*.

**OUT=(\*, label)**

Output is on the volume identified in the file catalog with a file identifier specified by *label*.

**OUT=(N)**

No object code output.

If omitted, output is in the job's \$\$RUN file.

$$\text{OUTSRC} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}, \text{bfd-name}) \\ (\text{RES}, \text{label}, \text{bfd-name}) \end{array} \right\}$$

Output source file definition for the cataloged source file option on the auto report U specification. The options are:

**OUTSRC=(vol-ser-no, label, bfd-name)**

Indicates the volume serial number, file identifier (*label*), and logical file name (*bfd-name*) of the output source file.

**OUTSRC=(RES, label, bfd-name)**

Indicates an output source file on SYSRES with a file identifier specified by *label* and a logical file name specified by *bfd-name*.

If omitted, there is no output file definition for the cataloged source file option.

$$\text{LST} = \left\{ \begin{array}{l} \text{K} \\ \text{M} \\ \text{N} \\ \text{S} \end{array} \right\}$$

Specifies the format of the compiler listing:

**K**

Inhibits error flags for sequence errors.

**M**

Inhibits source and diagnostic listings.

↑

↓

## AUTO

N  
Inhibits all listable output.

S  
Inhibits main storage map listing.

If omitted, the complete compiler listing is printed.

SCR1={vol-ser-no}  
      {RES}

Specifies the volume serial number of the first scratch work file.

SCR2={vol-ser-no}  
      {RUN}

Specifies the volume serial number of the second scratch work file.

ALTLOD={ (vol-ser-no, label)  
          { (RES, label)  
          { (RUN, label)  
          { (\*, label)  
          { (RES, SYSLOD)

Specifies the alternate load library containing the auto report product (AUTO#). The options are:

ALTLOD=(vol-ser-no, label)  
Indicates the volume serial number and file identifier (label).

ALTLOD=(RES, label)  
Alternate load library is on SYSRES; a file identifier is specified by *label*.

ALTLOD=(RUN, label)  
Alternate load library is on the volume containing the job's \$Y\$RUN file; a file identifier is specified by *label*.

ALTLOD=(\*, label)  
Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, the AUTO# load module is located on SYSRES in \$Y\$LOD.

EMB={NO }  
      {YES }

Indicates whether embedded linkage editor control statements are to be generated.

↑

**AUTO**

$$\text{MOD} = \left. \begin{array}{l} 3 \\ 4 \\ 5 \\ \text{IRAM} \end{array} \right\}$$
**MOD=3**

Specifies an RPG II compilation in IBM System/34 mode. When this mode is specified, IRAM (Series 90) or MIRAM (System 80) will be used to process all disk files; the logical file definition (LFD) for printer files is changed to PRNTR, PRNTR1...PRNTRn; and the control reader (CTLRDR) is used for card input even though the data management reader (READER) is specified.

**MOD=4**

Same as MOD=3 except that printer files are generated with the same names as used in the program and reader files use the data management reader (READER).

**MOD=5**

The program is to be compiled in the IBM System/34 mode with native mode data management accessing the disk files.

**MOD=IRAM**

Indicates that the IRAM processor is to be used to process all disk files in the program.

**SKIP=C**

Causes auto report to generate SKIP to channel for the printer output file. If omitted, auto report generates SKIP to line numbers for the printer output file.

$$\text{COPY} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}, \text{lfd-name}) \\ (\text{RES}, \text{label}, \text{lfd-name}) \\ (\text{RUN}, \text{label}, \text{lfd-name}) \end{array} \right\}$$

Input source file definition for /COPY statements within auto report source input. The number of copies corresponds to the number of unique files used to copy.

**COPYn=(vol-ser-no, label, lfd-name)**

Indicates the volume serial number, the file identifier (*label*), and the logical file name (*lfd-name*).

**COPYn=(RES, label, lfd-name)**

Indicates an input source file on SYSRES with a file identifier specified by *label* and a logical file name specified by *lfd-name*.

**COPYn=(RUN, label, lfd-name)**

Indicates an input source library in the job's \$Y\$RUN file with a file identifier specified by *label* and a logical file name specified by *lfd-name*.


If omitted, there is no input source file definition for /COPY statements.



  
**AUTO**

**ERRFIL=(vol-ser-no,label,module-name)**

Specifies an error log file. Error diagnostic messages are written to this file by the error log processor. The parameter *vol-ser-no* specifies the volume serial number, *label* specifies the file identifier, *module-name* specifies the 1- to 8-alphanumeric-character name of the error file element being created. (Does not need to match or be related to the RPG II source object module name.)



# COBL74

**Function:**

Generates the job control statements to execute the ANSI 1974 COBOL language processor. The COBL74 jproc call has three forms:

- COBL74

Compiles an ANSI 1974 COBOL source program.

- COBL74L

Compiles an ANSI 1974 COBOL source program and link-edits object modules.

- COBL74LG

Compiles an ANSI 1974 COBOL source program, link-edits object modules, and executes the load module.

**NOTE:**

*COBL74, COBL74L, and COBL74LG are the only COBOL jprocs supported in System 80.*

**Format:**

```
//[symbol] { COBL74
             { COBL74L
             { COBL74LG } [ PRNTR={ lun[,dest]
                          { N[,dest]
                          { Z0[,dest] }
                          [ ,IN={ (vol-ser-no,label)
                                (RES)
                                (RES,label)
                                (RUN,label)
                                (*,label) } [ ,LIN={ (vol-ser-no,label)
                                                    (RES,label)
                                                    (RUN,label)
                                                    (*,label)
                                                    (RES,$YSSRC) }
                          [ ,OBJ={ (vol-ser-no,label)
                                (RES,label)
                                (RUN,label)
                                (*,label)
                                (RUN,$YSRUN) } [ ,SCR1={ vol-ser-no
                                                    RES }
                          [ ,SCR2={ vol-ser-no
                                RES } ] [ ,SCR3={ vol-ser-no
                                                    RUN }
                          [ ,ALTLOD={ (vol-ser-no,label)
                                (RES,label)
                                (RUN,label)
                                (*,label)
                                (RES,$Y$LOD)
                                (RUN,$YSRUN) } [ ,option=specification
                          [ ,ERRFIL={ vol-ser-no,label,module-name }
```







## COBL74

Label:

## symbol

Source module name of one to six alphanumeric characters. Required only when the IN parameter is specified.

Parameters:

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \end{array} \right\}$$

Input file definition; also generates a PARAM IN job control statement. The options are:

**IN=(vol-ser-no, label)**

Specifies the file identifier (*label*) and the volume serial number for the input file.

**IN=(RES)**

Input is on SYSRES in \$Y\$SRC.

**IN=(RES, label)**

Input is on SYSRES, with the file identifier specified by *label*.

**IN=(RUN, label)**

Input is on the volume containing the job's \$Y\$RUN file, with a file identifier specified by *label*.

**COBL74**

IN>(\* , label )

Input volume is identified in the file catalog with a file identifier specified by *label*.

NOTE:

If this parameter is omitted, source input is in the form of data cards.

LIN= { (vol - ser - no , label )  
(RES , label )  
(RUN , label )  
(\* , label )  
(RES , \$\$\$SRC ) }

Specifies the location of the copy modules. The lfdname is COPY\$. The options are:

LIN=(vol - ser - no , label )

Specifies the file identifier (label) and the volume serial number.

LIN=(RES , label )

Modules are on SYSRES; the file identifier is specified by *label*.

LIN=(RUN , label )

Modules are on the volume containing the job's \$\$RUN file; a file identifier is specified by *label*.

LIN>(\* , label )

Modules are on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, modules are on SYSRES in the file \$\$\$SRC.

OBJ= { (vol - ser - no , label )  
(RES , label )  
(RUN , label )  
(\* , label )  
(RUN , \$\$\$RUN ) }

Specifies the output file definition and generates a PARAM OBJ job control statement. The options are:

OBJ=(vol - ser - no , label )

Specifies the location of the object module.

OBJ=(RES , label )

Specifies that the object module is located on the SYSRES device, with the file identifier specified by *label*.

OBJ=(RUN , label )

Specifies that the object module is located on the job's \$\$RUN file, with the file identifier specified by *label*.

OBJ>(\* , label )

The object module is on a volume identified in the file catalog with the file identifier specified by *label*.

**COBL74****OBJ=(RUN,\$Y\$RUN)**

The object module is located on the job's \$Y\$RUN file. This is the default value if the OBJ parameter is omitted.

*NOTE:*

The OBJ keyword parameter cannot be used with the COBL74L or COBL74LG forms of the jproc call.

$$\text{SCR1} = \left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RES} \end{array} \right\}$$

Specifies the volume serial number of the work file with an identifier of \$SCR1.

$$\text{SCR2} = \left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RES} \end{array} \right\}$$

Specifies the volume serial number of the work file with an identifier of \$SCR2.

$$\text{SCR3} = \left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RUN} \end{array} \right\}$$

Specifies the volume serial number of the work file with an identifier of \$SCR3.

$$\text{ALTLOD} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{RES}, \text{\$Y\$LOD}) \\ (\text{RUN}, \text{\$Y\$RUN}) \end{array} \right\}$$

Specifies the alternate load library that contains the ANSI 1974 COBOL language processor. The options are:

**ALTLOD=(vol-ser-no, label)**

Indicates the volume serial number and file identifier (label).

**ALTLOD=(RES, label)**

Alternate load library is on \$Y\$RES; a file identifier is specified by *label*.

**ALTLOD=(RUN, label)**

Alternate load library is on the volume containing the job's \$Y\$RUN file; a file identifier is specified by *label*.

**ALTLOD=(\*, label)**

Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, the alternate load library is in \$Y\$LOD when COBOL74 and COBOL74L are used, and \$Y\$RUN when COBOL74LG is used.

**option=specification**

Specifies the various compiler options.

## COBL74

↓  
ERRFIL=(vol-ser-no,label,module-name)

Specifies an error log file. Error diagnostic messages are written to this file by the error log processor. The *vol-ser-no* parameter specifies the volume serial number, *label* specifies the file identifier, *module-name* specifies the 1- to 8-alphanumeric-character name of the error file element being created.

↑





## COBOL

Label:

symbol

Source module name of one to six alphanumeric characters; only needed when the IN parameter is used.

Parameters:

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \end{array} \right\}$$

Input file definition; also replaces the PARAM IN control statement. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and the file identifier.

**IN=(RES)**

Input is on SYSRES, in \$\$\$SRC.

**IN=(RES, label)**

Input is on SYSRES, but in a file other than \$\$\$SRC.

**IN=(RUN, label)**

Input is on the volume containing the job's \$\$\$RUN file, with a file identifier specified by *label*.

**IN=(\*, label)**

Input volume is identified in the file catalog with a file identifier specified by *label*.

If omitted, source input is in the form of data cards.

## COBOL

$$\text{OBJ} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{RUN}, \text{SYSRUN}) \end{array} \right\}$$

Output file definition; also replaces the PARAM OBJ control statement. The options are:

**OBJ=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**OBJ=(RES, label)**

Output is on SYSRES, with a file identifier specified by *label*.

**OBJ=(RUN, label)**

Output is on the volume containing the job's  $\$Y\$RUN$  file, with a file identifier specified by *label*.

**OBJ=(\*, label)**

Output volume is identified in the file catalog with a file identifier specified by *label*.

$$\text{LIN} = \left\{ \begin{array}{l} (\text{vol-ser-no-label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{RES}, \text{SYS$SRC}) \end{array} \right\}$$

Indicates the location of the copy module. The options are:

**LIN=(vol-ser-no, label)**

Specifies the file identifier (label) and the volume serial number.

**LIN=(RES, label)**

Modules are on SYSRES; the file identifier is specified by *label*.

**LIN=(RUN, label)**

Modules are on the volume containing the job's  $\$Y\$RUN$  file; a file identifier is specified by *label*.

**LIN=(\*, label)**

Modules are on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, modules are on SYSRES in the  $\$Y\$SRC$  file.

**OUT=(p-1, ..., p-n)**

Parameter definitions for the COBOL compiler. See the COBOL supplementary reference manuals for your system for an explanation of these options.

