GC33-5381-2
File No. S370-32

Systems

DOS/VS System Utilities

Release 34

IBM

Summary of Amendments

Independent Component Release of IBM 3800 Printing Subsystem Support

Technical Newsletter GN33-9245 documents changes to support the IBM 3800 Printing Subsystem under DOS/VS.

Release 34

Technical Newsletter GN33-9211 documents:

- List System History (HISTLIST) utility (Chapter 14A)
- Full support of
  - IBM 3350 Direct Access Storage (DOS/VS previously supported the device only in 3330-1 compatibility mode)
  - IBM 3330-11
- Support of IBM 3277 Display Station as operator console
- Support of IBM 3540 Diskette Unit as an IPL communication device

Support of the IBM 3203-4 Printer and the System/370 CPU Models 135-3, 138, 145-3, and 148 did not result in any documentation changes.

Release 33

Edition GC33-5381-2 documents:

- Backup and Restore System utility (Chapter 4)
- Copy File and Maintain Object Module (OBJMAINT) utility (Chapter 9)
- Maintain System History (PTFHIST) utility (Chapter 15)
- Cardless system support

Changes in content are indicated by a vertical bar to the left of the change.

Third Edition (July 1976)

This edition, together with Technical Newsletters GN33-9211 and GN33-9245, applies to Version 5, Release 34, of the IBM Disk Operating System/Virtual Storage, DOS/VS, and to any subsequent versions and releases until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein. Before using this publication in connection with the operation of IBM systems, consult the *IBM System/370 Bibliography,* GC20-0001, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments has been provided at the back of the publication. If the form has been removed, comments may be addressed to IBM Laboratory, Programming Publications Department, Schoenaicher Str. 220, 7030 Boeblingen, Germany. Comments become the property of IBM.

PREFACE

This publication provides a full description of the use of the DOS/VS System Utility Programs for application programmers.

Some of the material in this publication has been previously published as special-purpose utility programs in the SRL publications *DOS and TOS Utility Programs,* GC24-3465, and *DOS Version 4,* GC33-5007.

For an effective use of this publication it is important to understand the

- Organization of the publication.
- Organization of program descriptions.
- Notational conventions.
- Required publications.
- Related publications.

These topics are discussed below.

Organization of the Publication

In addition to the preface you are now reading, a table of contents and a list of figures, this publication has the following parts:

- 'Introduction', which introduces the system utility programs. This chapter lists the devices supported and also contains basic information about how the programs are executed and the control statements used to specify program functions. New or infrequent users of the utility programs should give particular attention to this chapter.
- One chapter for each utility program.
- 'Glossary', which gives an explanation of terms related to DOS/VS.
- 'Index', which is a subject index to this publication.

Organization of Program Descriptions

The descriptions of the programs are all organized the same way. Each program is discussed according to the following pattern:

- Description of the functions that can be performed by the program.
- Utility modifier statement or control statements used by the program with a list of parameters and defaults.
- Control statement streams give examples of how to use the programs, including the job control statements and the utility control statements.

Notational Conventions

A uniform system of notation is used to describe the format of utility control statements.

- Uppercase letters and punctuation marks represent information that must be coded exactly as shown.
- Lowercase letters represent information which is to be supplied by you.
- Parentheses must be coded where shown.
- Stacked options contained within braces ({ }) represent alternatives, one of which must be chosen.
- Brackets ([ ]) indicate that the element they enclose is optional.
- Stacked options contained within brackets ([ ]) represent alternatives of which one may be chosen.
- Options separated by 'or' symbols (|) represent alternatives one of which may (if within brackets) or must (if within braces) be chosen. The 'or' symbol itself is for descriptive purposes only, and should not be coded.
- The letter b always indicates one blank space.

Required Publications

The reader should have a thorough knowledge of the information presented in *DOS/VS System Control Statements,* GC33-5376. This publication describes the system control programs.

See the publication *DOS/VS Messages,* GC33-5379, for a description of the system utility messages.

Related Publications

Introduction to DOS/VS, GC33-5370

DOS/VS Data Management Guide, GC33-5372

DOS/VS IBM 3800 Printing Subsystem Programmer's Guide, GC26-3900, which includes information on the IEBIMAGE utility program and how to use it.

DOS/VS System Generation, GC33-5377

DOS/VS Operating Procedures, GC33-5378

Titles and abstracts of other related publications are listed in the *IBM System/370 Bibliography*, GC20-0001.

TABLE OF CONTENTS

CHAPTER 12

CHAPTER 13

CHAPTER 14

CHAPTER 14A

CHAPTER 15

CHAPTER 16

CHAPTER 17

List of Figures

CHAPTER 1
INTRODUCTION

System utility programs perform particular functions, comprising:

- Assigning an alternate track on a data cell or a disk when a track has been proven defective.
- Clearing one or more areas of a data cell or a disk and establishing preformatted tracks.
- Copying an entire file or volume onto another specified medium and restoring the file or volume to its original medium (copy/restore diskette, copy/restore disk to card, copy/restore disk/data cell to tape, copy disk to disk, fast copy disk volume).
- Initializing a data cell or a disk pack and performing surface analysis.
- Initializing a tape with IBM or ANSI standard volume labels.
- Displaying the VTOC (volume table of contents) of a data cell or a disk pack.
- Printing the hardcopy file from a disk pack.
- Blocking, deblocking, and copying a file; printing job control statements and file contents; selecting jobs from a blocked file.
- Updating object programs.
- Saving the DOS/VS system and private libraries on tape and restoring these libraries to a disk.
- Selecting PTFs from a master PTF file and generating job streams to update the PTF history records in the system.

See the descriptions of the various programs for a more detailed survey of their functions.

The Analysis Program-1 is not described here; it is described in the manual *OS/VS and DOS/VS Analysis Program-1 (AP-1) User's Guide,* GC26-3855. This program enables the operator and the system programmer to determine if a data error on the IBM 3344 or 3350 Direct Access Storage device is a recording surface error or a drive error, since these devices have nonremovable storage.

The IEBIMAGE utility program is not described here; it is described in the *DOS/VS IBM 3800 Printing Subsystem Programmer's Guide*, GC26-3900. This utility enables the user to define, modify, print, or catalog forms control buffer phases, copy modification phases, character arrangement table phases, and graphic character modification phases for the IBM 3800 Printing Subsystem.

The system utility programs process either consecutive or split-cylinder type files. For more information on these file arrangements, see the publication *DOS/VS Supervisor and I/O Macros,* GC33-5373. Indexed-sequential files are not generally supported by these programs; the only exceptions are noted in the Copy and Restore Disk or Data Cell utility. VSAM files are not supported by file to file system utilities; they are supported by Access Method Services. See the publication *Access Method Services User's Guide,* GC33-5382.

The minimum machine configuration required is a System/370 central processing unit with 64K bytes of real storage. See also the section *Storage Requirements.*

Devices

Control Statement Loading

SYSIPT is used for control statement loading and may be assigned to a card reader, tape, disk or diskette extent.

Program Operation

All references to card, data cell, disk, diskette, printer, tape devices refer to all of the devices listed in the respective category, unless stated otherwise in the individual utility descriptions.

Card

IBM 1442 Card Read Punch
IBM 2501 Card Reader
IBM 2520 Card Read Punch
IBM 2540 Card Read Punch
IBM 2560 Multi-Function Card Machine
IBM 3504 Card Reader
IBM 3505 Card Reader
IBM 3525 Card Read Punch
IBM 5425 Multi-Function Card Unit

Data Cell

IBM 2321 Data Cell Drive

Disk

IBM 2311 Disk Storage Drive
IBM 2314 Direct Access Storage Facility
IBM 2319 Disk Storage (as a 2314; see note)
IBM 3330 Disk Storage (Models 1, 2, and 11)
IBM 3333 Disk Storage (as a 3330; see note)
IBM 3340 Direct Access Storage Facility
IBM 3344 Direct Access Storage (as a 3340)
IBM 3350 Direct Access Storage (also as a 3330 Model 1; see note)

Note: References to	Imply the
a) 2314	a) 2319
b) 3330	b) 3333, 3350 in 3330-1 compatibility mode
c) 3340	c) 3344

respectively, unless stated otherwise. Device assignments must be valid for the base device; for example, the program checks for a 3330 value when it analyzes or diagnoses a device assignment for any device supported as a 3330.

Diskette

IBM 3540 Diskette Input/Output Unit

Printer

IBM 1403 Printer
IBM 1443 Printer
IBM 3203 Printer
IBM 3211 Printer
IBM 3800 Printing Subsystem
IBM 5203 Printer

Tape

IBM 2400 Series Magnetic Tape Unit (with or without the 7-track feature) (see note)
IBM 3400 Series Magnetic Tape Unit

Note: The IBM 2495 Tape Cartridge Reader does not belong to the IBM 2400 Series Magnetic Tape Units. The data conversion feature on 7-track tapes may be required.

Logging and Error Messages

Error messages are logged on SYSLST and/or SYSLOG.

Control Statements

Job Control

Job control statements related to channel and unit assignment, label processing, and physical-device description, are used with the utility programs. For information on job control statements, see the publication *DOS/VS System Control Statements,* GC33-5376. Job control requirements specific to a system utility are mentioned in the individual chapter.