**LST=(p-1, ..., p-n)**

Specifies the format of the compiler listing. See the COBOL supplementary reference manuals for your system for an explanation of these options.



## COBOL

SCR1={vol-ser-no}  
      {RES}

Specifies the volume serial number of the first scratch work file.

SCR2={vol-ser-no}  
      {RES}

Specifies the volume serial number of the second work file. This is ignored in using the basic COBOL compiler.

SCR3={vol-ser-no}  
      {RUN}

Specifies the volume serial number of the third work file. This is ignored in using the basic COBOL compiler.

ALTLOD={ (vol-ser-no, label)  
          (RES, label)  
          (RUN, label)  
          (\*, label)  
          (RES, SYSLOD)  
          (RUN, SYSRUN) }

Specifies the alternate load library that contains the COBOL compiler. The options are:

ALTLOD=(vol-ser-no, label)

Indicates the volume serial number and file identifier (label).

ALTLOD=(RES, label)

Alternate load library is on SYSRES; a file identifier is specified by *label*.

ALTLOD=(RUN, label)

Alternate load library is on the volume containing the job's *\$\$\$RUN* file; a file identifier is specified by *label*.

ALTLOD=(\*, label)

Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, the alternate load library is in *\$\$\$LOD* unless you use LG. Then, the load library is in *\$\$\$RUN*.

↓

## DVCDKT

### Function:

Assigns the same logical unit number to a diskette volume having different files used in a job. The logical unit numbers 130-132 (for diskette) are assigned by OS/3 to the different volumes as they are encountered in the control stream. Once a logical unit number is assigned to a volume, this logical unit number is associated with the volume for the length of the job. Any other specifications of a different logical unit number for this volume within the job is ignored.

### Format:

```
//[ symbol] DVCDKT vol-ser-no[, lun] [,NOVOL={Y}]
```

### Parameters:

vol-ser-no

Volume serial number of the volume.

lun

Logical unit number to be used if different from the default logical unit number assigned by OS/3.

NOVOL={Y}

Suppresses volume serial number checking (NOVOL=Y).

↑

**DVCVOL****Function:**

Assigns the same logical unit number to a disk or volume having different files used in a job. Up to 10 different volumes can be defined in a single job. ←

By default, the logical unit numbers 50-59 (for disk) are assigned by OS/3 to the different volumes as they are encountered in the control stream. Once a logical unit number is assigned to a volume, this logical unit number is associated with the volume for the length of the job. Any other specification of a different logical unit for this volume within the job is ignored.

**Format:**

```
//[symbol] DVCVOL {vol-ser-no}[, lun] [,NOVOL={Y}]
```

RES  
RUN

←

**Parameters:****vol-ser-no**

Volume serial number of the volume.

**RES**

Indicates the SYSRES volume is to be used.

**RUN**

Indicates the volume containing the job's \$Y\$RUN file is to be used.

**lun**

Logical unit number to be used if different from the default logical unit numbers 50-59, which are assigned by OS/3.

**NOVOL={Y}**

Suppresses volume serial number checking (NOVOL=Y). ↓

**NOTES:**

1. The DVCVOL procedure call must be the last statement on a job control card if multistatement coding is used. ↑
2. The LBL and LFD control statements must be present in the control stream after the DVCVOL procedure call. An EXT control statement must also be used when you are allocating the file.

## DVCVTP

### Function:

Assigns the same logical unit number to a tape volume having different files used in a job. Up to 10 different volumes can be defined in a single job.

By default, the logical unit numbers 90-99 are assigned by OS/3 to the different volumes as they are encountered in the control stream. Once a logical unit number is assigned to a volume, this logical unit number is associated with the volume for the length of the job. Any other specification of a different logical unit number for this number within the job is ignored.

### Format:

→ `//[symbol] DVCVTP vol-ser-no[,lun] [ ,PREP={Y} ] [ ,NOVOL={Y} ]`

### Parameters:

`vol-ser-no`

Volume serial number of the volume.

`lun`

Logical unit number to be used if different from the default logical numbers 90-99, which are assigned by OS/3.

`PREP={Y}`

If specified (PREP=Y), any information currently on the volume is effectively erased.

`NOVOL={Y}`

Suppresses volume serial number checking (NOVOL=Y).

### NOTES:

1. The DVCVTP procedure call must be the last statement on a job control card if multistatement coding is used.
2. The LBL and LFD control statements must be present in the control stream after the DVCVTP procedure call.

# FORT

Function:

Generates the job control statements to execute the FORTRAN language processor. Optionally, it can generate the job control statements that specify the following:

- input-source library;
- output-object library;
- PARAM job control statement to define the format of the compiler listing; and
- OPTION job control statements to automatically execute the linkage editor and the generated load module.

The FORTRAN procedure call can be used for basic, extended, or FORTRAN IV versions of FORTRAN:

- Basic - Use the FORT, FORTL, or FORTLG procedure call.
- Extended - Use the FOR, FORL, or FORLG procedure call.
- FORTRAN IV - Use the FOR4, FOR4L, or FOR4LG procedure call.

When L is added to the FOR, FORT, or FOR4 procedure call, an OPTION LINK control statement is generated, and the linkage editor is automatically executed. When LG is added to the FOR, FORT, or FOR4 procedure call, an OPTION LINK,GO control statement is generated, and the linkage editor and the generated load module are automatically executed.

NOTE:

*FOR4, FOR4L, and FOR4LG are the only FORTRAN jprocs supported in System 80.*

When L or LG is used, the OBJ parameter cannot be used to define a specific output library.

Format:

```

//[symbol] { FORT
            { FORTL
            { FORTLG
            { FOR
            { FORL
            { FORLG
            { FOR4
            { FOR4L
            { FOR4LG }
            [ PRNTR= { lun[,dest]
                    { N[,dest]
                    { 20[,dest] } ]
            [ ,IN= { (vol-ser-no,label)
                  { (RES)
                  { (RES,label)
                  { (RUN,label)
                  { (*,label) } ]
            [ .OUT= { (vol-ser-no,label)
                   { (RES,label)
                   { (RUN,label)
                   { (*,label)
                   { NO
                   { (RUN,SYSRUN) } ]
    
```



(continued)

# FORT

[ , SCR1={vol-ser-no} ] [ , ALTL0D={ (vol-ser-no, label) } ]  
 { RES, label }  
 { RUN, label }  
 { (\*, label) }  
 { RUN, \$\$\$RUN }

[ , OPT=(D, N, X) ] [ , MDE=I ] [ , STX=options ]

[ , CNL=k ] [ , LIN={filename} ] [ , LST={k } ]  
 { IBI } { option }

[ , MAP=(S, A, L) ] [ , SIZE={L } ]  
 { S }

[ , ERRFIL=(vol-ser-no, label, module-name) ]

Label:

symbol

Source module name of one to six alphanumeric characters; only needed when the IN parameter is used.

Parameters:

PRNTR={ lun[, dest] }  
 { N[, dest] }  
 { 20[, dest] }

Specifies a printer. The options are:

PRNTR=lun[, dest]

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

PRNTR=N[, dest]

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

IN={ (vol-ser-no, label) }  
 { (RES) }  
 { (RES, label) }  
 { (RUN, label) }  
 { (\*, label) }

Input file definition; also replaces the PARAM IN control statement. The options are:

IN=(vol-ser-no, label)

Indicates the volume serial number and file identifier.

IN=(RES)

Input is on SYSRES, in \$\$\$SRC.

## FORT

IN=(RES, label)

Input is on SYSRES, but in a file other than  $\$Y\$SRC$ .

IN=(RUN, label)

Input is on the volume containing the job's  $\$Y\$RUN$  file, with a file identifier specified by *label*.

IN=(\*, label)

Input volume is identified in the file catalog with a file identifier specified by *label*.

If omitted, source input is in the form of data cards.

OUT= { (vol-ser-no, label)  
(RES, label)  
(RUN, label)  
(\* , label)  
NO  
(RUN,  $\$Y\$RUN$ ) }

Output file definition; also replaces the PARAM OUT control statement. The options are:

OUT=(vol-ser-no, label)

Indicates the volume serial number and file identifier.

OUT=(RES, label)

Output is on SYSRES, with a file identifier specified by *label*.

OUT=(RUN, label)

Output is on the volume containing the job's  $\$Y\$RUN$  file, with a file identifier specified by *label*.

OUT=(\*, label)

Output volume is identified in the file catalog with a file identifier specified by *label*.

OUT=NO

No object code output.

SCR1= { vol-ser-no  
RES }

Specifies the volume serial number of the scratch work file.

ALTLOD= { (vol-ser-no, label)  
(RES, label)  
(RUN, label)  
(\* , label)  
(RUN,  $\$Y\$RUN$ ) }

Specifies the alternate load library that contains the FORTRAN language processor. The options are:

ALTLOD=(vol-ser-no, label)

Indicates the volume serial number and file identifier (*label*).

## FORT

**ALTLOD=(RES, label)**

Alternate load library is on SYSRES; a file identifier is specified by *label*.

**ALTLOD=(RUN, label)**

Alternate load library is on the volume containing the job's  $\$Y\$RUN$  file; a file identifier is specified by *label*.

**ALTLOD=(\*, label)**

Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

→ If omitted, the alternate load library is in the job's  $\$Y\$RUN$  file.

**OPT=(D, N, X)**

Specifies one or all of the following compilation options:

**D**

Specifies double spacing of the compiler listing.

**N**

Specifies that no object program is to be generated. The program units are merely compiled and cannot be executed.

**X**

Specifies compilation of all cards with the character X in column 1. If this option is not specified, these cards will be treated as comments.

This parameter is only supported for the extended and FORTRAN IV compilers.

**MDE=I**

Specifies that the compiler is to evaluate expressions in a strict left-to-right order when there is a choice, and that storage is to be allocated for variables and arrays in the sequence in which they were encountered.

**STX=options**

Used with the extended compiler to execute two STXIT program status word (PSW).

**CNL=k**

Specifies compiler termination if a diagnostic with a severity level is generated. The values for k are:

**2**

Indicates academic messages, e.g., a truncated constant.

**4**

Indicates warning diagnostics, e.g., an extraneous comma in a list.



**FORT**

6

Indicates serious diagnostics, e.g., an array reference without a preceding array declarator.

8

Indicates fatal errors, e.g., insufficient storage to complete the compilation.

If the CNL parameter is not specified, the compiler processes all program units in the control stream, regardless of errors encountered. When specified, the CNL parameter remains in force until redefined.

This parameter is only supported for the extended compiler.

$$\text{LIN}=\left\{\begin{array}{l} \text{filename} \\ \text{lib1} \end{array}\right\}$$

Specifies the name of the default file name in which the source modules reside.

**filename**

Specifies a 1- to 8-alphanumeric-character file name.

This parameter is used with the IN parameter and is only supported for the extended and FORTRAN IV compilers.

LIB1 is the default and is not supported for the FORTRAN IV compiler.

$$\text{LST}=\left\{\begin{array}{l} k \\ \text{options} \end{array}\right\}$$

Specifies the format of the compiler listings.

**LST=k**

Specifies the options for the basic compiler only. The values for k are:

1

Indicates source code listing.

2

Indicates diagnostic listing.

4

Indicates storage allocation map.

7

Is the default value.

8

Indicates object code listing.

Multiple options may be specified by adding the values for k; i.e., to produce both the source code and diagnostic listings (options 1 and 2), specify a 3 for the value of k.

## FORT

### LST=options

Specifies the options for the extended and FORTRAN IV compilers only. The options are:

N

Indicates an abbreviated listing consisting of only the compiler identification, parameters, error counts, and termination conditions.

S

Indicates, in addition to the N listing, the source code listing with any serious diagnostics.

M

Indicates, in addition to the S listing, a storage map showing the addresses assigned to variables and arrays. (For FORTRAN IV, M can be superseded by the MAP parameter.)

W

Indicates, in addition to the M listing, academic and warning messages (extended compiler only).

O

Indicates, in addition to the W listing, an object code showing the instructions generated for the executable statements (extended compiler only).

The default is M for the extended compiler and S for the FORTRAN IV compiler.

### MAP=(S,A,L)

Specifies the type of maps produced by the compiler. One or all of the following options may be chosen:

S

Specifies object summary information, including module size and external subroutines called.

A

Specifies an alphabetical listing of the addresses assigned to variables, arrays, and statement labels.

L

Specifies a listing of the variables, arrays, and statement labels in order by the storage locations assigned.

When a MAP argument is specified, it supersedes the maps selected by the LST parameter. Also, when a MAP argument is specified, it is not necessary to specify LST=M.

This parameter is only supported for the FORTRAN IV compiler.

**FORT****SIZE={ L }  
      { S }**

Specifies the size of the FORTRAN IV compiler. The options are:

**L**

Specifies large version.

**S**

Specifies small version.

**ERRFIL=(vol-ser-no, label, module-name)**

Applies to FORTRAN IV only; specifies an error log file. Error diagnostic messages are written to this file by the error log processor. The *vol-ser-no* parameter specifies the volume serial number, *label* specifies the file identifier, *module-name* specifies the name of the error log file element being created.



# LINK

## Function:

Generates the job and linkage editor control statements needed to execute the linkage editor for the purpose of creating a load module. The object modules to be specifically included in the load module are assumed to be in the job's \$Y\$RUN file; any object code that may have to be automatically included in the load module is assumed to be in \$Y\$OBJ, and the load module produced is assigned the default name LNKLOD and is stored in the job's \$Y\$RUN file. All these default options may be altered by specific parameters. The generated load module can also be automatically executed by specifying the LINKG procedure call statement.

## Format:

```

//[symbol] {LINK } [input-module-name-1, ..., input-module-name-10]
           {LINKG} [ ,PRNTR={ lun[,dest]
                    { N[,dest]
                    { 20[,dest]
                    }
                    }
                    }
                    [ .IN={ (vol-ser-no,label)
                          { (RES)
                          { (RES,label)
                          { (RUN,label)
                          { (*,label)
                          { (RUN,SYSRUN)
                          }
                          }
                          }
                          [ .OUT={ (vol-ser-no,label)
                                { (RES,label)
                                { (RUN,label)
                                { (*,label)
                                { (N)
                                { (RUN,SYSRUN)
                                }
                                }
                                }
                                [ .RLIB={ (vol-ser-no,label)
                                          { (RES,label)
                                          { (RUN,label)
                                          { (*,label)
                                          }
                                          }
                                          }
                                          [ .ALIB={ (vol-ser-no,label)
                                                    { (RES,label)
                                                    { (RUN,label)
                                                    { (*,label)
                                                    }
                                                    }
                                                    }
                                                    [ .SCRI={ vol-ser-no } [ .STD={ YES }
                                                    { RES } { NO }
                                                    ]
                                                    [ .ALTLOD={ (vol-ser-no,label)
                                                            { (RES,label)
                                                            { (RUN,label)
                                                            { (*,label)
                                                            { (RUN,SYSRUN)
                                                            }
                                                            }
                                                            }
                                                            ]
                                                    [ .OPT='options' ] [ .CLIB={ (vol-ser-no,label,modname)
                                                                 { (RES,label,modname)
                                                                 { (RUN,label,modname)
                                                                 { (*,label,modname)
                                                                 }
                                                                 }
                                                                 ]
                                                    [ ,CMT='comments' ] [ ,ENTER=expression ]

```

## NOTES:

1. The LINK procedure call is used to produce a load module from object modules. When the LINKG procedure call is used, an OPTION GO job control statement is generated, which automatically executes the generated load module.
2. When the LINKG procedure call is used, the OUT parameter to define a specific output library cannot be used.

## LINK

Label:

**symbol**

Output load module name of one to six alphanumeric characters; this name is used for LOADM linkage editor control statements. If *symbol* is omitted, the value for the first input-module-name is used for the load module name. If the *input-module-name* parameter is also omitted, the load module name is LNKLOD.

Parameters:

**input-module-name**

Names up to 10 input modules for the generated INCLUDE linkage editor control statements, each name being from 1 to 8 alphanumeric characters in length. If COBOL-68 (extended or basic) is used, you cannot take the 6-character value of *symbol*; you must use all eight positions of the first *input-module-name*, zero-filled to the right if necessary. (You can use the *input-module-name* parameter by itself; the first *input-module-name* truncates to six characters for *symbol*.)

If the *input-module-name* parameter is omitted, the value in the symbol field is assumed to be the name of the object module to be link-edited. If the symbol field is blank, all object modules residing in the job's \$Y\$RUN file are included.

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \\ (\text{RUN, } \$\text{Y\$RUN}) \end{array} \right\}$$

Input file definition; also adds a file name to the generated INCLUDE linkage editor control statement. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

## LINK

IN=(RES)

Input is on SYSRES, in \$Y\$OBJ.

IN=(RES, label)

Input is on SYSRES, but in a file other than \$Y\$OBJ.

IN=(RUN, label)

Input is on the volume containing the job's \$Y\$RUN file, with a file identifier specified by *label*.

IN=(\*, label)

Input volume is identified in the file catalog with a file identifier specified by *label*.

If omitted, the job's \$Y\$RUN file is searched for object modules to include. The file defined in the RLIB parameter is searched next (if the RLIB parameter was specified), and finally \$Y\$OBJ is searched.

$$\text{OUT} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{N}) \\ (\text{RUN}, \text{\$Y\$RUN}) \end{array} \right\}$$

Output file definition for the output load module; also replaces the PARAM OUT control statement. The options are:

OUT=(vol-ser-no, label)

Indicates the volume and serial number and file identifier.

OUT=(RES, label)

Output load module is on SYSRES; a file identifier is specified by *label*.

OUT=(RUN, label)

Output load module is on the volume containing the job's \$Y\$RUN file; a file identifier is specified by *label*.

OUT=(\*, label)

Output volume is identified in the file catalog with a file identifier specified by *label*.

OUT=(N)

Output load module is not written.

$$\text{RLIB} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \end{array} \right\}$$

Renames the file to be searched in place of \$Y\$OBJ during automatic inclusion. This library is searched when no private-library (ALIB) is specified, or when the module being sought is not found in a private library. The options are:

RLIB=(vol-ser-no, label)

Indicates the volume serial number and file identifier.

**LINK****RLIB=(RES, label)**Library is on SYSRES; a file identifier is specified by *label*.**RLIB=(RUN, label)**Library is on the volume containing the job's \$Y\$RUN file; a file identifier is specified by *label*.**RLIB=(\*, label)**Library is on the volume identified in the file catalog with a file identifier specified by *label*.
$$\text{ALIB} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \end{array} \right\}$$

Names an additional library to be used in the automatic inclusion. This library is always searched first during automatic inclusion in an attempt to locate the needed modules. If they are not found, \$Y\$OBJ, or the library named by the RLIB parameter, is searched. The options are:

**ALIB=(vol-ser-no-label)**

Indicates the volume serial number and the file identifier.

**ALIB=(RES, label)**Library is on SYSRES; a file identifier is specified by *label*.**ALIB=(RUN, label)**Library is on the volume containing the job's \$Y\$RUN file, with a file identifier specified by *label*.**ALIB=(\*, label)**Library is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, the library specified by the RLIB parameter or \$Y\$OBJ is searched, in that order.

$$\text{SCR1} = \left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RES} \end{array} \right\}$$

Specifies the volume serial number of the scratch work file.

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

Indicates whether the LOADM and INCLUDE linkage editor control statement should be generated automatically or placed physically in the control stream as data. If YES is used, the proper LOADM and INCLUDE linkage editor control statements are generated between the /\$ and /\* job control statements. If NO is used, no linkage editor control statements are generated; they must be placed physically in the control stream following the procedure call.

## LINK

→

$$\text{ALTLOD} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}) \\ (\text{RES}, \text{label}) \\ (\text{RUN}, \text{label}) \\ (*, \text{label}) \\ (\text{RUN}, \text{\$Y\$RUN}) \end{array} \right\}$$

Specifies the alternate load library containing the linkage editor. The options are:

**ALTLOD=(vol-ser-no, label)**

Indicates the volume serial number and file identifier (*label*).

**ALTLOD=(RES, label)**

Alternate load library is on SYSRES; a file identifier is specified by *label*.

**ALTLOD=(RUN, label)**

Alternate load library is on the volume containing the job's *\\$Y\\$RUN* file; a file identifier is specified by *label*.

**ALTLOD=(\*, label)**

Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

→ If omitted, the alternate load library is in the job's *\\$Y\\$RUN* file.

**OPT='options'**

Specifies the linkage editor options to be in effect.

$$\text{CLIB} = \left\{ \begin{array}{l} (\text{vol-ser-no}, \text{label}, \text{modname}) \\ (\text{RES}, \text{label}, \text{modname}) \\ (\text{RUN}, \text{label}, \text{modname}) \\ (*, \text{label}, \text{modname}) \end{array} \right\}$$

Specifies the location of the source module containing the linkage editor control statements to be processed. RES indicates a library on SYSRES; RUN indicates a library on the volume containing the job's *\\$Y\\$RUN* file; the \* indicates the volume is identified in the file catalog. The file identifier is specified by *label*. The name of the source module is specified by *modname*.

**CMT='comment'**

Indicates the character string that will be inserted in the comment field of each phase header record produced for the generated load module. The comment must be enclosed in apostrophes and may be up to 30 characters in length. It may contain blanks, commas, and other special symbols, excluding apostrophes.

**ENTER=expression**

Specifies the transfer address. The expression may be a decimal number from one to eight digits, a hexadecimal number from one to six digits in the form X'nnnnnn', a previously defined symbol, or a previously defined symbol plus or minus a decimal or hexadecimal number as just described. If this parameter is not specified, control is transferred to the address obtained from the first object module in the phase that had a valid transfer address. If no valid transfer address is found, control is transferred to the first CSECT specifically involved in the phase.

### NOTE:

If the ASM, COBOL, FORT, or RPG procedure call is used previously in the job control stream, the same disk unit assigned in that procedure call must be assigned in this procedure call.



**LNKUPL**

## Function:

Generates the job control statements for the device assignment set needed by the UTS 400 up-line linker routine.

## Format:

```
//ignored LNKUPL [ PRNTR={n } ], FILn={vsn,label,filename}
                   {RES,label,filename}
                   {RUN,label,filename}
```

## Label:

The label will be ignored.

## Parameters:

```
PRNTR={n }
      {20}
```

Specifies the logical unit number for the printer.

```
FILn={vsn,label,filename}
      {RES,label,filename}
      {RUN,label,filename}
```

Generates the device assignment set for the disk pack.



## RPG II

Label:

symbol

Source module name of one to six alphanumeric characters; only needed when the IN parameter is used.

Parameters:

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.

$$\text{IN} = \left\{ \begin{array}{l} (\text{vol-ser-no, label}) \\ (\text{RES}) \\ (\text{RES, label}) \\ (\text{RUN, label}) \\ (*, label) \end{array} \right\}$$

Input file definition; also replaces the PARAM IN control statement. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**IN=(RES)**

Input is on SYSRES, in \$\$\$SRC.

**IN=(RES, label)**

Input is on SYSRES, but in a file other than \$\$\$SRC.

**IN=(RUN, label)**

Input is on the volume containing the job's \$\$\$RUN file; a file identifier is specified by *label*.

**IN=(\*, label)**

The input volume is identified in the file catalog with a file identifier specified by *label*.

If omitted, source input is in the form of data cards.

## RPG II

OUT= { (vol-ser-no, label)  
(RES, label)  
(RUN, label)  
(\* , label)  
(N)  
(RUN, SYSRUN) }

Output file definition; also replaces the PARAM IN control statement. The options are:

OUT=(vol-ser-no, label)

Indicates the volume serial number and file identifier.

OUT=(RES, label)

Output is on SYSRES, with a file identifier specified by *label*.

OUT=(RUN, label)

Output is on the volume containing the job's \$Y\$RUN file, with a file identifier specified by *label*.

OUT=(\* , label)

The output volume is identified in the file catalog with a file identifier specified by *label*.

OUT=(N)

No object code output.

LST= { (K)  
(M)  
(N)  
(S) }

Specifies the format of the compiler listing:

K

Inhibits error flags for sequence errors.

M

Inhibits source and diagnostic listings.