The required job control statements to run these programs are summarized in Figure 1.1. The job control parameter values unique to the utility programs are shown in Figure 1.2.

Utility Control

In order to prepare a utility program for a particular run, utility control statements must be submitted in addition to job control statements. Format and usage of the utility control statements are described in each utility chapter.

	ASSIGN ALTERNATE TRACK DISK OR DATA CELL, INITIALIZE DISK OR DATA CELL, INITIALIZE TAPE	CLEAR DISK, CLEAR DATA CELL, COPY/RESTORE DISKETTE	COPY AND RESTORE DISK OR DATA CELL PROGRAMS	VTOC DISPLAY	DEBLOCK	PRINT HARDCOPY FILE (PRINTLOG)
JOB	Required	Required	Required	Required	Required	Required
LBLTYP	Not Used	Not Used	Required for link edit if tape label checking or copying an NSD file.	Required if output is labeled tape.	Not Used	Not Used
TLBL	Not Used	Not Used	Required if tape label processing.	Required if output is labeled tape.	Used for labeled tape output only.	Not Used
DLBL and EXTENT[1]	Not Used	Required for DASD files.	Required for copy file and restore functions. Not used for copy volume function.	Required if output is disk.	Required for DASD files.	Required
ASSGN	Required	Required	Required	Required	Required	Required
UPSI	Required for the 1401/1440 System/370 Emulator (program number SCEML 5745) when initializing disk.	Required for the 1401/1440 System/370 Emulator (program number SCEML 5745) when clearing cylinder 200 of a disk.	Optional	Not Used	Used for unlabeled output tapes only.	Not Used
EXEC	Required	Required	Required	Required	Required	Required
/*	Used only if update records are present.	Required for Copy/Restore Diskette.	Not Used	Not Used	Used for card input.	Not Used
/&	Required	Required	Required	Required	Required	Required

1 A programmer logical unit must be used in the EXTENT statement.

Figure 1.1 **Job control statements used by utility programs**

For Backup and Restore System, Copy File and Maintain Object Module, Fast Copy Disk Volume, List System History, and Maintain System History see the respective chapters.

	ASSIGN ALTERNATE TRACK DISK OR DATA CELL, INITIALIZE DISK OR DATA CELL OR TAPE	CLEAR DISK, CLEAR DATA CELL, COPY/RESTORE DISKETTE	COPY AND RESTORE DISK OR DATA CELL PROGRAMS	VTOC DISPLAY	DEBLOCK	PRINT HARDCOPY FILE (PRINTLOG)
Filename (TLBL or DLBL)	Not Used	UOUT UTEMP for Copy/Restore Diskette.	UIN for input devices, UOUT for output devices, UCHKPT for labeled checkpoint file.	UOUT	UIN for input devices, UOUT for output devices.	IJSYSCN
ASSGN Device for Logging Operator Messages	SYSLOG	SYSLOG	SYSLOG	SYSLOG	SYSLOG	SYSLOG
ASSGN Utility Control Statement Input Device	SYSIPT	SYSIPT	SYSIPT	Not Used	SYSIPT	Not Used
ASSGN Device for Logging Programmer Messages	SYSLST (not applicable to Initialize Tape)	SYSLST	SYSLST	SYSLST	SYSLST	SYSLOG
ASSGN Primary Tape, Card, and Diskette Input and Alternate Tape Input	Not Used	Not Used. SYS004 for Copy/Restore Diskette	SYS004	Not Used	SYS004 for all input devices.	Not Used
ASSGN Primary Tape, Diskette, and Printer Output and Alternate Tape Output	For Initialize Tape only: SYS000 required SYS001–SYS015 optional	Not Used. SYS005 for Copy/Restore Diskette if no intermediate disk is used.	SYS005[1]	SYS005	SYS005 for all output devices.	SYSLST
ASSGN Card Output Device	Not Used	Not Used	SYS006	Not Used	SYS005	Not Used
ASSGN DASD Input and/or Output Device[2]	SYS000	SYS000–SYSnnn SYS005 for Copy/Restore Diskette	SYS000–SYSnnn[3]	SYS004 SYS005	SYS004 SYS005	SYSREC SYSLST

1 For copy volume function also DASD output.

2 SYSnnn can be no greater than the greatest physical unit block assigned and must not conflict with the assignment of any other device.

3 SYS004 is required input assignment for the copy volume function.

Figure 1.2 **Job control statement file names and assignments used by utility programs**

For Backup and Restore System, Copy File and Maintain Object Module, Fast Copy Disk Volume, List System History, and Maintain System History see the respective chapters.

Execution Considerations

All system utilities are distributed in the core image library and in the relocatable library. Exceptions: Assign Alternate Track Data Cell, Clear Data Cell, and Initialize Data Cell are available in the relocatable library only and must therefore be link-edited before they can be executed. The Maintain System History utility has to be link-edited via a job stream generated by assembling the HIST macro. (The macro itself and its use are described in *DOS/VS System Generation,* GC33-5377.)

The system utility programs may be executed in any real or virtual partition. The only exception is the Initialize Data Cell program, which does not run in the foreground partition if the multiple cells option is used.

For examples of control statements used when executing a utility, refer to the individual utility descriptions.

Link-Editing To a Core Image Library

When there is a need to catalog a utility program into a core image library, you must supply linkage editor statements in addition to job control statements. You may either use the link-edit statements given in Module 24 of the manual *DOS/VS System Generation,* GC33-5377, or the cataloged link-edit procedures in Module 1 of the same manual. For a detailed description of linkage editor statements, refer to the publication *DOS/VS System Control Statements,* GC33-5376.

For the Copy Disk or Data Cell utility, the name of the cataloged procedure is LINKDDC; given below are the control statements used to invoke the procedure:

`// JOB LNKUTLY`	User-defined job name.
`// ASSGN SYSLNK,X'cuu'`	Input to the linkage editor.
`// ASSGN SYS001,X'cuu'`	Linkage editor work file.
`// LBLTYP NSD(3)`	Defines the reserved label area in the partition. (3) indicates that the link-edited utility can process DTFIS, DTFDA, or DTFPH files of up to three extents.
`// EXEC PROC=LINKDDC`	Executes the cataloged procedure.
`/&`	Defines the end of the job.

If the utility program is to be cataloged in a private core image library, SYSCLB must be assigned in addition to SYSLNK and SYS001.

Storage Requirements

The system utilities can run in either a real or a virtual partition. The real partition sizes required are given in Figure 1.3. If you run the utility program in a virtual partition, you need not be concerned about these real partition sizes.

Program	Real Partition Size
Assign Alternate Track Data Cell	14K
Assign Alternate Track Disk	24K (Note 2)
Backup System	16K
Clear Data Cell	14K
Clear Disk	24K (Note 2)
Copy Disk to Card	10K (Note 1)
Copy Disk to Disk	10K (Note 1)
Copy Disk/Data Cell to Tape	10K (Note 1)
Copy File and Maintain Object Module (OBJMAINT)	48K
Copy/Restore Diskette	20K
Deblock	16K
Fast Copy Disk Volume	30K (Note 2)
Initialize Data Cell	14K
Initialize Disk	14K (Note 3)
Initialize Tape	10K
List System History (HISTLIST)	64K
Maintain System History (PTFHIST)	64K
Print Hardcopy File (PRINTLOG)	6K
Restore Card to Disk	10K
Restore System	36K
Restore Tape to Disk/Data Cell	10K (Note 1)
VTOC Display	14K

Note 1: When processing very large records, the indicated partition size may not be large enough. Refer to *Storage Requirements* in the respective chapters.

Note 2: This is the requirement for an IBM 3350. For other DASDs the requirement is less, depending on the track capacity.

Note 3: 24K are recommended for an IBM 3350.

Figure 1.3 Real partition sizes for utility program execution

Program	Real Partition Size
Assign Alternate Track Data Cell	14K
Assign Alternate Track Disk	10K
Backup System	16K
Clear Data Cell	14K
Clear Disk	10K
Copy Disk to Card	10K*
Copy Disk to Disk	10K*
Copy Disk/Data Cell to Tape	10K*
Copy File and Maintain Object Module (OBJMAINT)	48K
Copy/Restore Diskette	20K
Deblock	16K
Fast Copy Disk Volume	24K
Initialize Data Cell	14K
Initialize Disk	14K
Initialize Tape	10K
Maintain System History (PTFHIST)	64K
Print Hardcopy File (PRINTLOG)	6K
Restore Card to Disk	10K*
Restore System	30K
Restore Tape to Disk/Data Cell	10K*
VTOC Display	14K

* When processing very large records, the indicated partition size may not be large enough. Refer to *Storage Requirements* in the respective chapters.

Figure 1.4 Minimum real partition sizes for utility program execution

CHAPTER 2
ASSIGN ALTERNATE TRACK DATA CELL

Description

Purposes:

- To assign an alternate track on a data cell, and to copy data from a defective track to an alternate track.

  If an alternate track is found defective, a new alternate track must be assigned to the primary track.

- To recopy data from the alternate track to the primary track if this track is no longer defective.