N

Inhibits all listable output.

S

Inhibits main storage map listing.

SCR1= { vol-ser-no  
RES }

Specifies the volume serial number of the first scratch work file.

SCR2= { vol-ser-no  
RUN }

Specifies the volume serial number of the second scratch work file.

## RPG II

EMB= { NO }  
      { YES }

Indicates whether embedded linkage editor control statements are to be generated.

MOD= { 3 }  
      { 4 }  
      { 5 }  
      { IRAM }

MOD=3

Specifies an RPG II compilation in IBM System/3 and System/34 mode. When this mode is specified, IRAM (Series 90) or MIRAM (System 80) will be used to process all disk files; the logical file definition (LFD) for printer files is changed to PRNTR,PRNTR1,...,PRNTRn; and the control reader is used for card input even though the data management reader (READER) is specified.

MOD=4

Same as MOD=3 except that printer files are generated with the same names as used in the program and reader files use the data management reader (READER).

MOD=5

The program is to be compiled in the IBM System 3 and System 34 mode with native mode data management accessing.

MOD=IRAM

Indicates that the IRAM processor is to be used to process all disk files in the program.

ALTLOD= { (vol-ser-no,label) }  
          { (RES,label) }  
          { (RUN,label) }  
          { (\*,label) }  
          { (RUN,\$\$RUN) }

Specifies the alternate load library that contains the RPG II compiler. The options are:

ALTLOD=(vol-ser-no,label)

Indicates the volume serial number and file identifier (label).

ALTLOD=(RES,label)

Alternate load library is on SYSRES; a file identifier is specified by *label*.

ALTLOD=(RUN,label)

Alternate load library is on the volume containing the job's \$\$RUN file; a file identifier is specified by *label*.

ALTLOD=(\*,label)

Alternate load library is on the volume identified in the file catalog with a file identifier specified by *label*.

If omitted, the alternate load library is in the job's \$\$RUN file.

## RPG II

### COL=7

Indicates that RPG II code is to begin in column 7 of the source code (columns 1 through 5 being used for a sequence field and column 6 being left blank).

If omitted, code will begin in column 1.

### CONSOLE

Specifies the lfd-name of a CONSOLE (interactive data entry) file – not a system console file.

### ERRFIL=(vol-ser-no,label,module-name)

Specifies an error log file. Error diagnostic messages are written to this file by the error log processor. The parameter *vol-ser-no* specifies the volume serial number, *label* specifies the file identifier, *module-name* specifies the 1- to 8-alphanumeric-character name of the error file element being created. (Does not need to match or be related to the RPG II source object module name.)

↓  
**SPOOL****Function:**

Controls output spooling. When used, SPOOL must occur within the DVC-LFD sequence for the file to be spooled. Ignored in a nonspooling environment.

**Format:**

```
//[ symbol] SPOOL [ REDIRECT={ DISK
                        TAPE
                        DISKETTE } ] [ , BUF=nXm ] [ , COPIES={ n } ]
                        [ , SKIPCODE={ n } ] [ , RECORDS={ n } ] [ , FORMNAME=forms ]
                        [ , HDR={ NO } ] [ , TESTPAGE={ NO } ] [ , PAGEBRK=n ]
                        [ , UPDATE={ NO } ] [ , COMPRESS={ NO } ] [ , RETAIN={ YES } ]
                        [ , HOLD={ YES } ] [ , SECURE={ YES } ]
                        [ , NO } ] [ , NO } ]
```

**NOTE:**

Only the BUF, RETAIN, UPDATE, COMPRESS, and HOLD keyword parameters can be used for spooled data-set-label diskette output files.

**Parameters:**

```
REDIRECT={ DISK
            TAPE
            DISKETTE }
```

Redirects spooled output (for temporary storage) to a disk, tape, or format-label diskette; output is printed or punched later.

**REDIRECT=DISK**

Redirects spooled output to another disk volume.

**REDIRECT=TAPE**

Redirects spooled output to a tape volume.

**REDIRECT=DISKETTE**

System 80 only. Redirects spooled output to a format-label diskette.

**NOTES:**

1. A DEV operator command must not be in effect (for this copy of the output writer) if you specify REDIRECT=TAPE.
2. With Series 90 systems, you can redirect spooled output to disk only if the system is configured with basic dynamic buffer management. See the system installation user guide/programmer reference, UP-8074 (current version), for information about basic dynamic buffer management.







↓  
**SPOOL****BUF=nXm**

Establishes buffers for use by the spool subfile where  $n$  is the number of buffers,  $X$  is a constant, and  $m$  is the size of each buffer in 256-byte increments. No advantage is gained by making  $n$  greater than 2 unless system symbionts are being used. If this keyword parameter is omitted, the file shares the buffer pool with the job log and other spooled files not having their own buffers.

**COPIES={ n }**  
**1**

Number of times the spooled file is to be printed or punched (output), in a range of 0 through 255. Zero indicates no output. If this keyword parameter is not specified, the file is output once and then deleted from the spool file.

**SKIPCODE={ n }**  
**7**

Used when a filed vertical format buffer having more than seven skip codes is requested (via //VFB) or when the system default buffer has more than seven skip codes. Indicates the number of lines in the buffer that contain skip codes, plus three (one for forms overflow and two for home paper position). Zero indicates no skip codes.

**RECORDS={ n }**  
**5120**

Maximum number of records (lines, including spaces and skipped lines from print files, cards for punch files) that can be placed in the spool file before the operator is queried as to whether the job should be continued or terminated. The largest number you can specify is 262144. If this parameter is omitted, 5120 is assumed.

**FORMNAME=forms**

A 1- to 8-alphanumeric-character name identifying a special printer form or card type to the operator.

**HDR={ NO }**  
**YES**

Suppresses the printing of page headers in burst mode. If omitted, a page header is automatically printed.

**TESTPAGE={ NO }**  
**YES**

Suppresses the query to the operator asking if a sample test pattern page is to be printed when a formname (*FORMNAME=forms*) is detected. If omitted, a query is automatically directed to the operator.

**PAGEBRK=n**

Specifies the number of pages or cards to be spooled out before a file is breakpointed and printed or punched. The largest number you can specify is 32000.



## SPOOL

UPDATE= { NO }  
          { YES }

Indicates that the spool file subdirectory is to be updated only when a file is closed; if the program cancels, any output generated before cancellation is lost. If omitted, the spooler updates the subdirectory entry each time it crosses a logical track in the program file; if the program cancels, output generated before cancellation can be printed.

↓  
↑  
COMPRESS= { NO }  
           { YES }

Prevents the system from attempting to compress data that's directed to the output spool file. Normally, it should not be specified if data in the file contains a large number of embedded blanks or if the file has a block size larger than 120. Specifying COMPRESS=NO when the block size is 121 or greater results in an output spool file containing only one line per sector and requires that nXm be at least 2X4.

RETAIN= { YES }  
          { NO }

The output file is retained in the spool file after it is printed, punched, or placed on data-set-label diskette; the file can be printed, punched, or placed on data-set-label diskette again at a later time. If *RETAIN=YES* is specified with *REDIRECT*, the output file is redirected but is also retained in the spool file for later processing.

HOLD= { YES }  
      { NO }

Spooled output file (print, punch, or data-set-label diskette) is to be held for later processing. Files on hold can be released by a BEGIN SPL command or by a CC job control statement specifying the BEGIN SPL command. If *HOLD=YES* is specified with *RETAIN*, *REDIRECT*, or both, the file is put on hold and *RETAIN* and *REDIRECT* are acted upon when the file is released.

SECURE= { YES }  
          { NO }

Suppresses the printing of the output file if the workstation to which the auxiliary printer is connected is not logged on. If not specified, the file is printed at the auxiliary printer whether or not the workstation is logged on.

UDD

Function:

Generates the job control statements for the device assignment sets needed by the data utility routine to copy or compare one disk, data-set-label diskette, or format-label diskette to another disk, data-set-label diskette, or format-label diskette.

Format 1 (Series 90):

```
//ignored UDD IN= ( {vol-ser-no} .label [ {noext} ] [ .ACCEPT ]
                  {RES
                  {RUN
                  .OUT= ( {vol-ser-no} .label [ {noext} ] [ {ACCEPT
                        {RES
                        {RUN
                        {EXTEND
                        {INIT
                        {RELOD
                        [ ,PRNTR= { lun[,dest] } [ .PUNCH= { NO } } [ .COMPARE= { NO } }
                          { N[,dest] }
                          { 20[,dest] }
                        [ .EXT= ( { DA
                              { IR
                              { NI
                              { MI
                              { IS
                              { SQ
                              [ { C } ] [ { inc } ] [ { addr
                                { F } [ { } ] [ Tccc:hh ] [ { mi
                                { CF } [ { 1 } ] [ BLK
                                { TBLK
                                { CYL
                                { TRK
                                { OLD
                                [ { mj
                                  { (bj,aj) } ] ... ] [ .OLD ]
                                )
```

# UDD

Format 2 (System 80):

```
//ignored UDD IN= ( (vol-ser-no), label [, {noext}] [, ACCEPT] )
                  ( RES
                    RUN )
                  ,OUT= ( (vol-ser-no), label [, {noext}] [, {ACCEPT} ] )
                       ( RES
                         RUN )
                       [ {EXTEND} ]
                       [ INIT ]
                  [, PRNTR= ( Lun [, dest]
                             N [, dest]
                             20 [, dest] )
                  , PUNCH= {YES} , COMPARE= {YES}
                           {NO}   {NO}
                  [, EXT= [ MI ] [ {C} ] [ {inc} ] [ {addr} ]
                         [ {CF} ] [ { } ] [ {Tccc:hh} ]
                         [ {F} ] [ { } ] [ {BLK} ]
                         [ { } ] [ { } ] [ {TBLK} ]
                         [ { } ] [ { } ] [ {CYL} ]
                         [ { } ] [ { } ] [ {TRK} ]
                         [ { } ] [ { } ] [ {OLD} ]
                  [ { {mi} } [ { {mj} } , ... ] [ ,OLD ] [ ,FIX ]
                    [ (bi, ai) ] [ (bj, aj) ] ) ]
```

Label:

The label will be ignored.

Parameters:

**IN=**

Supplies the information needed to define the input file. The information is coded within parentheses.

**vol-ser-no**

Volume serial number.

**UDD****RES**

Input is on SYSRES.

**RUN**

Input is on the volume containing the job's \$Y\$RUN file.

**Label**

File identifier for the input file.

**noext**

Specifies the number of extents in the file. Reserves extent table storage for use by the data access method.

**ACCEPT**

Indicates that the DTF management specifications for data management should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.

**OUT=**

Supplies the information needed to define the output file for a copy operation or the secondary input for a compare operation.

**vol-ser-no**

Volume serial number.

**RES**

Output (or secondary input) is on SYSRES.

**RUN**

Output (or secondary input) is on the volume containing the job's \$Y\$RUN file.

**Label**

File identifier for the output (or secondary input) file.

**noext**

Number of extents in the file. Reserves extent table storage for use by the data access method.

**ACCEPT**

Indicates that the data management specifications should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.

**EXTEND**

Adds information to the end of a sequential (SQ) file.

**INIT**

Initializes the file, starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file close time.

**UDD****RELOD**

Series 90 only. ISAM file is not to be reformatted when it is reloaded.

↓

PRNTR={ lun[,dest] }  
      { N[,dest] }  
      { 20[,dest] }

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

**PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

↑

If omitted, a local printer with a logical unit number of 20 is assumed.

PUNCH={ NO }  
      { YES }

Generates the device assignment set for the punch. This is needed when the punch dual output feature (DC) is selected on the data utility control card.

COMPARE={ NO }  
          { YES }

Indicates that this is a compare operation; will generate the appropriate file name for the secondary input.

**EXT=**

→

Supplies the extent specifications to be used when reserving system resources for a particular file. The information is coded within parentheses. In System 80, all the parameters (except FIX) that apply to disk also apply to format-label-diskette.

**DA**

Series 90 only. Direct access file.

**IR**

Series 90 only. IRAM file.

**NI**

Series 90 only. Nonindexed (DA or SQ) file.