- To replace faulty records on a specified track if update records are supplied as input.

Assigning an Alternate Track

The program begins by checking the format-4 record of the VTOC (volume table of contents) to determine if there are any remaining tracks in the alternate track area of the data cell from which an alternate track can be assigned. If there are no more available tracks, the job is terminated.

If an alternate track is assigned, the records on the defective track are transferred to the alternate track, beginning with the data portion of R0 (track descriptor record), and continuing with the count, key, and data of R1 through Rn. The record in error is transferred to the alternate track as read. The address of the alternate track is written in R0 of the defective track. This address is used as a pointer to the alternate track. The condition of the defective track may then be analyzed, depending upon your specifications in the utility modifier statement.

When the condition of the track is not analyzed, or when it is analyzed and found defective, the track is flagged defective. The R0 is modified to contain the address of the alternate track.

When the condition of the track is analyzed and found to be nondefective, the track is flagged nondefective. The records are transferred back to the primary track. The VTOC format-4 record is reset to indicate that the alternate track is still available for assignment.

If, after surface analysis of the defective track, the HA (home address) and/or R0 are found defective, the track is flagged defective. It points to

the alternate track by advancing the HA and R0 by 800 bytes. If an error occurs at this time, the data remains on the alternate track and the job is terminated.

Effects of Defective Areas

Defective areas on the defective track may affect the transfer of records to the alternate track. They may also cause the program to end. The possible location of defective areas and their effect on program processing are:

Defective area	Effect
Address marker of a record. Gap preceding the count area of a record. Count area of a record.	The record is bypassed and is not transferred to the alternate track.
Key area of a record. Data area of a record.	The record is transferred to the alternate track exactly as it is read.
Gap following the count area of a record.	The count area is transferred to the alternate track. The key and data areas are filled with A's on the alternate track.
Gap following the key area of a record.	The count and key areas of the record are transferred to the alternate track. The data area is filled with A's on the alternate track.
Gap between the data area of a record and the address marker of the next record.	The records are transferred to the alternate track exactly as they are read.
HA of the track. R0.	The gap between the track index marker and the HA is extended by 800 bytes. An attempt is made to write the HA and R0 following the extended gap. If the HA or R0 cannot be written, the program is terminated (the track cannot be flagged defective).

Record Printing Option

By means of the output option parameter Ox, you can specify in the utility modifier statement whether you want to print all records transferred to the alternate track, or only those that were read in error from the defective track. The records are printed on the device assigned to SYSLST. If an error is found in the HA or R0 of the defective track, all records are printed, regardless of what you specified in the utility modifier statement.

Update Record

By means of the update parameter Ux, the last phase of the program can recognize any number of update records. Each update record in the input stream must be immediately preceded by a TRACK statement. This TRACK statement specifies the location where the update record should be written. The program writes an update record at that location, whether or not a record exists at that location. If the track capacity is exceeded because an update record has been added, the program writes an error message and terminates. In this case the last record on the track may be invalid.

Update records must be in hexadecimal representation, two characters per byte. They must include the key and data portions of the record. Do not imbed insignificant characters in an update record, that is, use every column of each input record until all data has been supplied. Forty bytes of data will fit in one 80-column input record.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// UAT R=(cccchhh),Ox,Ix,Ux
```

The parameters are not positional. The first parameter, R=(cccchhh), must be supplied. The other parameters have the following default values:

```
// U R=(cccchhh),OY,IF,UN
```

`// U` — Utility modifier statement entry.

`AT` — Indicates the Assign Alternate Track Data Cell program. Can be omitted.

`R=` — Indicates the track location parameter.

$(c_1c_2c_2c_3h_4h_5h_5)$ — Indicates the track to which an alternate track will be assigned, or a track whose condition flag is to be changed from defective to nondefective.

c_1	= cell (0-9)
c_2c_2	= subcell (00-19)
c_3	= cylinder (0-9)
h_4	= cylinder (0-4)
h_5h_5	= track (00-19)

These entries must be enclosed in parentheses.

Restriction: Only strip 0-5 may be specified with subcell 19.

`Ox` — Identifies the output option parameter.

`x=Y` — Indicates that all records must be printed as they are transferred to the alternate track.

	x=N	Indicates that only the records found to be in error must be printed.
Ix		Identifies the input option parameter.
	x=F	Indicates that an alternate track is to be assigned without surface analysis.
	x=A	Indicates that surface analysis must be performed on the specified track (or on an alternate, if one is already assigned). If it is a permanent error, an alternate track will be assigned.
	x=U	Indicates that the flag of a defective track must be unconditionally removed and the data recopied onto the primary track. Any error in the transfer of data is communicated to the operator.
Ux		Identifies the update option parameter.
	x=Y	Indicates that one or more update records follow in the input stream.
	x=N	Indicates that no update records follow in the input stream.

TRACK Statement

The format and entries for the TRACK statement are:

```
TRACK=cccchhhhrrkkdddd[S]
```

TRACK= Record location parameter.

$c_1c_1c_2c_2h_1h_1h_2h_2$rrkkdddd[S]

This entry specifies the track location and record description necessary to write the update record. The information must be specified in hexadecimal representation.

c_1c_1	= subcell (00-13)
c_2c_2	= strip (00-09)
h_1h_1	= cylinder (00-04)
h_2h_2	= track (00-13)
rr	= record number (00-FF)
kk	= key length (00-FF)
dddd	= data length (0000-FFFF)

S must be specified if the record must be written using the write special count, key, and data command.

Control Statement Stream

An example of a control statement stream that is used to run an Assign Alternate Track Data Cell program from the core image library:

```
// JOB ALTRK
// ASSGN SYS000,X'293'
// EXEC ALTDC
// UAT R=(2050212),OY,IA
/&
```

An example of using this program to update a record:

```
// JOB UPDATE
// ASSGN SYS000,X'293'
// EXEC ALTDC
// UAT R=(2050212),OY,IA,UY
TRACK=0500020C04000050
       (first 40 bytes of the update record)
       (last 40 bytes of the update record)
/*
/&
```

CHAPTER 3
ASSIGN ALTERNATE TRACK DISK

Description

Purposes:

- To assign an alternate track on a disk device, and to copy data from a defective track to an alternate track.

  If an alternate track is found to be defective, a new alternate track must be assigned to the primary track.

- To replace faulty records on a specified track if update records are supplied as input.

- To change the track-condition indication on an IBM 2311 or IBM 2314, and to recopy data from the alternate track to the primary track.

Assigning an Alternate Track

Specifications in the utility modifier statement identify the defective track. The format-4 record of the VTOC (volume table of contents) on the disk pack identifies the alternate track.

The records from the defective track are transferred to the alternate track, beginning with the data area of R0 (track descriptor record), and continuing with the count, key, and data areas of R1 through Rn.

The VTOC format-4 record is modified to contain the address of the next available alternate track. The condition of the defective track may then be analyzed, depending upon the specifications in the utility modifier statement.

When an IBM 3330, 3340, or 3350 is used, an alternate track is assigned unconditionally. The condition of the track is not analyzed.

When the condition of the track is not analyzed, or when it is analyzed and found defective, the track is flagged defective. R0 is then modified to contain the address of the alternate track. This address is used as a pointer to the alternate track.

When the condition of the track is analyzed and found nondefective, the flag is reset. The records are transferred back to the primary track. The VTOC format-4 record is reset to indicate that the alternate track is still available for assignment.

The program terminates when an alternate track has been assigned, or when the defective track has been found nondefective and has been flagged accordingly.

Effects of Defective Areas

Defective areas on the defective track may affect the transfer of records to the alternate track. They may also cause the program to terminate The possible location of defective areas and their effect on program processing are:

Defective area	Effect
Address marker of a record. Gap preceding the count area of a record. Count area of a record.	The record is bypassed and is not transferred to the alternate track.
Key area of a record. Data area of a record.	The record is transferred to the alternate track exactly as it is read.
Gap following the count area of a record.	The count area is transferred to the alternate track. The key and data areas are filled with A's on the alternate track.
Gap following the key area of a record.	The count and key areas of the record are transferred to the alternate track. The data area is filled with A's on the alternate track.
Gap between the data area of a record and the address marker of the next record	The records are transferred to the alternate track exactly as they are read.
HA of the track. R0.	IBM 2311, 3330: The program is terminated (the track cannot be flagged defective). IBM 3340, 3350: The appropriate Skip Displacement (SD) value, which depends on error location and device type, is used to bypass the defective area and assign an alternate track. If an error persists after use of SD value the program is terminated (the track cannot be flagged defective). IBM 2314: An attempt is made to write the HA and R0 on an alternate track. If the error persists, the program is terminated (the track cannot be flagged defective).

Record Printing Option

By means of the output option parameter Ox, you can specify in the utility modifier statement whether you want to print all records transferred to the alternate track, or only those that were read in error from the defective track. The records are printed on the device assigned to SYSLST. If an error is found in the HA or R0 of the defective track, all records are printed, regardless of what you specified in the utility modifier statement.

Update Record

By means of the update parameter Ux, the last phase of the program can recognize any number of update records. Each update record in the input stream must be immediately preceded by a TRACK statement. This

TRACK statement specifies the location where the update record should be written. The program writes an update record at that location, whether or not a record exists at that location. If the track capacity is exceeded because an update record has been added, the program writes an error message and terminates. In this case the last record on the track may be invalid.

Update records must be in hexadecimal representation, two characters per byte. They must include the key and data areas of the record. Do not imbed insignificant characters in an update record, that is, use every column of each input record until all data has been supplied. Forty bytes of data will fit in one 80-column input record.

Changing the Track Condition Indication

This function is used for tracks that have previously been flagged defective and to which alternate tracks have been assigned. The track that has been flagged defective is specified in the utility modifier statement. The alternate track is identified in the R0 of the specified track.

The program flags the specified track nondefective without analyzing the condition of the track. The records from the alternate track are transferred back to the specified track. The address of the alternate track is removed from R0 of the specified track. Any error encountered during the transfer is communicated to the operator. The flag byte on the alternate track is changed to indicate that the alternate track is not assigned.

Restriction: This flag byte cannot be changed on the IBM 3330, 3340, and 3350.

If this alternate track is the last alternate track that has been assigned, the VTOC format-4 record is changed. It then indicates that this track is now the next alternate track that can be assigned. Otherwise, the VTOC format-4 record is not changed and the alternate track cannot be re-assigned.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// UAT R=(ccchhh),Ox,Ix,Cn,Ux
```

The parameters are not positional. The first parameter, R=(ccchhh), must be supplied. The other parameters have the following default values:

```
// U R=(ccchhh),OY,IF,UN
```

`// U`	Utility modifier statement entry.
`AT`	Indicates the Assign Alternate Track Disk program. Can be omitted.
`R=`	Indicates the track location parameter.
`(ccccchhh)`	Indicates the track to which an alternate track will be assigned, or a track whose condition flag is to be changed from defective to nondefective. cccc = cylinder number (decimal) hhh = head number (decimal) cylinders: for device: 0-199 IBM 2311, 2314 0-403 IBM 3330 (except Model 11) 0-807 IBM 3330 Model 11 0-347 IBM 3340 35MB 0-695 IBM 3340 70MB 0-554 IBM 3350 These entries must be enclosed in parentheses.
`Ox`	Identifies the output option parameter.
`x=Y`	Indicates that all records must be printed as they are transferred to the alternate track.
`x=N`	Indicates that only the records found to be in error must be printed.
`Ix`	Identifies the input option parameter.
`x=F`	Indicates that an alternate track is to be assigned without surface analysis.
`x=A`	Indicates that surface analysis must be performed on the specified track if the track has not been previously flagged. If the primary track has been flagged, then the surface of the alternate track will be analyzed. *Restriction:* This option is not applicable to the IBM 3330, 3340, and 3350.
`x=U`	Indicates that the flag of a defective track

must be unconditionally removed and the data recopied onto the primary track. Any error in the transfer of data is communicated to the operator.

Restriction: This option is not applicable to the IBM 3330, 3340, and 3350.

`Cn` — Identifies the count option parameter.

`n` — If the input parameter is IA: n indicates the number of times that the surface of the track must be analyzed. Enter a decimal value of 1 to 255. The default value is 1. Program run time increases in proportion to the value of n.

If the input parameter is IF or IU, the count option parameter is ignored.

`Ux` — Identifies the update option parameter.

`x=Y` — Indicates that one or more update records follow in the input stream.

`x=N` — Indicates that no update records follow in the input stream.

TRACK Statement

The format and entries for the TRACK statement are:

```
TRACK=ccccchhhhrrkkdddd[S]
```

`TRACK=` — Record location parameter.

`ccccchhhhrrkkdddd[S]`

This entry specifies the track location and record description necessary to write the update record. The information must be specified in hexadecimal representation.

cccc	= cylinder number
hhhh	= track number
rr	= record number (00-FF)
kk	= key length (00-FF)
dddd	= data length (0000-FFFF)

S must be specified if the record must be written using the write special count, key, and data command.

Control Statement Stream

An example of a control statement stream that is used to run the Assign Alternate Track Disk program:

```
// JOB ALTRK
// ASSGN SYS000,X'191'
// EXEC ALTDK
// UAT R=(0027003),IA,C3,OY
/&
```

An example of using this program to update two records:

```
// JOB UPDATE
// ASSGN SYS000,X'192'
// EXEC ALTDK
// UAT R=(0004002),IA,C3,OY,UY
TRACK=0004000203000050
       (first 40 bytes of the update record)
       (last 40 bytes of the update record)
TRACK=00040002040001E0
       (first 40 bytes of the update record)
       (last 40 bytes of the update record)
/*
/&
```

CHAPTER 4
BACKUP AND RESTORE SYSTEM

Backup and Restore are two separate programs that produce a device-independent backup tape of the system and/or private DOS/VS libraries and permit restoration of this copy to all DASD devices supported by DOS/VS as SYSRES. The programs have the following possible applications:

- The Backup program alone can be used to create a backup copy of a DOS/VS system for future use.
- The Restore program alone can be used to restore such a backup copy of the DOS/VS system to disk.
- The Restore program alone can also be used to restore the DOS/VS distribution tape to disk prior to system generation.
- Backup and Restore together can be used to transfer DOS/VS libraries from one type of disk device to another type.
- Backup and Restore together can also be used to condense the DOS/VS libraries.

These programs do not support the IBM 5425 Multi-Function Card Unit. The IBM 2311 Disk Storage Drive may be used to store temporarily the stand-alone version of the Restore program; it is, of course, not supported by DOS/VS as a SYSRES device.

The two programs are described separately in the following sections.

The Backup Program

Description

Purposes:

- To create a backup copy on tape of the system files and of private libraries in a format suitable for later use with the Restore utility.
- To create a stand-alone backup tape, that is, a tape complete with the necessary Supervisor, Job Control, and Restore programs for restoring the files and libraries without the use of DOS/VS.

The program writes to tape the stand-alone Restore program (if required), the IPL records (only if the system libraries are being saved), the system core-image library (CIL), the system relocatable library, the system source

statement library, the procedure library, the private core image library, the private relocatable library, and the private source statement library.

The program is located in the core image library under the name BACKUP and as a relocatable module in the relocatable library under the name IJWSABK.

The Backup Tape

Generation of a backup tape suitable for use with DOS/VS Restore or of a tape suitable for stand-alone use is controlled by the assignment of SYS004 during the execution of Backup.

If you want to produce a backup tape for the DOS/VS Restore program, SYS004 must be unassigned. The first record on the output tape is the INFO record. The INFO record contains an entry for each library which is allocated or assigned; it thus indicates which system/private libraries are present on the tape. The INFO record is preceded by two dummy tape marks.

If you want to produce a backup tape that is suitable for stand-alone restore operations, you must include the stand-alone version of the Restore program on the tape. The stand-alone version consists of the following: Supervisor, Job Control, and the Initialize Disk utility; Job Control again and the Restore utility. (How to obtain the stand-alone version from the system source statement library is described in *Obtaining the Stand-Alone Restore Program.)* You assign SYS004 to read the stand-alone version of the Restore program; the data read is placed in front of the INFO record, in the form of two files.

The tape is closed when all libraries have been written. If there is insufficient space on one volume for the file, a multi-volume file is created. Each time before a library is dumped to tape, the program tests whether the partition is large enough to accommodate two I/O areas. If this is not the case, only one I/O area is used, thus causing a performance decrease.

The program issues messages informing the user of the progress of the backup operation and of any error conditions which may occur. These messages are documented in *DOS/VS Messages,* GC33-5379.

Obtaining the Stand-Alone Restore Program

Two basic methods are available for obtaining the stand-alone version of the Restore utility in order to include it on the backup tape.

Without POWER/VS Support

In this case, one of the following job streams is used to punch out the file from the source statement library onto SYSPCH:

If SYSPCH is a card punch or a tape drive

```
// JOB PUNCH STAND ALONE RESTORE
// ASSGN SYSPCH,X'cuu'
// EXEC SSERV
   PUNCH Z.BPSFILES
/*
/&
```

If SYSPCH is a disk drive

```
// JOB PUNCH STAND ALONE RESTORE
// DLBL IJSYSPH,'file-id'
// EXTENT SYSPCH,volserno,1,0,address,tracks
// ASSGN SYSPCH,X'cuu'
// EXEC SSERV
   PUNCH Z.BPSFILES
/*
/&
```

If SYSPCH is a diskette unit

```
// JOB PUNCH STAND ALONE RESTORE
// DLBL IJSYSPH,'file-id'
// EXTENT
// ASSGN SYSPCH,X'cuu'
// EXEC SSERV
   PUNCH Z.BPSFILES
/*
/&
```

Note: Include ASSGN statements for SYSIPT, SYSLST, and SYSLOG if the current assignments are not those required.

The output of the job stream is a file suitable for input to the Backup program via SYS004. The first two cards or records in the output are a CATALS and a BKEND statement; the last two cards or records are a BKEND and a /* statement. It is not necessary to remove these statements before the file is used as input to the Backup program, as they will be ignored by the program.