**MI**

→

MIRAM file; only valid specification for System 80 data-set-label diskette.

## UDD

- IS**  
Series 90 only. ISAM file.
- SQ**  
Series 90 only. Sequential file; only valid specification for data-set-label diskette file. ←
- C**  
Allocate contiguous space; only valid specification for a data-set-label diskette file. ←
- F**  
Reformat the file at allocation time. Subparameter 4 must be **BLK**.
- CF**  
Both of the preceding.
- inc**  
Secondary increment (in cylinders) for automatic extension. If there is no *EXT* keyword parameter for this file, the value of the most recently specified secondary increment is used. ↓
- ∅**  
File cannot be dynamically extended; must be specified for data-set-label diskette file. ↑
- addr**  
Absolute beginning cylinder address, in terms of cylinders.
- Tccc:hh**  
Absolute track address (hexadecimal) in cylinder/head format where the file is to begin. Allocation is in terms of tracks.
- BLK**  
Allocation in terms of blocks; only valid specification for data-set-label diskette files. ←
- TBLK**  
Allocates space in blocks; actual allocation is in terms of tracks.
- CYL**  
Allocation in terms of cylinders.
- TRK**  
Allocates space in tracks.

**NOTE:**

*TRK, TBLK, and Tccc:hh cannot be used for ISAM files, indexed IRAM files, or indexed IRAM characteristic MIRAM files.*

**UDD****OLD**

Secondary allocation increment change (subparameter 3) for an existing file; cannot be followed by any other subparameters if specified.

**mi**

Numbers of cylinders or tracks to allocate for this file; subparameter 4 must be *CYL*, *addr*, *TRK*, or *Tccc:hh*.

**(bi,ai)**

Allocation in terms of blocks (by cylinder or track); subparameter 4 must be *BLK* or *TBLK*.

**bi**

Is the average block length.

**ai**

Is the number of blocks to allocate.

Subparameters 6 through n:

Same as subparameter 5 and used to describe additional extents.

Subparameter n+1:

**OLD**

Indicates the file's previously allocated extent is to be increased by the allocation amount (*mi*, *bi*, *ai*, etc) specified.

If OLD is omitted, the request is for a new extent.

**FIX**

System 80 only. Indicates this extent is in the fixed-head area of an 8417 disk.





## UDT

## Function:

Generates the job control statements for the device assignment sets needed by the data utility routine to copy or compare a disk file to a tape file.

## Format:

```
//ignored UDT IN= ( (vol-ser-no), Label [, {noext} ] [, ACCEPT]
                  {
                  RES
                  RUN
                  }
                  ,OUT=(vol-ser-no, Label) [, PRNTR= ( Lun[, dest]
                                                    {
                                                    N[, dest]
                                                    20[, dest]
                                                    }
                  [, PUNCH= { NO } ] [, COMPARE= { NO } ]
```

## Label:

The label will be ignored.

## Parameters:

## IN=

Supplies the information needed to define the input file. The information is coded within parentheses.

## vol-ser-no

Volume serial number.

## RES

Input is on SYSRES.

## RUN

Input is on the volume containing the job's \$Y\$RUN file.

## Label

File identifier for the input file.

## noext

Number of extents in the file. Reserves extent table storage for use by the data access method.

## ACCEPT

Indicates that the data management specifications should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.

**UDT****OUT=**

Specifies the information needed to define the output file for a copy operation or the secondary input for a compare operation.

vol - ser - no

Volume serial number.

Label

File identifier for the output (or secondary input) file.

↓

PRNTR={ lun[,dest] }  
      { N[,dest] }  
      { 20[,dest] }

Specifies a printer. The options are:

PRNTR=lun[,dest]

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

PRNTR=N[,dest]

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

↑

If omitted, a local printer with a logical unit number of 20 is assumed.

PUNCH={ NO }  
      { YES }

Generates the device assignment set for the punch. This is needed when the punch dual output feature (DC) is selected on the data utility control card.

COMPARE={ NO }  
          { YES }

Indicates this is a compare operation; will generate the appropriate file name for the secondary input.

## UPLCNV

### Function:

Generates the job control statements for the device assignment set needed by the UTS 400 up-line conversion routine.

### Format:

```
//ignored UPLCNV [PRNTR={n }],FILn={vsn,label,filename}
                  {20}          {RES,label,filename}
                  {RUN,label,filename}
```

### Label:

The label will be ignored.

### Parameters:

```
PRNTR={n }
       {20}
```

Specifies the logical unit number for the printer.

```
FILn={vsn,label,filename}
      {RES,label,filename}
      {RUN,label,filename}
```

Generates the device assignment set for the disk pack.

# UTD

Function:

Generates the job control statements for the device assignment sets needed by the data utility routine to copy or compare a tape file to a disk format-label diskette, or data-set-label diskette file.

Format 1 (Series 90):

//ignored UTD IN=(vol-ser-no,label),

OUT= ( { vol-ser-no } , label [ { n } ] [ { ACCEPT  
EXTEND  
INIT  
RELOD } ] )

[ , PRNTR= { lun[,dest] } [ , PUNCH= { NO } { YES } ] [ , COMPARE= { NO } { YES } ]  
[ { 20[,dest] } ]

[ , EXT= ( ( DA  
IR  
NI  
MI  
IS  
SQ ) [ { C } ] [ { F } ] [ { CF } ] [ { inc } ] [ { 0 } ] [ { 1 } ] [ { addr  
Tccc:hh  
BLK  
TBLK  
CYL  
TRK  
OLD } ] [ { mi  
(bi,ai) } ] )

[ { {mj  
(bj,aj) } , ... } [ , OLD ] ) ]



**UTD****OUT=**

Supplies the information needed to define the output file for a copy operation or the secondary input for a compare operation.

**vol-ser-no**

Volume serial number.

**RES**

Output (or secondary input) is on SYSRES.

**RUN**

Output (or secondary input) is on the job's \$Y\$RUN file.

**Label**

File identifier for the output (or secondary input) file.

**n**

Number of extents in the file. Reserves extent table storage for use by the data access method.

**ACCEPT**

Indicates that the data management specifications should be obtained from the format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management.

**EXTEND**

Adds information to the end of a sequential (SQ) file.

**INIT**

Initializes the file, starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file close time.

**RELOD**

Series 90 only. ISAM file is not to be reformatted when it is reloaded.

↓

$$\text{PRNTR} = \left\{ \begin{array}{l} \text{lun[,dest]} \\ \text{N[,dest]} \\ \text{20[,dest]} \end{array} \right\}$$

Specifies a printer. The options are:

**PRNTR=lun[,dest]**

Indicates a logical unit number for a printer (*lun*) and optionally a 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote destination for spooled printer output (when dealing with remote batch processing).

↑

**UTD****PRNTR=N[,dest]**

Indicates that you are supplying your own device assignment set for the printer – a DVC-LFD sequence is not to be generated. This option allows use of the // LCB and // VFB statements. A 1- to 6-alphanumeric-character identifier (*dest*) indicating a remote device for spooled printer output may also be specified when dealing with remote batch processing.

If omitted, a local printer with a logical unit number of 20 is assumed.







## UTD

PUNCH={ NO }  
          { YES }

Generates the device assignment set for the punch. This is needed when the punch dual output feature (DC) is selected on the data utility control card.

COMPARE={ NO }  
          { YES }

Indicates this is a compare operation; will generate the appropriate file name for the secondary input.

EXT=

Supplies extent specification to be used when reserving system resources for a particular file. This information is coded within parentheses. In System 80, all the parameters (except FIX) that apply to disk also apply to format-label diskette.

DA

Series 90 only. Direct access file.

IR

Series 90 only. IRAM file.

NI

Series 90 only. Nonindexed (DA or SQ) file.

MI

MIRAM file; only valid specification for System 80 data-set-label diskette.

IS

Series 90 only. ISAM file.

SQ

Series 90 only. Sequential file; only valid specification for data-set-label diskette file.

C

Allocate contiguous space; only valid specification for data-set-label diskette file.

F

Reformat the file at allocation time. Subparameter 4 must be **BLK**.

CF

Both of the preceding.

inc

Secondary increment (in cylinders) for automatic extension. An increment of zero indicates that there can be no dynamic extension of the file. If there is no *EXT* keyword parameter for this file, the value of the most recently specified secondary increment is used.

Ø

File cannot be dynamically extended; must be specified for data-set-label diskette files.

**UTD****addr**

Absolute beginning cylinder address, in terms of cylinders.

**Tccc:hh**

Absolute track address (hexadecimal) in cylinder/head format where the file is to begin. Allocation is in terms of tracks.

**BLK**

Allocation in terms of blocks; only valid specification for data-set-label diskette file.

**TBLK**

Allocates space in blocks; actual allocation is in terms of tracks.

**CYL**

Allocation in terms of cylinders.

**TRK**

Allocates space in tracks.

**NOTE:**

*TRK, TBLK, and Tccc:hh cannot be used for ISAM files, indexed IRAM files, or indexed IRAM characteristic MIRAM files.*

**OLD**

Secondary allocation increment change (subparameter 3) for an existing file; cannot be followed by any other subparameters if specified.

**mi**

Number of cylinders or tracks to allocate for this file; subparameter 4 must be *CYL, addr, TRK, or Tccc:hh*.

**(bi, ai)**

Allocation in terms of blocks (by cylinder or track); subparameter 4 must be *BLK* or *TBLK*.

**bi**

Is the average block length.

**ai**

Is the number of blocks to allocate.

Subparameters 6 through n:

Same as subparameter 5 and used to describe additional extents.

**UTD**

Subparameter n+1:

**OLD**

Indicates the file's previously allocated extent is to be increased by the allocation amount specified (*mi*, (*bi,a*), etc).



If this subparameter is omitted, the request is for a new extent.

**FIX**

System 80 only. Indicates this extent is in the fixed-head area of an 8417 disk.

## WORK/TEMP

### Function:

The WORK and TEMP jproc calls generate any device assignment sets needed for temporary work files. The differences between the two jproc calls are:

- WORK sets up temporary files for one job step and deletes them at the end of the job step.
- TEMP sets up temporary files for the duration of a job and deletes them at the end of the job.
- WORK generates default file identifiers beginning with \$SCRn.
- TEMP generates default file identifiers beginning with \$JOB.

### NOTE:

The file name in your program must match the file identifier generated by WORK or TEMP.

### Format:

```
// [ lfdname ] {WORKn} [ DVC=nn, VOL={vol-ser-no} ] [ {BLK=nnn} ]
                {TEMPn} [ {RES} ] [ {BLK=nnn} ]
                        [ {RUN} ] [ {CYL=nn} ]
                        [ VOL={vol-ser-no} ]
                        [ {RES} ]
                        [ {RUN} ]
                        [ ,EXTSP={nn} ] [ ,SECALL={n} ] [ ,TYPE={file type} ]
```

### NOTE:

The n of WORKn/TEMPn is a number in the range 1 to 10.

### Label:

#### lfdname

File name used when the file is defined by the generated procedure definition. This name corresponds to the name in the LFD control statement.

If omitted, the name generated for WORK1 is \$SCR1; for WORK2, \$SCR2; for WORK3, \$SCR3, etc; the name generated for TEMP1 is \$JOB1; for TEMP2, \$JOB2, etc.

**WORK/TEMP**

Parameters:

$$DVC=nn, VOL=\left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RES} \\ \text{RUN} \end{array} \right\}$$

Specifies the work file device type.

nn

Is the logical unit number of the device.

If this keyword parameter is omitted, the file is assigned to the volume specified by the VOL parameter, or to the device containing the job's  $\$Y\$RUN$  file for even-numbered files (WORK2, TEMP2, WORK4, TEMP4, etc) or the SYSRES volume for odd-numbered files (WORK3, TEMP3, WORK5, TEMP5, etc) if the VOL parameter is omitted. These may both be the same physical volume.

$$VOL=\left\{ \begin{array}{l} \text{vol-ser-no} \\ \text{RES} \\ \text{RUN} \end{array} \right\}$$

Specifies the volume serial number where the temporary file resides.

VOL=RES

Specifies the default if n in WORKn or TEMPn is odd.

VOL=RUN

Specifies the default if n in WORKn or TEMPn is even.

If this keyword parameter is omitted, SYSRES on the volume containing the  $\$Y\$RUN$  file is used.

BLK=nnnn

Specifies the file size.

nnnn

Is the number of 256-byte blocks to be allocated.

Although specified in increments of 256 bytes, the file may be formatted to other block sizes at OPEN time. The size is rounded up to an integral number of cylinders.

If this keyword parameter is omitted and  $CYL=nn$  is also omitted, four thousand 256-byte blocks are allocated to the file.

CYL=nn

Specifies the file size.

nn

Is the number of cylinders to be allocated.

If this keyword parameter is omitted and  $BLK=nnnn$  is also omitted, four thousand 256-byte blocks are allocated to the file.

## WORK/TEMP

EXTSP={nn}  
{16}

Number of extents reserved in the prologue.

SECALL={n}  
{1}

Number of cylinders a file is extended if overflow occurs during a write operation.

TYPE={file type}  
{ST}

Specifies one of the following file types for the extent: DA, IS, IR, MI, NI, or SQ. If this keyword parameter is omitted, the file type is ST(SAT).

**WRTBIG/WRTSML**

## Function:

WRTBIG and WRTSML print block letters on your printed output. They function the same, except that the characters produced by WRTBIG are larger than the characters produced by WRTSML. The following characters may be used:

Letters	A through Z
Digits	0 through 9
Special characters	.( ) + & \$ * - / ? : # =
Embedded blanks	

**NOTE:**

On 48-character printers, some special characters may not print.

## Format:

```
//[symbol] WRTBIG 'block-1' [, 'block-2', ..., 'block-8']
           WRTSML
```

```
[ , IN = { (vol-ser-no, label)
           (RES, label)
           (RUN, label)
           (*, label)
           (RES, $SY$LOD) } ]
[ , LUN = { ( (nnn) [ , { lfdname } [ , dest ] )
            ( 20 ) [ , { PRNTR } ]
            ( N ) } } ]
```

## Parameters:

**WRTBIG**

This option allows block printing of up to 12 characters on each line and up to 4 lines on each page.

**WRTSML**

This option allows block printing of up to 20 characters on each line and up to 6 lines on each page.

**NOTE:**

It may be necessary to use the statement continuation feature of job control when coding your block parameters. Put a nonblank character in column 72 and begin the next line with //n (n is a decimal number in the range 1-9 and is used to keep track of the number of continued lines).

## WRTBIG/WRTSML

### block

Information to be printed on a line. Can be up to 12 or 20 characters long and must be enclosed within apostrophes. Blanks may be used for visual presentation. If the first four characters are coded as 'JOB\$', 'DAT\$', 'TIM\$', or 'VER\$', the job name, system date, system time, or version number of the operating system in use is printed. Each may be specified alone, or with other options.

$$IN = \left\{ \begin{array}{l} (vol-ser-no, label) \\ (RES, label) \\ (RUN, label) \\ (*, label) \\ (RES, SYSLOD) \end{array} \right\}$$

Identifies the file that contains the load module called by the WRTBIG/WRTSML jproc. The options are:

**IN=(vol-ser-no, label)**

Indicates the volume serial number and file identifier.

**IN=(RES, label)**

Load module is on SYSRES, in the file identified by *label*.

**IN=(RUN, label)**

Load module is on the volume containing the job's \$Y\$RUN file, in the file identified by *label*.

**IN=(\*, label)**

Load module is on the volume identified in the file catalog with a file identifier specified by *label*.

↓

$$LUN = \left\{ \left( \left( \begin{array}{l} nnn \\ 20 \\ N \end{array} \right) \left[ \left\{ lfdname \right\} \right] \left[ , dest \right] \right) \right\}$$

Provides a logical unit number (*nnn*) and a file name (*lfd-name*) for the printer. A 1- to 6-alphanumeric-character identifier (*dest*) indicates a remote destination for spooled printer output (when dealing with remote batch processing).

↑

If the job control proc is not to generate a printer device assignment set, code N as the value for this parameter and manually insert a device assignment set. This allows for the insertion of LCB and VFB control statements.



## **4. Proc Definition Statements**



**END**

Function:

End of a procedure definition. The statements between the PROC and END proc definition statements are considered to be the body of the procedure definition, and, therefore, these statements are paired.

Format:

LABEL	△OPERATION△	OPERAND
[ [ // ] symbol ]	END	unused

Label:

symbol

Specifies a 1- to 8-character symbol used as the target address of a branch instruction.

If omitted, the END statement cannot be referenced.

No parameters required.

## NAME

### Function:

Supplies the procedure definition name. It must immediately follow the PROC statement. You may use more than one NAME statement but all must be in the beginning of the definition. Each statement provides a unique name for the same definition. This name provides a reference point within the body and a source code level.

Multiple NAME statements allow you to specify different parameters in the operand fields of each NAME statement; then you may select the desired parameters by calling the proc by that particular name.

### Format:

LABEL	△OPERATION△	OPERAND
[ // ] symbol	NAME	parameter

### Label:

#### symbol

Symbolic name by which the procedure definition may be called. May not be any valid job control statement names (DVC, QGBL, etc).

### Parameters:

#### parameter

Parameter or a parameter sublist to be used at job execution time.

**PROC**

## Function:

Procedure definition start. A valid symbol in the label field is used as a dummy label that represents the *procname* statement label. This dummy label attaches to the expanded procedure definition each time it enters the control stream. The *procname* statement label replaces the dummy label when the procedure definition is called. If the *procname* statement has no label, a null character string is used.

## Format:

LABEL	△OPERATION△	OPERAND
[ [ // ] symbol ]	PROC	[ pos , n ] [ , k , . . . , k ]

## Label:

## symbol

Specifies a 1- to 8-character entry point to this procedure definition when it is expanded and inserted into the control stream. There may be a corresponding label on the *procname* statement.

## Parameters:

## pos

Positional parameter reference symbol in the body of the procedure definition.

Positional parameters specified in a *jproc* call are associated with positional parameters specified in job control statements in the body of the *jproc* definition. The PROC directive specifies the number of positional parameters allowed.

If omitted, no positional parameter substitution is possible.

## n

Decimal number that represents the total number of positional parameters to be found in the procedure call statement.

If omitted, zero is assumed.

**NOTE:**

*If you omit the pos and n parameters, you must still code two commas before you can code any keyword name values.*

## k

Represents the name or names used in referencing keyword parameters contained in the body of the procedure definition.

When a keyword parameter is given a value in the *jproc* definition, it takes that value if the keyword parameter is omitted in the *jproc* call. To preset a keyword value, the *k* parameter takes the form [ , K = value-1 , . . . , K = value-n ] .

If omitted, no keyword parameters are contained in the body of the procedure definition.

## procname

### Function:

Calls a control stream procedure definition. This definition must be either defined and available in \$Y\$JCS or embedded within the control stream and must precede any reference to it. The specified positional parameters are associated with certain individual positional parameters of a job control statement in the procedure definition body. The amount of positional parameters in this procname statement must be equal to the amount in the PROC statement.

If the value of an omitted positional or keyword parameter has been preset in the PROC statement, then the preset value is used in the called procedure definition. If the value has not been preset, then the omitted positional or keyword parameter is set to a null character string. Multiple procedure calls are not allowed on the same line.

### Format:

```
//[ symbol] procname p1,p2,...,pn,ki=vi,kj,...,km=vm
```

### Label:

#### symbol

Dummy label. When specified, this symbol must correspond to the symbol specified in the label field of the PROC statement.

If omitted, the value of the keyword parameter in the label field is null.

### Operation:

#### procname

Procedure definition name. This must be the same name as that specified in the label field of the NAME statement of the called procedure definition.

### Parameters:

#### p1,p2,...,pn

Represent positional parameters.

#### ki=vi,kj,...,km=vm

Represent keyword parameters.

**Appendix A. Logical Unit Number Assignment  
Information**





A device may be designated by a logical unit number at system generation (SYSGEN) time. You may use a standard table or relate the logical unit numbers according to the conventions of the installation. The range of numbers should be as compact as possible to save main storage space. The logical unit numbers are set so that certain ranges of numbers refer to certain categories of devices. These ranges cannot be changed; only the specific entries within the ranges can be changed. Within each category may be several types of devices. For example, tape is a category; SPERRY UNIVAC magnetic tape units are the types within that category. The association of the logical unit number to the category and the type of device is maintained as a 512-byte table on the system resident device (SYSRES). The logical unit number is used as an index into the table. This association may be altered at job execution time via the *EQU* statement.

Each entry in the table is four bytes long. The general format for an entry is as follows:

Byte	0	1	2	3
	device-category	device-type	special-hardware-features	

The following paragraphs summarize the logical unit number values for each device category.

## SPECIAL DEVICE CATEGORY

The value range of logical unit numbers for special devices is 1-15. The standard logical unit number assignments are listed in Table A-1.

The special devices that may be used only with the SPERRY UNIVAC 90/30 System\* are:

- SPERRY UNIVAC 0920 Paper Tape Subsystem
- SPERRY UNIVAC 2703 Optical Document Reader
- SPERRY UNIVAC 9200/9300 Series Subsystem

The byte-bit settings for the 2703 optical document reader and the 0920 paper tape subsystem (read mode) are the same as for the card reader subsystems. The byte-bit settings for the 0920 paper tape subsystem in the punch mode are the same as for the card punch subsystems.

## PRINTERS

The value range of logical unit numbers for printers is 16-29. The values 20 and 21 indicate that any printer may be used. If a specific type of printer is required, the proper logical unit number must be used in assigning the device. The standard logical unit number assignments are listed in Table A-1.

The printer units that may be used with the 90/30 system are:

- SPERRY UNIVAC 0768 Printer Subsystem
- SPERRY UNIVAC 0770 Printer Subsystem

\*Terminology for the 90/30 system includes the 90/25 and 90/40 systems.

- SPERRY UNIVAC 0773 Printer Subsystem
- SPERRY UNIVAC 0776 Printer Subsystem
- SPERRY UNIVAC 0778 Printer Subsystem

System 80 uses the following printers:

- SPERRY UNIVAC 0789 Printer Subsystem
- SPERRY UNIVAC 0776 Printer Subsystem

## CARD READER SUBSYSTEMS

The value range of logical unit numbers for card readers is 30–39. The values 30 and 31 indicate that any reader may be used. If a specific type of reader is required, the proper logical unit number must be used in assigning the device. The standard logical unit number assignments are listed in Table A-1.

The reader units that may be used with the 90/30 system are:

- SPERRY UNIVAC 0716 Card Reader Subsystem
- SPERRY UNIVAC 0717 Card Reader Subsystem

System 80 uses the following card readers:

- SPERRY UNIVAC 0719 Card Reader Subsystem
- SPERRY UNIVAC 0723 Card Reader Subsystem

## CARD PUNCH SUBSYSTEMS

The value range of logical unit numbers for punch units is 40–49. The values 40 and 41 indicate that any punch may be used. If a specific type of punch is required, the proper logical unit number must be used in assigning the device. The standard logical unit number assignments are listed in Table A-1.

The punch units that may be used with the 90/30 system are:

- SPERRY UNIVAC 0604 Card Punch Subsystem
- SPERRY UNIVAC 0605 Card Punch Subsystem

System 80 uses the following card punch subsystem:

- SPERRY UNIVAC 0608 Card Punch Subsystem

## DISK SUBSYSTEMS

The value range of logical unit numbers for disk units is 50-89 and 160-177. The values of 50, 51, 52, and 53 indicate that any disk unit may be used; the specific type of disk unit is not essential to the execution of the program. If a specific type of disk unit is required, the proper logical unit number must be used in assigning the device. The standard logical unit number assignments are listed in Table A-1. Additional characteristics of disk subsystems are presented in Appendix C.