With POWER/VS Support

The POWER/VS control statement SLI (Source Library Inclusion) makes it possible to include the stand-alone Restore utility directly from the source statement library without having to punch out the cards beforehand. In this case, one statement must be added to the normal Backup jobstream:

```
// JOB GENERATE STAND ALONE RESTORE TAPE
// ASSGN SYS004,X'cuu' (spooled card reader)
   .
   .
   .
// EXEC BACKUP
* $$ SLI Z.BPSFILES  <====
/&
```

In this case, POWER/VS logically inserts the file into the jobstream on the reader. This means that no further handling is necessary.

`// JOB name`	Required.
`// ASSGN SYS004,X'cuu'`	Required. Either X'cuu' or UA must be specified. If X'cuu' is used, the program expects the stand-alone utility file to be on the specified card, tape, disk, or diskette unit.
`// ASSGN SYS005,X'cuu'`	Optional. Input unit for the system residence files.
`// ASSGN SYS006,X'cuu'`	Required. Output unit for the backup tape.
`// ASSGN SYS007,X'cuu'`	Optional. Input unit for the private core image library.
`// ASSGN SYS008,X'cuu'`	Optional. Input unit for the private relocatable library.
`// ASSGN SYS009,X'cuu'`	Optional. Input unit for the private source statement library. If a library is not to be restored, the corresponding system logical unit has to be unassigned.

`// DLBL IJSYSRS,'file-id'`
`// EXTENT SYS005,volserno,`
`1,0,addr,tracks`
	These statements must be used if no partition or standard label information is present for the file.

`// DLBL IJSYSCL,'file-id'`
`// EXTENT SYS007,volserno,`
`1,0,addr,tracks`	

`// DLBL IJSYSRL,'file-id'`
`// EXTENT SYS008,volserno,`
`1,0,addr,tracks`	

`// DLBL IJSYSSL,'file-id'`
`// EXTENT SYS009,volserno,`
`1,0,addr tracks`	

`// DLBL BPSFILE,'file-id'`
`// EXTENT SYS004,volserno,`
`1,0,addr,tracks`
	These statements must be used if SYS004 is a disk or diskette. For diskette // EXTENT SYS004 is sufficient. BPSFILE is the name of the DTF block in the Backup program.

`// MTC REW,SYS006`	Optional. This statement must be used if the output tape is to be rewound before the program writes on the tape as the Backup program does not do any rewinding.
`// EXEC BACKUP`	Required. The program name is BACKUP.
`,SIZE=nnK`	Optional. The minimum size is 16K with one I/O area and 24K with two I/O areas.
`// MTC RUN,SYS006`	Optional. Rewind and unload the output tape.
`/&`	Required.

The Restore Program

Description

Purposes:

- To restore a DOS/VS system from a tape created by the Backup program.
- To restore the DOS/VS distribution tape to disk.

The Restore program restores the IPL records and any of the DOS/VS libraries (system and/or private) which are present on the input tape, depending on the device assignments and allocations supplied.

If the DOS/VS version of the Restore utility is used, the allocations for the libraries can be supplied by means of an ALLOC statement in the input job stream or by means of console responses to prompting messages; in the case of the stand-alone version, only console responses are permissible.

Program Versions

There are two versions of the Restore program, one of which operates in a batch partition of a running DOS/VS system, and a second which can be used without an operational DOS/VS system.

Note that this second stand-alone version of the program must have been generated on the backup tape by the Backup program. In the case of distribution tapes, the stand-alone Restore is already present on the tape.

Running the DOS/VS Version

As mentioned above, the DOS/VS version of the Restore utility runs in a batch partition of an operational DOS/VS system. The name of the program is RESTORE, and its operation is controlled by the following job control statements:

`// JOB jobname` — Required.

`// UPSI`

`1` — Optional.
Indicates that the input of allocations will be done via SYSLOG, and that the program must issue the necessary prompting messages.
The permissible replies to these messages are identical to the parameters of the ALLOC statement, and are described below.
If the operator replies with an EOB (END or ENTER) to the allocation prompting messages, the default allocations from the backup tape will be used. UPSI 1 will also cause a prompting message for a disk label for SYSRES and each private library being restored. An EOB response to this message causes the default label to be used. This default is DOS.SYSRES.FILE, or DOS.SYSxxx.FILE, where xxx is CLB, RLB, or SLB.

`11` — Optional.
In this case, the default allocations from the Backup tape will be used for SYSRES. No prompting messages will be issued during restoration of SYSRES.

`Other` — Invalid; the job will be canceled.

If the UPSI statement is included, any ALLOC statements or commands provided will be ignored by the program.

If the UPSI statement is omitted, the program expects the allocations to be provided by means of the ALLOC statements in the job stream.

`// ASSGN SYS005,X'cuu'` — Optional.
Output unit for the system residence file.
When using the DOS/VS version, SYS005 must not be assigned to the same physical device as the current SYSRES.

`// ASSGN SYS006,X'cuu'` — Required.
Input unit for the backup tape.

```
// ASSGN SYS007,X'cuu'
```

Optional.
Output unit for the private core image library.

```
// ASSGN SYS008,X'cuu'
```

Optional.
Output unit for the private relocatable library.

```
// ASSGN SYS009,X'cuu'
```

Optional.
Output unit for the private source statement library.

Note that at least one output unit has to be specified.

```
// DLBL IJSYSRS,'file-id'
// EXTENT SYS005,volserno,
   1,0,addr,tracks
// DLBL IJSYSCL,'file-id'
// EXTENT SYS007,volserno,
   1,0,addr,tracks
// DLBL IJSYSRL,'file-id'
// EXTENT SYS008,volserno,
   1,0,addr,tracks
// DLBL IJSYSSL,'file-id'
// EXTENT SYS009,volserno,
   1,0,addr,tracks
```

Optional.
These statements are needed only if the UPSI statement is omitted. The statements define the files for the libraries to be restored. 'addr' is the starting address of the file and 'tracks' the number of tracks the file will occupy. For the SYSRES file, 'addr' must be 1 and 'tracks' is the actual number of tracks (minus 1 for the IPL record); this value must include one extra cylinder (two if the device is an IBM 3340) to be used as label cylinder(s).

```
// MTC REW,SYS006
```

Optional.
This statement must be used if the input tape is to be rewound before the program reads from the tape, as the Restore program does not do any tape positioning.

```
// EXEC RESTORE
```

Required.
The program name is RESTORE.

```
   ,SIZE=nnK
```

Optional.
The minimum size of the partition is 28K to 36K depending on the device used.

```
   ALLOC
```

Optional.
This statement is required if the UPSI byte was not set.

```
   CL=PVT|ccc(tt)
   [,RL=ccc(tt)|Eccc(tt)]
   [,SL=ccc(tt)|Eccc(tt)]
   [,PL=ccc(tt)|Eccc(tt)]
   [,PC=ccc(tt)]
   [,PR=ccc(tt)]
   [,PS=ccc(tt)]
```

The abbreviations, the possible allocations, and the defaults are described in the section *Allocations*, below. That section also contains a summary of the file names.

/*	Required.
// MTC RUN,SYS006	Optional. Rewind and unload the output tape.
/&	Required.

Allocations

The basic allocation format is ccc(tt), where ccc is the number of cylinders to be allocated to the library and tt is the number of tracks within these cylinders which are to be reserved for the directory.

If you specify CL=PVT, all system libraries (except a procedure library) present on the tape will be restored as private libraries.

If you specify an allocation of 0 (0) for any library, this library will not be restored.

If you specify Eccc(tt) (only possible for RL, SL, and PL), the requested amount of space will be allocated, but no entries will be made. This results in the generation of an empty library.

Identifier	Library	Logical Unit	File Name
CL	System Core Image Library	SYS005	IJSYSRS
RL	System Relocatable Library	SYS005	IJSYSRS
SL	System Source Statement Library	SYS005	IJSYSRS
PL	System Procedure Library	SYS005	IJSYSRS
PC	Private Core Image Library	SYS007	IJSYSCL
PR	Private Relocatable Library	SYS008	IJSYSRL
PS	Private Source Statement Library	SYS009	IJSYSSL

Figure 4.1 **Library identifiers, file names, and logical units**

For allocations via SYSRDR:
You must specify the allocation for each library you want to restore. If you omit the allocation for any library, this library will not be restored.

When restoring system libraries as private libraries, the allocations for the three system libraries (CL, RL, SL) must be given with the private library identifiers PC=, PR=, PS=. Any private libraries present on the tape will be bypassed in this run.

For allocations via SYSLOG:
The program will prompt you with appropriate messages for the allocations (ALLOC=) and also for the starting tracks (START=) for any of the private libraries. If you reply EOB when prompted for an allocation, the default allocation from the backup tape is used. The default allocations on the backup tape are device-independent and are converted to suit the device being used by the Restore program.

When restoring system libraries as private libraries, you will be prompted for the allocations for the three system libraries (CL, RL, SL) with the private library identifiers PC=, PR=, PS=. After restoring the system

libraries as private libraries, the program will prompt you for allocations for any private libraries present on the tape; and it will again use the same prompting messages. This allows the system libraries and the private libraries to be processed in the same Restore run. You can reply with EOJ to any allocation prompting message, and thus cause the job to be terminated.

Running the Stand-Alone Version

As already mentioned above, the stand-alone version of the Restore program exists only on a DOS/VS distribution tape or on a backup tape created by the Backup utility. The handling of the stand-alone version of the Restore program is identical to the handling of the DOS/VS distribution tape. For details on this, and for examples of the jobstream, refer to *DOS/VS System Generation*, GC33-5377.

CHAPTER 5
CLEAR DATA CELL

Description

Purposes:

- To clear one or more areas on a data cell.
- To establish preformatted tracks throughout the areas cleared.
- To create a file label in the VTOC (volume table of contents).

The area to be cleared can be as small as one track or as large as a complete data cell. Any number of areas can be cleared with one run.

When an area of a data cell is cleared, fixed-length blocks containing count, key, and data areas are established on the data cell. The count area is generated with

cylinder number (2 bytes)
head number (2 bytes)
record number (1 byte)
key length (1 byte)
data length (2 bytes)

The information which defines the key and data areas is indicated in the utility modifier statement. If this statement is omitted, default values are assumed.

The defined key and data areas are filled with a character defined by you in the C'c' (or X'xx') parameter of the utility modifier statement. However, the first eight bytes of the data area of R0 (track descriptor record) will contain:

cylinder number (bytes 1-2)
head number (bytes 3-4)
record number, always zero (byte 5)
number of unused bytes on the track (bytes 6-7)
binary zero (byte 8).

The DLBL statement used in this program must indicate a sequential file.

The labels are checked to determine whether the area to be cleared contains all or part of an unexpired file. Expired labels for the area to be cleared are deleted from the VTOC. For the preformatted tracks a file label is written in the VTOC of the data cell.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// UCM B=(K=n,D=n),{ C'c' },Ox,E=(e)
                   { X'xx' }
```

If this statement is omitted, or if one or more parameters are omitted, the following default values are assumed for the non-specified parameters:

```
// UCM B=(K=0,D=100),X'00',OY,E=(2321)
```

`// U`	Utility modifier statement entry.
`CM`	Indicates the Clear Data Cell program. Can be omitted.
`B=`	Identifies the key and data block parameter.
`(K=n,D=n)`	Indicates the length (in bytes) of the key and data block. These entries must be enclosed in parentheses.
`C`	Identifies the EBCDIC fill character parameter.
`'c'`	EBCDIC fill character. This entry must be enclosed in apostrophes.
`X`	Identifies the hexadecimal fill character parameter.
`'xx'`	Hexadecimal fill character. This entry must be enclosed in apostrophes.
`Ox`	Identifies the output option parameter.
`x=Y`	x=Y (write data cell check) is the only option accepted by the program. Any other specification is ignored.
`E=`	Identifies the device type parameter.
`(e)`	The valid entry for output devices is 2321. This entry must be enclosed in parentheses.

END Statement

Must be supplied to signal the end of the utility control statements.

The format is:

```
// END
```

Control Statement Stream

An example of a control statement stream that is used to run the Clear Data Cell program from the core image library:

```
// JOB CLRDC
// ASSGN SYS008,X'193'
// DLBL UOUT,'FILE LABEL',73/300
// EXTENT SYS008,000012,1,0,05000,00100,B=4
// EXEC CLRDC
// UCM B=(K=0,D=900),C'X'
// END
/&
```

CHAPTER 6
CLEAR DISK

Description

Purposes:

- To clear one or more extents on a disk device.
- To establish preformatted tracks throughout the extents cleared.
- To create a file label in the VTOC (volume table of contents).

The area to be cleared can be as small as one track or as large as a complete disk pack. Any number of areas can be cleared with one run.

When an area of a disk is cleared, fixed-length blocks containing count, key, and data areas are established on the disk. The count area is generated with

cylinder number (2 bytes)
head number (2 bytes)
record number (1 byte)
key length (1 byte)
data length (2 bytes).

The information which defines the key and data areas is indicated in the utility modifier statement. If this statement is omitted, default values are assumed.

The defined key and data areas are filled with a character defined by you in the C'c' (or X'xx') parameter of the utility modifier statement. However, the first eight bytes of the data area of R0 (track descriptor record) will contain:

cylinder number (bytes 1-2)
head number (bytes 3-4)
record number, always zero (byte 5)
number of unused bytes on the track (bytes 6-7)
binary zero (byte 8).

The DLBL statement used in this program must indicate a sequential file.

The labels are checked to determine whether the area to be cleared contains all or part of an unexpired file. Expired labels for the area to be cleared are deleted from the VTOC. For the preformatted tracks a file label is written in the VTOC of the disk.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// UCL B=(K=n,D=n),{C'c'}   ,Ox,E=(e)
                   {X'xx'}
```

The parameters are not positional. If this statement is omitted, or if one or more parameters are omitted, the following default values are assumed for the non-specified parameters.

```
// UCL B=(K=0,D=100),X'00',OY,E=(2311)
```

`// U`	Utility modifier statement entry.
`CL`	Indicates the Clear Disk program. Can be omitted.
`B=`	Identifies the key and data block parameter.
`(K=n,D=n)`	Indicates the length (in bytes) of the key and data block.
`C`	Identifies the EBCDIC fill character parameter.
`'c'`	EBCDIC fill character. This entry must be enclosed in apostrophes.
`X`	Identifies the hexadecimal fill character parameter.
`'xx'`	Hexadecimal fill character. This entry must be enclosed in apostrophes.
`Ox`	Identifies the output option parameter.
`x=Y`	Indicates write disk check.
`x=N`	Indicates do not write disk check.
`E=`	Identifies the device type parameter.
`(e)`	The valid entries for output devices are 2311, 2314, 3330, 3340, and 3350. This entry must be enclosed in parentheses.

END Statement

Must be supplied to signal the end of the utility control statements.

The format is:

```
// END
```

Control Statement Stream

An example of a control statement stream that is used to run the Clear Disk program:

```
// JOB CLRDK
// ASSGN SYS012,X'191'
// DLBL UOUT,'DISK LABEL',77/300
// EXTENT SYS012,001221,1,0,00310,00630
// EXEC CLRDK
// UCL B=(K=38,D=480),X'55',ON
// END
/&
```

Note for 1401/1440 System/370 Emulation:
When you clear cylinder 200 of an IBM 2311 or 2314 disk pack to be used by the 1401/1440 System/370 Emulator, you must include an UPSI statement before the EXEC statement in the control statement stream. This UPSI statement must have the following format:

```
// UPSI 00000001
```

By means of this UPSI statement, cylinder 200 is cleared together with the area specified on the EXTENT statement. Cylinder 200 must not be specified on the EXTENT statement.

If only part of the disk pack must be cleared and not cylinder 200, you must not include the UPSI statement.

Restriction: You cannot use the UPSI statement for the IBM 3330, 3340, and 3350.

CHAPTER 7
COPY AND RESTORE DISK OR DATA CELL

Description

Purposes:

- To copy a volume or a file from a disk device to another disk of the same type.
- To copy a volume or a file from a disk device to cards or tape and to restore the data later to a disk of the type from which it was originally copied.
- To copy a volume or a file from a data cell to tape and to restore the data later to a data cell.

If you want to copy a complete volume from an IBM 3330 Disk Storage, IBM 3340 Direct Access Storage Facility, or an IBM 3350 Direct Access Storage Facility either directly to a disk device of the same type, or via tape for intermediate storage, you should consider using the Fast Copy Disk Volume utility, which performs faster.

The Copy and Restore Disk or Data Cell utility can process complete volumes as well as individual files. The desired function, copy file or copy volume, is specified in the utility modifier statement. You can copy data directly from one disk volume to another or you can copy data from disk or data cell onto another medium for intermediate storage and, later restore it to the device type from which it was originally copied. Due to this functional variety the Copy and Restore Disk or Data Cell utility really consists of three copy programs and two restore programs; they are:

Copy disk to disk
Copy disk to card
Restore card to disk
Copy disk/data cell to tape
Restore tape to disk/data cell.

Restrictions:

- Volume copy from the IBM 3330 Model 1 to the Model 11 is not supported. Support of this function is provided by the Fast Copy Disk Volume utility.
- An ISAM file cannot be copied onto an IBM 3330 Model 11.

During transfer, the data area of R0 (track descriptor record), and the count, key, and data areas of records R1 through Rn are processed. These areas are necessary to restore the data so that it will be identical to the original volume or file.

The output created by the copy programs, with the exception of the copy disk to disk program, can only be used by the restore programs. The System/370 versions of these programs do not accept input created by the System/360 versions.

Storage Requirements

The Copy and Restore Disk or Data Cell utility can be executed in a real or in a virtual partition. If the program is executed in a real partition, you must take into account that all copy and restore programs require a buffer large enough to contain all records of a track. Figure 7.1 shows the minimum real partition sizes for records of maximum size (=track capacity).

Device	Maximum record size in bytes	Problem program partition (real)
IBM 2311	3,625	10K
IBM 2314	7,294	14K
IBM 2321	2,000	10K
IBM 3330	13,030	20K
IBM 3340	8,368	14K
IBM 3350	19,069	26K

Figure 7.1 Maximum record size and problem program partition per device

Checkpoint and Restart Facility

The copy and restore programs provide checkpoint and restart facilities through DOS/VS. See the publication *DOS/VS Data Management Guide,* GC33-5372, for more information on this subject.

Checkpoint information may be taken at the beginning of every eightieth track or at the beginning of each new extent. This information must be written on a disk or tape separate from the input and/or output device for the particular job. SYS003 is the unit to which you must assign the file on which you want checkpoint information to be written. To do this you must include the following job control statement:

```
// ASSGN SYS003,X'cuu'
```

To ignore the checkpoint facility make sure that SYS003 is unassigned. To do this you must use the following job control statement:

```
// ASSGN SYS003,UA
```

The size of the checkpoint records taken depends on the record size specified in the A=(a) parameter of the utility modifier statement. It also depends on the amount of available storage.

Considerations for checkpoint records written on disk are given in the publication *DOS/VS Supervisor and I/O Macros,* GC33-5373.

Message 80681
CHECKPOINT BEING TAKEN FOLLOWING CARD NO. xxxxxx
is printed before a checkpoint is taken during the copy disk to card program. Then, supervisor message 0C001
CHKPT NO. xxxx WAS TAKEN ON SYSxxx=cuu
is printed. (SYSxxx=cuu refers to the logical and physical unit on which the checkpoint information has been stored. This number must be included in the RSTRT STATEMENT.) All cards following the card referenced in message 80681 must be deleted before you try to restart.

You must reload all cards following the card referenced in message 80681 in the card reader, when you run the restore card to disk program.

The name of the checkpoint file is UCHKPT. This file name must be used in the TLBL and DLBL statements when standard labels must be processed. The format is:

```
// TLBL UCHKPT  or  // DLBL UCHKPT
```

The file name must also be used in the RSTRT statement when the checkpoints are on disk. The format of the RSTRT statement is

```
// RSTRT SYS003,nnnn,UCHKPT
```

where nnnn must be identical to the checkpoint number xxxx from message 0C001.

Copy Programs

The copy programs can transfer data from

- disk to cards
- disk to disk
- disk to tape
- data cell to tape.

There are two program functions:

- The *copy volume* function transfers data from one disk pack to cards, disk, or tape. It may also transfer data from one data cell to tape. The complete volume is transferred, including the two IPL records, the volume label(s), and the VTOC (volume table of contents).
- The *copy file* function transfers one data file from disk to cards, disk, or tape. It may also transfer one data file from data cell to tape. You must supply the DLBL and the EXTENT statements to describe the file you want to be copied. For multi-extent files: if you did not copy all extents you must nevertheless specify all extents (as present in the VTOC) when you want to restore the extents that were copied. Whether copying or restoring, the file described by the job control statements

is transferred to the same extent limits from which it was copied. The copy file function can also be used for a mutiple-extent sequential disk file on more than one disk pack, if each extent is assigned.

The eopy programs assume standard R0s with a data length of eight bytes. A pack with a nonstandard R0 cannot be copied; the disk must first be initialized and the file re-created.

Sequential, indexed-sequential, and direct access file organizations are supported by this group of programs.

I/O overlap may be possible if the channel assignment permits it, and if sufficient storage is available to process two complete tracks at a time.

Opening a Disk Pack or Data Cell

For the copy file function, standard label checking is performed by the OPEN macro. You must supply the necessary information in DLBL and EXTENT statements.

For information about the output file on disk or data cell, refer to the sections *Opening a Disk Pack or Data Cell for Restoring a File* and *Opening a Disk Pack or Data Cell for Restoring a Volume*, respectively.

Processing User Standard Labels on DASD Files

User standard labels on sequential and direct access files on a disk pack or data cell can be copied by the copy file function, if you supply an UPSI job control statement.

The user standard labels are only copied if you assign the value 1 to bit 7 of the UPSI byte. The UPSI statement has the following format:

```
// UPSI 00000001
```

Bits 0 and 2 are only significant for tape label processing. See the section *Opening the Tape Volume*. All other bits of the UPSI byte are not significant to the copy disk or data cell programs.

If you did not specify the UPSI statement, the program assumes that there are no user standard labels present, or that those labels present must not be copied and restored.

When a file that was copied with a // UPSI 00000001 statement is consequently restored, you must reset bit 7 to 1 to ensure that the restore extents are identical to the copy extents.

Opening the Tape Volume

Tape files containing either no labels or IBM standard labels can be processed.

Nonstandard labels are not supported by the copy and restore programs.

When label processing must be performed, you must set bits 0 and 2 as follows (0 equals off, 1 equals on):

bit 0 off = standard input label checking
on = no input label checking

bit 2 off = standard output label checking
on = no output label checking.

Note: The output tape will be rewound to the load position before writing takes place.
If you want standard label checking:

- an UPSI statement is optional (the copy and restore programs assume standard label checking)
- a TLBL statement is required.

If you do not want standard label checking:

- an UPSI statment is required
- a TLBL statement is not required.

Utility Modifier Statement

Contains information required to run the copy programs. The parameters are not positional.

The format and entries are:

```
// UCR Tt,A=(a),CELLS=(n,n,n,n,n),N=(n),On,
       IPL,E=(e),Mx
```

`// U`	Utility modifier statement entry.
`CR`	Indicates the copy program.
`Tt`	Identifies the function type parameter.
`t=F`	Copy file function.
`t=V`	Copy volume function.
`A=`	Identifies the physical record length parameter.
`(a)`	Indicates the most common physical record length of the area to be copied. The relationship between this length and the actual record length can affect the performance, depending on how accurate this length is.

If performance is not important, any valid record length may be specified.

CELLS=	Identifies the CELLS parameter. Is only used for data cell operations.
(n,n,n,n,n)	Each n is a decimal digit from 0 to 9, designating a cell number. From one to five cell numbers may be included. The order in which the cell numbers are given in this parameter is the order in which they will be processed by the program.
N=	Identifies the number-of-volumes parameter for the copy file function.
(n)	A decimal number from 1 to 10, specifying the number of volumes in the file to be copied. The default value is (1).
On	Identifies the I/O area parameter. Can only be used for the copy to tape program.
n=1	When you specify 1, one I/O area is assigned by the program, regardless of available storage. There will be no overlap in the I/O operations.
n=2	When you specify 2, the I/O assignments are determined by the program. If sufficient real or virtual storage is available, two I/O areas are assigned for I/O overlap. When the On parameter is omitted, the program assumes that O2 was specified.
IPL	This parameter is optional for the copy file function. If present, the IPL records are copied and restored. The parameter is invalid for the copy volume function.
E=	Identifies the device type parameter.
(e)	The valid entries are 2311, 2314, 2321, 3330, 3340, and 3350. If this entry is omitted, the default value is 2311. This entry must be enclosed in parentheses.

For the copy disk to tape program - copy volume function - the device type is generated from the PUB (physical unit block) table; the E=(e) parameter is ignored.

`Mx` — Identifies the file management parameter.

This parameter can only be used for the copy-file function.

`x=S` — Indicates sequential file management.

`x=D` — Indicates direct file management.

`x=I` — Indicates indexed-sequential file management. You must specify all extent limits for the entire indexed-sequential file, that is, master index, cylinder index, and overflow areas.

If this parameter is omitted, the default value is **MS**.

Input Processing

The performance of the program is highly dependent on the value supplied for the A=(a) parameter. This value may not exceed the maximum record size (track capacity).

To copy files with fixed-length records, the A=(a) parameter should be equal to the physical size (key length + data length). In the case of variable record length, or when copying a complete volume, the most common record size should be supplied for the A=(a) parameter. You must observe the following considerations to obtain maximum performance:

- The number of chained records that are read is obtained from the A=(a) parameter. On the assumption that all records on the track have a record size equal to the record size of the A=(a) parameter, the number of records per track is calculated. This number is taken as the number of chained Channel Command Words read. If the actual number of records on the track is larger, more than one operation per track is necessary.
- If the actual record size of a record on the track is larger than the record size in the A=(a) parameter, control is given to a routine that reads the records one by one.

If a record is read that exceeds the available I/O area, the size of the actual record is retained. When the record is restored, the record maintains its original format. During restoring, the section of the record that could not be read is then filled with binary zeros. A message is given which identifies this condition during the copy program.

Performance may be decreased when the tracks are not filled to their maximum capacity. For example, this may occur when the direct access method is used.

When you use the copy to tape program, each tape reel is rewound and unloaded when processing is completed for that tape.

Control Statement Stream

This section shows examples of control statement streams.

1. Copy Disk to Tape (copy volume)

```
// JOB COPY
// ASSGN SYS004,X'191'
// ASSGN SYS005,X'180'
// ASSGN SYS003,UA (no checkpoints)
// UPSI 00100001   (copy user standard labels)
// EXEC CDKTP
// UCR TV,A=(1600)
/&
```

2. Copy Disk to Card (copy file)

```
// JOB COPY
// ASSGN SYS005,X'191'
// ASSGN SYS006,X'00D'
// ASSGN SYS003,UA (no checkpoints)
// DLBL UIN,'DISK FILE',76/100
// EXTENT SYS005,111111,1,0,00011,00050
// EXEC CDKCD
// UCR TF,A=(1600)
/&
```

3. Copy Disk to Disk with indexed-sequential files (copy file)

```
// JOB COPY
// ASSGN SYS006,X'191'
// ASSGN SYS005,X'192'
// ASSGN SYS003,UA
// DLBL UIN,'INDEXED SEQUENTIAL',,ISE (for copying)
// EXTENT SYS006,333333,4,1,10,2 (cylinder index)
// EXTENT SYS006,333333,1,2,20,30 (prime data area)
// EXTENT SYS006,333333,2,3,12,8 (overflow area)
// DLBL UOUT,'INDEXED SEQUENTIAL',,ISC (for restoring)
// EXTENT SYS005,222222,4,1,10,2
// EXTENT SYS005,222222,1,2,20,30
// EXTENT SYS005,222222,2,3,12,8
// EXEC CDKDK
// UCR TF,A=(1800),MI
/&
```

4. Copy Disk to Disk (copy volume)

```
// JOB COPY
// ASSGN SYS004,X'191'
// ASSGN SYS005,X'192'
// ASSGN SYS003,UA (no checkpoints)
// DLBL UOUT
// EXTENT SYS005,222222,1,0,00010,01990
// EXEC CDKDK
// UCR TV,A=(1800)
/&
```

5. Copy Data Cell to Tape (copy volume)

```
// JOB COPY
// ASSGN SYS003,X'283'
// TLBL UCHKPT
// ASSGN SYS004,X'293'
// ASSGN SYS005,X'282'
// TLBL UOUT
// UPSI 10100
// EXEC CDKTP
// UCR TV,E=(2321),A=(80),CELLS=(5)
/&
```

Restore Programs

The restore programs can restore data from

- cards to disk
- tape to disk
- tape to data cell.

This depends on the method used in the corresponding copy program.

There are two program functions:

- The *restore volume* function transfers one volume of data from tape or cards to a disk pack. It may also transfer data from tape to data cell.

  The IPL records, volume label(s), and VTOC are automatically restored, together with the data. Both restore programs modify those areas in the VTOC format-4 label that are unique to the volume, that is, alternate track information.

- The *restore file* function completes the transfer of data so that the restored file is identical in location and format to the file at the time it was copied.

When you run the restore program, make sure to use a disk pack or data cell that has been initialized according to IBM standards.

Specific parameters that are unique to the program are not needed, because the output created by the programs contains all the control information necessary to execute the corresponding restore program.

Processing disk or data cell output is dependent upon the processing performed in the corresponding copy program. That is, when restoring, a partition of greater capacity will not improve the performance of the programs.

The exact I/O assignment made in the corresponding copy program is assigned for restoring.

Opening a Disk Pack or Data Cell for Restoring a File

The restore programs perform standard label checking using the OPEN macro. You must supply the DLBL statement and the EXTENT statement. They are used to create the label for the file that you want to be restored. The extent limits must be the same as the extent limits you supplied to the copy program.

If you specified user standard label processing in the copy program, you must reset bit 7 of the UPSI byte to 1 when the file is restored. The UPSI statement has the following format:

```
// UPSI 00000001
```

Opening a Disk Pack or Data Cell for Restoring a Volume

The restore programs open the disk pack as a sequential file using the OPEN macro. You have to supply the DLBL and the EXTENT statements. See the publication *DOS/VS Supervisor and I/O Macros*, GC33-5373.

The information of the DLBL and EXTENT statements is used to determine the area of the volume to be searched for unexpired labels. This information must indicate a sequential file. The file name in the DLBL statement must be UOUT:

```
// DLBL UOUT
```

The extent provided through job control for restoring a volume must be as large as possible. It must, however, not include track zero on cylinder zero, or the VTOC. An unexpired file that overlaps this extent may be deleted if desired.

The new VTOC is automatically restored from tape to data cell, or from tape or cards to disk. The following fields, which are dependent upon the physical volume, are modified:

- Format-4 label, field 5 (highest alternate track) is modified to indicate the highest alternate track assigned on the volume.
- Format-4 label, field 6 (number of alternate tracks) is modified to indicate the number of alternate tracks available on the volume.

If you specified user standard label processing in the copy program, you must reset bit 7 of the UPSI byte to 1 when the file is restored. The UPSI statement has the following format:

```
// UPSI 00000001
```

Opening the Tape Volume(s)

Tapes files containing either no labels or IBM standard labels can be processed.

Nonstandard labels are not supported by the copy and restore programs.

When label processing must be performed, you must set bits 0, 2, and 7 of the UPSI statement as follows (0 equals off, 1 equals on):

bit 0 off = standard label input checking
on = no input label checking

bit 2 off = standard output label checking
on = no output label checking

bit 7 off = no user standard labels
on = restore user standard labels (only valid for card or tape files from copy programs).

If you want standard label checking:

- an UPSI statement is optional (the copy and restore programs assume standard label checking)
- a TLBL statement is required.

If you do not want standard label checking:

- an UPSI statement is required
- a TLBL statement is not required.

If you specified label processing when copying onto tape, you must also specify label processing when restoring from tape.

Closing the Tape Volume(s)

Each tape reel is rewound and unloaded when processing is completed for that tape. If you specified an alternate tape drive, the program alternates between the primary drive and the alternate drive, until processing is completed for the program.

At the end of each unlabeled tape, the restore program requests the operator to decide whether the tape is at the end of the file or at the end of the volume.

Control Statement Stream

Under the section *Control Statement Stream* for the copy programs, this example was given of how to copy a volume from disk to tape:

```
// JOB COPY
// ASSGN SYS004,X'191'
// ASSGN SYS005,X'180'
// ASSGN SYS003,UA (no checkpoints)
// UPSI 00100001 (copy user standard labels)
// EXEC CDKTP
// UCR TV,A=(1600)
/&
```

To restore this same volume from tape to disk, you must use the following job stream:

```
// JOB RESTORE
// ASSGN SYS004,X'180'
// ASSGN SYS005,X'191'
// ASSGN SYS003,UA (no checkpoints)
// DLBL UOUT
// EXTENT SYS005,111111,1,0,00010,01990
// UPSI 10000001 (restore user standard labels)
// EXEC RTPDK
/&
```

CHAPTER 8
COPY AND RESTORE DISKETTE

Description

Purposes:

- To replace faulty labels on a diskette.
- To copy the entire contents of a diskette onto another diskette.
- To eliminate the 'special records' from all data files.
- To create a backup copy.

Execution of the program is controlled by the following statements:

- utility modifier statement
- file descriptor statement
- // END statement (required to start the copy operation)
- /* statement (required to close the files).

Assignments

SYS004 must be assigned to the input diskette.

If the copy operation is to be performed directly from diskette to diskette, SYS005 must be assigned to the output diskette. If the copy operation is to be performed via intermediate disk, SYS005 must be assigned to the disk.

Input Diskette

The program reads the VOL1 label on track 0 of the input diskette. If the VOL1 label on the input diskette cannot be read, you must supply label information on the utility modifier statement.

If the input diskette contains faulty file descriptor labels, you can specify replacement information on file descriptor statements.

Output Diskette

For output, you must use a diskette without defective sectors on track 0 and without protected files.

You are strongly advised to use a new or error-free diskette for output.

If the input does not supply sufficient information to make a successful copy possible, the program will cancel after printing track 0 (volume and file information) together with an error indication. Complete track 0 information must be present to start copying.

Intermediate Disk

If only one diskette drive is available, a disk device must be used as an intermediate temporary storage device. You must provide the DLBL and EXTENT statements for a file (labeled UTEMP) that is large enough to contain all data records from the input diskette. The program then copies the input diskette to the temporary file and issues a message to the operator to mount an output diskette on SYS004.

Utility Modifier Statement

Contains information to run the Copy and Restore Diskette program. The statement is optional, except when the VOL1 sector on the input diskette is defective.

The format and entries are:

```
// UCD [TV],[VOL=xxxxxx]
```

The parameters are not positional.

// U	Utility modifier statement entry.
CD	Indicates the Copy and Restore Diskette program.
TV	Identifies the copy function. Full volume copy is the only option. This parameter can be omitted.
VOL=	Identifies the volume parameter. Only required when the input VOL1 label is defective.
xxxxxx	Identifies the volume. Normally, a numerical value from 000001 to 999999, although any or all of the characters may be alphabetic.

If a utility modifier statement with a valid VOL parameter is supplied, the specified volume is checked against the volume mounted on SYS004.

If the VOL1 label on the input diskette cannot be read, the VOL parameter on the utility modifier statement is required to provide the program with a volume identifier for the output diskette.

File Descriptor Statement

File descriptor statements are used to replace faulty file descriptor labels (HDR1 labels) on track 0. If the file is not to be copied, omit all operands on the file descriptor statement. In this case you must specify only:

```
␣DSnn
```

If the file is to be copied, at least all the required parameters must be supplied.

The format and entries of the file descriptor statement are:

```
␣DSnn␣file-id,nnnnn,boe,eoe,eod,B,S,P,E,m,dd,
crdate,exdate,V
```

The parameters are positional. If a parameter is