The disk drives that may be used with the 90/30 system are:

- SPERRY UNIVAC 8411 Disk Subsystem
- SPERRY UNIVAC 8414 Disk Subsystem
- SPERRY UNIVAC 8415 Disk Subsystem
- SPERRY UNIVAC 8416 Disk Subsystem
- SPERRY UNIVAC 8418 Disk Subsystem
- SPERRY UNIVAC 8430 Disk Subsystem
- SPERRY UNIVAC 8433 Disk Subsystem
- SPERRY UNIVAC 8424/8425 Disk Subsystem

System 80 uses the following disk subsystems:

- SPERRY UNIVAC 8417 Disk Subsystem
- SPERRY UNIVAC 8419 Disk Subsystem

## DISKETTE SUBSYSTEMS

The range value of logical unit numbers for diskettes is 130-153. The values 130-133 indicate that any diskette may be used. The 90/30 system uses the SPERRY UNIVAC 8413 Diskette and System 80 uses the SPERRY UNIVAC 8420 and 8422 Diskettes.

## MAGNETIC TAPE SUBSYSTEMS

The value range of logical unit numbers for tape units is 90-127. The value of 90 indicates that any tape unit may be used; the specific type of tape unit is not essential to the execution of the program. If a specific type of tape unit is required, the proper logical unit number must be used in assigning the device. The standard logical unit number assignments are listed in Table A-1.

The magnetic tape units that may be used with the 90/30 system are:

- UNISERVO VI-C Magnetic Tape Subsystem
- UNISERVO 12/16 Magnetic Tape Subsystem
- UNISERVO 20 Magnetic Tape Subsystem

- UNISERVO 10 Magnetic Tape Subsystem (This is the only magnetic tape subsystem common to all OS/3 systems.)
- UNISERVO 14 Magnetic Tape Subsystem

## WORKSTATIONS

The value range of logical unit numbers for workstations is 200-219. The values 200-215 indicate that any workstation may be used.

## STANDARD LOGICAL UNIT NUMBERS

Table A-1 provides the standard logical unit numbers and their corresponding device types.

Table A-1. Standard Logical Unit Number Assignments (Part 1 of 2)

Logical Unit No.	Device Type and Features	Device Type Code
1	Reader of 0920/0930 <sup>①</sup>	08020000
2	paper tape subsystem	
3	Punch of 0920/0930 <sup>①</sup>	02020000
4	paper tape subsystem	
5	2703 optical document reader	08010000
6		
7	9200/9300 printer	04080000
8	9200/9300 card reader	08080000
9	9200/9300 card punch	02080000
10	Spare	FFFFFFF
↓		
13		
14	1744 <sup>①</sup> printer, no features specified	04040000
15		
16	0798/0786 printer, no features specified	04010000
17		
18	0789 printer	04020000
19		
20	Any printer, no features specified	04FF0000
21		
22	0773, 0778 <sup>②</sup> printer, no optional features	04400000
23		
24	0776 printer, no optional features	04100000
25		
26	0768 printer, no optional features	04200000
27		
28	0770 printer, no optional features	04800000 <sup>③</sup>
29		
30	Any card reader, no features specified	08F80000
31		
32	0717, 0719 <sup>②</sup> card reader, no features specified	08200000
33		
34	0716 card reader, no features specified	08800000
35		
36	Spare	FFFFFFF
37		
38	0723 card reader, no features specified	08100000
39		
40	Any card punch, no features specified	02F80000
41		

Logical Unit No.	Device Type and Features	Device Type Code
42	0605 card punch, no features specified	02200000
43		
44	0604 card punch, no features specified	02400000
45		
46	0608 card punch	02010000
47		
48	Any remote printer, no features specified	04FF8000
49	Spare	FFFFFFF
50	Any disk	20FF0000
↓		
59		
60	8416/8419 disk subsystem	20100000
↓		
63		
64	8418 MOD-I disk subsystem	20020000
↓		
66		
67	8418 MOD-II disk subsystem	20020004
↓		
69		
70	8430 disk subsystem	20200000
↓		
74		
75	8433 disk subsystem	20200004
↓		
79		
80	8414 disk subsystem	20400000
↓		
85		
86	8411 disk subsystem	20800000
↓		
89		
90	Any tape, no features specified	10FF0000
↓		
99		
100	Any tape, 9-track phase encoded	10FF000A
↓		
102		

Table A-1. Standard Logical Unit Number Assignments (Part 2 of 2)

Logical Unit No.	Device Type and Features	Device Type Code
103 ↓ 105	Any tape, 9-track NRZI	10FF0006
106 ↓ 109	Any tape, 7-track NRZI	10FF0005
110 ↓ 112	Slow tape, 9-track <sup>④</sup> phase encoded	10CB000A
113 ↓ 115	Slow tape, 9-track <sup>④</sup> NRZI	10C80006
116 ↓ 119	Slow tape, 7-track <sup>④</sup> NRZI	10C80005
120 ↓ 122	Fast tape, 9-track <sup>⑤</sup> phase encoded	1034000A
123 ↓ 125	Fast tape, 9-track <sup>⑤</sup> NRZI	10340006
126 127	Fast tape, 7-track <sup>⑤</sup> NRZI	10340005
128 129	Reserved	
130 ↓ 133	Any diskette	40FF0000
134	8413 diskette	40800000
135		
136 137	8420/8422 diskette	40010000
138	Any diskette, 128 byte	400F0001
139		
140	Any diskette, 256 byte	400F0002
141		
142	Any diskette, 512 byte	400F0004
143		
144	Any diskette, 1024 byte	400F0008
145		
146	Spare	FFFFFFFF
147		
148	Double-density diskette	40FF0020
149		
150	Any diskette, autoloader	40FF0100
151		
152	Any diskette, double-sided	40FF0040
153		

Logical Unit No.	Device Type and Features	Device Type Code
154 ↓ 159	Reserved	
160	8415 disk subsystem (fixed)	20080004
161	8415 disk subsystem (removable)	20080000
162	8415 disk subsystem (fixed)	20080004
163	8415 disk subsystem (removable)	20080000
168 169	Any fixed-head disk	201800200
170 ↓ 173	8417 disk subsystem	20080000
174 ↓ 179	Reserved	
180 ↓ 185	8424/8425 disk subsystem	20400004
186 ↓ 199	Reserved	
200 ↓ 215	Any workstation	01FF0000
216 ↓ 219	Any workstation with 24 x 80 screen	01FF0004
220 ↓ 223	Any printer, class=1	04FF0001
224 ↓ 227	Any printer, class=2	04FF0011
228 ↓ 232	Any printer, class=3	04FF0111
232 ↓ 254	Spare	FFFFFFFF
255 256	Reserved	

## NOTES:

- ①NUK only  
 ②Configured with the 90/25 system  
 ③Device type is changed to 04100000 if an 0776 printer is used in place of the 0770 printer.  
 ④UNISERVO 10, UNISERVO 12, UNISERVO VI-C  
 ⑤UNISERVO 14, UNISERVO 16, UNISERVO 20



## **Appendix B. Extent Specification Information**





Files are defined on direct access storage devices in terms of extents. An extent is space on the volume reserved for the file and is made up of contiguous cylinders. Files may be 1 to 16 extents for each volume in the file and may be allocated to any available area on the disk pack. Space on a disk may be allocated on a contiguous or noncontiguous basis, depending on your job requirements. You can request space on a disk in one of the following ways:

- Absolute Address (cylinder or track)

The starting address of your file specified as the absolute address of a cylinder or track. The number of cylinders and tracks required also must be specified.

- Number of Tracks

You can allocate files by specifying the number of tracks needed. This allows you to manage your file space more efficiently than if you allocate files by cylinder.

- Number of Cylinders

You can state the number of cylinders your file needs, rather than stating the exact starting address. Job control places your file in an area that is the best fit with the other files on the disk. Your starting address is placed in the volume table of contents and returned to the job in the file control block. However, you need only request the file by name; the software accesses it for you.

The allocation is in terms of whole cylinders.

- Number of Blocks (by cylinder)

You can state the number of blocks in the file and the average length of the blocks; job control calculates the amount of space needed to fulfill your request. The amount of space assigned varies with the type of disk subsystem used. This method is the most flexible way to allocate extents on a disk since it is not device dependent.

- Number of Blocks (by track)

You can also allocate a file by specifying the number and average length of blocks it requires and have that specification converted to the number of tracks needed. The actual allocation, in this case, is by track.



**Appendix C. SPERRY UNIVAC Disk  
Subsystem Characteristics**



Characteristics	Description						
	8411 Disk Subsystem	8413 Diskette Subsystem	8414 Disk Subsystem	8415 Disk Subsystem	8416 Disk Subsystem	8417 Disk Subsystem ①	8418 Disk Subsystem
Data capacity (8-bit bytes)	7.25 million	242,944 bytes (using tracks 1—73 for data)	29.17 million	33.1 million	28.95 million	118.2 million	28.9 million or 57.9 million
Number of disk units	1 to 8	2 to 4	2 to 8	1 to 2	2 to 8	1 to 8	2 to 8
Disk/diskette speed (rpm)	2400	360	2400	2800	2800	3400	2800
Rotation period (ms/rotation)	25	166.7	25	21.5	21.5	17.6	21.5
Data bit rate (MHz)	1.25	.250	2.5	5.0	5.0	9.05	5.0
Bit density (ppi)	1100	3268	2200	4040 fixed 4040 removable	4040	6366	4040
Track density (tracks/inch)	100 (free format)	48	200	370 fixed 185 removable	192	476	370
Track capacity (bytes/track)	3625	3,328	7294	10,240*	10,240*	15,360	10,240
Number of tracks	200 + 3 spare usable tracks per disk surface	77 total, 73 for data use per disk surface	200 + 3 spare usable tracks per disk surface	808 + 7 spare tracks 404 + 4 spare tracks	404 + 7 spare usable tracks per disk surface	550 + 10 spare tracks per disk surface	404 or 808 + 7 spare usable tracks per disk surface
Number of surfaces per disk unit	10	1	20	Data 3 positioning 1 fixed Data 2 removable	Data 7 positioning 1	14	Data 7 positioning 1
Positioning time (seek time)							
Minimum (ms)	25	—	25	10	10	7	10
Average (ms)	75	83.33	60	33	30	35	27
Maximum (ms)	135	—	130	60	60	70	45
Transfer rate (kilobytes/second)	156	128 bytes in < 6ms	312	625	625	1130	628

\* In fixed 256-byte sectors, 40 sectors per track

Characteristics	Description						
	8419 Disk Subsystem ①	8420/8422 Diskette Subsystem ①		8424 Disk Subsystem	8425 Disk Subsystem	8430 Disk Subsystem	8433 Disk Subsystem
Data capacity (8-bit bytes)	72.39 million	Single density ④ 1 side 303,104 ② 2 sides 606,208	Double density ④ 1 side 563,320 2 sides 1,136,640 ③	58.35 million	58.35 million	100 million	200 million
Number of disk units	1 to 8	1 to 4		1 to 4	2 to 8	1 to 8 (with optional feature up to 16)	1 to 8 (with optional feature up to 16)
Disk/diskette speed (rpm)	2800	360		2400	2400	3600	3600
Rotation period (ms/rotation)	21	166		25	25	16.7	16.7
Data bit rate (MHz)	6.2	—		2.5	2.5	6.45	6.45
Bit density (ppi)	5050	—		2200	2200	4040	4040
Track density (tracks/inch)	—	—		400	400	192	370
Track capacity (bytes/track)	12,800	3328 to 7680 ④		7294	7294	13,030	13,030
Number of tracks	808 + 7 spare usable tracks per disk surface	77 total, 75 ⑤ for data use per diskette surface		400 + 6 spare usable tracks per surface	400 + 6 spare usable tracks per surface	404 + 7 spare usable tracks per disk surface	808 + 7 spare tracks per disk surface
Number of surfaces per disk unit	7	2		20	20	19	19
Positioning time (seek time)							
Minimum (ms)	10	3		10	7.5	7	10
Average (ms)	33	15		30	29	27	30
Maximum (ms)	60	35		55	55	50	55
Transfer rate (kilobytes/second)	784	Dependent on sector sequence arrangement		312	312	806	806

## NOTES:

- ① System 80
- ② 242,944 for data set label BDE (basic data exchange) diskette file.
- ③ 971,776 for format label diskette file
- ④ Maximum value. Actual value is dependent on diskette type (single sided, single density; single sided, double density; double sided, single density; double sided, double density), physical sector size (128, 256, or 512 bytes) and file type (format label or data set label).
- ⑤ 73 for format label diskette file and data set label BDE (basic data exchange) file.  
75 for other data set label non-BDE files.

# Glossary

## A

### **audit file**

Optional file produced by the audit version of the dialog processor. It contains a record of your responses during a dialog session.

### **audit version of the dialog processor**

The version of the dialog processor that produces an audit file.

## C

### **COMREG**

A 12-byte communications region in the *job preamble* available to the user for passing vital information from one *job step* to another. The last byte is the *UPSI* byte (user program switch indicator).

## D

### **direct access method (DAM)**

Allows you to process your disk files in a random or sequential manner, by a matching key or by relative record address.

### **data set delimiters**

The /\$ and the /\* control statements. They should follow the EXEC control statement in the control stream or, if used, any PARAM control statements. When job control detects the /\$ statement, it reads without verifying and places in the job's \$Y\$RUN file all subsequent statements up to and including the /\* statement. The data set within the delimiters may be source code or other control statements.

### **data set labels**

The data set labels are similar in function to a VTOC. They reside on head 0 and describe the types of records and their characteristics that make up the diskette file.

**device assignment sets**

The job control statements used to identify your devices and files to the system. A device assignment set can include these statements: DVC, VOL, EXT, LBL, UID, USE, LFD.

**dialog processor**

An OS/3 system program that manages interactive dialogs, including the *job control dialog*.

**dummy data set**

A dummy data set consists of only a /\$ and /\* statements. They are used with some language jprocs.

**F****file cataloging**

The method of maintaining, in a centralized catalog, all the information needed by a job to access a file that is common to other jobs as well. Also provides protection against unauthorized access or update by using read/write passwords.

**file control block (FCB)**

Contains information needed to control a specific file.

**file identifier**

The physical label of a file. It is specified in the LBL control statement and may be up to 17 characters for a tape file and 44 characters for a disk or format-label diskette file. It is used to locate the *volume table of contents (VTOC)* entry that contains control information for the file.

It is advisable to have the file identifier different from the *file name*.

**file name**

The name of the logical file specified on the LFD statement is considered to be an "internal" file name. It must be the same as the name specified in the label field of the DTF macroinstruction when the file is defined for data management, or in the label field of the DTFF macroinstruction when the physical IOCS function of the supervisor is used. It is this logical file name that you use when you want to access the file.

The LFD file name is used to access the *file control block* associated with this particular file. Job control generates several control tables/blocks; each is made up of related information.

**file processor**

Reads control streams from the card reader and permanently stores them in \$Y\$JCS. The control streams may consist of control statements, procedure definitions, or both, and each control stream must be given a unique name. No validity check is made at the time a control stream is stored in \$Y\$JCS. The file processor is activated by the FILE operator command or the FILE workstation command.

**I****indexed random access method (IRAM)**

Allows processing of logical records by relative record number in a random or a consecutive order or by key (indexed) in a random or a sequential order.



**indexed sequential access method (ISAM)**

Allows disk files to be processed in a random or sequential manner by using an index and a key.

**interactive processing**

A computing environment where you communicate with the operating system through a workstation.

**interstep processor**

Performs end-of-job functions and final housekeeping duties required at the end of a *job step*. These include.

- releasing devices not required by the remaining job steps in the *job*;
- releasing temporary and user-specified disk space;
- storing pertinent logging data;
- calling the *job step processor* (if this is not the last job step); and
- calling the *job termination* function (if this is the last job step).

**J****JC\$BLD**

A system program that uses your responses to the job control dialog to build a job control stream or user jproc. The program is initiated by the SC JC\$BLD workstation command.

**job**

A unit of work to be performed. Each job can be divided into *job steps* (programs) to be executed serially and must be given a unique name.

**job control dialog**

SPERRY UNIVAC supplied dialog that guides you step-by-step through the process of building a job control stream or user jprocs.

**job run library file**

The default job temporary library for most system processors to use for input/output storage.

The *file identifier* for the temporary job run library file is \$Y\$RUN.

**job prologue**

Area reserved at the beginning of the job region in main storage that includes the job preamble, any extent request tables, the task control blocks, open file table, and job accounting table.

**job queue**

Contains an entry for each *job* defined and stored in a \$Y\$RUN file. The entry is filed according to one of three priorities: normal, high, or preemptive.

**job scheduler**

Schedules jobs for execution based on a 3-level priority queue.

**job step**

Unit of work associated with one processing program. A job step is an executable program consisting of one or more tasks that requires a specific amount of the hardware resources of the system.

**job step processor**

Prepares a *job step* for execution. This includes:

- allocating devices for the job step;
- locating and updating the *file control blocks* for this job step;
- posting the disk address of data in the job preamble;
- setting option indicators;
- requesting disk space allocation or file extension;
- storing system logging information;
- performing any requested utility functions; and
- performing housekeeping and bookkeeping functions for the job step as required by the system.

The job step processor terminates when the user program is brought in for execution.

**job termination**

Either the normal or abnormal end of a job. All terminations result in the deallocation of all system resources (such as peripheral devices, main storage, and disk scratch area) previously allocated to the *job*. Any remaining data images or control statements in the control stream are bypassed.

- Normal Terminations

These are initiated by the program, control stream, or system operator.

- Abnormal Terminations

These are caused by program errors, control stream errors, or the expiration of the estimated processing time for the job.

**L****logical unit number**

A number preassigned to a unit by device type and characteristic at each installation. This number is used to access an entry in a control table that contains the type of device being assigned, plus information regarding the use of that device.

## M

### main storage allocation/requirements

The assignment of blocks of main storage to jobs and system functions.

The minimum amount of space need be only that required for the longest load module in your program. Job control calculates how much additional space is required for the various control tables. This establishes the true minimum amount of main storage space that your program uses during its execution.

### multijobbing

The concurrent scheduling, loading, and execution of more than one job at a time. OS/3 job control can initiate up to 7 or 14 jobs concurrently, depending on your system configuration.

### multiple indexed random access method (MIRAM)

Allows processing of logical records by relative record number in a random or a consecutive order or by multiple keys (indexed) in a random or a sequential order. Up to five keys can be searched.

### multitasking

Executing asynchronous multiple tasks within a given job step.

## P

### phase name

The program name plus the 2-digit phase number appended by the linkage editor. The combined program name/phase number is placed in the phase header generated by the linkage editor.

### physical unit block (PUB)

A control block in main storage containing the peripheral device information used by the physical I/O control system (PIOCS).

### program name

The name of the program that is to be executed in the *job step*. This is the name given to the load module by the linkage editor.

## R

### run processor

Identifies, interprets, and analyzes the statements in the control stream. Based on this information, the run symbiont:

- builds the control blocks that describe the job requirements to the system;
- creates a temporary run library for the *job*;
- expands job control procedure calls;
- places an entry for the job in the *job queue* for scheduling; and
- passes control to the job scheduler.

## S

### **scheduling priorities**

The control streams for jobs submitted for execution are queued on the system resident device (SYSRES) by a scheduling priority. This priority is specified on either the JOB statement or the RUN statement and is normal, high, or preemptive. The priority applies to the entire *job* and is considered to be normal unless otherwise specified. High priority should be reserved for emergency scheduling situations and is, therefore, rarely used. Preemptive priority is higher and causes the currently executing job to be preempted. A priority specified on the RUN statement overrides a priority specified on the JOB statement.

### **sequential access method (SAM)**

Allows you to process files with records that follow one another serially. The records are processed one at a time or as block records. Block size is limited to track capacity.

### **spooling**

The process of having the central processor reading or writing records from or to a high speed device, rather than directly from a slower device.

### **summary report**

An optional printed report output by the dialog processor that contains a summary of a dialog session organized by sequentially numbered paragraphs.

### **switching priority**

Determines the order in which central processor control is passed from task to task. The number of user switching priorities varies from 1 to 62. The number of priorities is established at system generation time. The minimum supervisor configuration requires only one priority, regardless of whether this is a *multijobbing* or *multitasking* environment.

### **system access technique (SAT)**

A specialized block level device handler that provides great efficiency in handling disk files.

### **system information block (SIB)**

An area in main storage containing system control information.

### **system job control library file (\$Y\$JCS)**

Provides permanent storage for control streams. It can be used as the output file for the *file processor* and can be input to the *run processor* whether by specification or by default.

### **system library**

A name, generally used to refer to the standard system files in an operating system. In OS/3, the system library includes five permanent system library files and a temporary job run library file for each job input to the system:

- System load library file (\$Y\$L0D)
- System object library file (\$Y\$0BJ)
- System source library file (\$Y\$SRC)
- System macro, or procedure, library file (\$Y\$MAC)

- System job control stream library file (\$Y\$JCS)
- Temporary job run library file (\$Y\$RUN jobname)

The preceding breakdown is defined by the operating system. OS/3 supports different system programs. Library files may be composed of both user and system programs. Such a mix is transparent to the librarian.

### **system load library file**

Provides permanent storage for the executable programs (load modules) supplied as part of the operating system. Can also be used to provide permanent storage for user load modules. This file is used as the input library by the system loader.

The *file identifier* for the system load library file is \$Y\$L0D.

### **system macro library file**

Provides permanent storage for the standard system macro definitions. Can also provide permanent storage for user macro definitions. This file is used as an input library by the assembler macro facility.

The *file identifier* for the system macro library file is \$Y\$MAC.

### **system object library file**

Provides permanent storage for the object modules supplied as part of the operating system. Can also provide permanent storage for object modules of user programs. This file is used as an input library by the linkage editor.

The *file identifier* for the system object library file is \$Y\$OBJ.

### **system source library file**

Provides permanent storage for source modules that consist of source coding as processed by language processors.

The *file identifier* for the system source library file is \$Y\$SRC.

## **T**

### **task**

A point in a program where central processor control is required for continued program execution. Every job step has at least one task, and there may be a maximum of 256 tasks. Also see *switching priority* and *multitasking*.

### **task control block (TCB)**

An area in the job prologue containing information needed to control a specific task.

## **U**

### **user program switch indicator (UPSI)**

Last byte of the 12-byte communication region in the *job preamble*. The two most significant bits of the UPSI byte are used to communicate error conditions to the user.

## V

### **volume serial number**

One to six alphanumeric characters used to identify a physical tape or disk volume. All disk packs and magnetic tape units, including scratch volumes, should be assigned a volume serial number. Each installation may have its own system for numbering volumes.

### **volume table of contents (VTOC)**

Contains the address of all information stored on a direct access volume.

The VTOC is made up of seven different format labels, defined as follows:

- **Format 0 Label**

The format 0 label represents available space within the extent of the VTOC itself. The format 0 label contains all binary 0's. When a format 1, 2, 3, 5, or 6 label is deleted from the VTOC, a format 0 label is written in its place.

- **Format 1 label**

The format 1 label identifies any file on a disk volume except for the VTOC file itself. There must be one format 1 label for each file or part of a file on each volume. Up to three contiguous or noncontiguous areas, or extents, occupied by the file can be identified in the format 1 label. One format 2 label or one format 3 label can be chained to a format 1 label.

- **Format 2 Label**

The format 2 label provides additional description for a format 1 label. If present, the format 2 label is chained to a format 1 label.

- **Format 3 Label**

The format 3 label is used if the file consists of more than three extents. If present, the format 3 label is chained to a format 1 label, format 2 label, or another format 3 label. A format 3 label can describe up to 13 extents.

- **Format 4 Label**

The format 4 label describes the VTOC itself. It is always the first record in the VTOC.

- **Format 5 Label**

The format 5 label describes up to 26 contiguous or noncontiguous extents that are available for allocation on a volume. It is always the second record in the VTOC. Several format 5 labels may be chained together if more than 26 contiguous or noncontiguous extents are available on the volume.

- **Format 6 Label**

The format 6 label describes up to 26 split-cylinder extents. Format 6 labels may be chained together, with the first format 6 label being chained to the format 4 label.

All format labels are described in detail in the data management user guide.

**W****workstation**

A terminal device with a screen for display of system and user information (including dialog text) and a keyboard for user input (including responses to a dialog).





## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

**ATTN.: SYSTEMS PUBLICATIONS**

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD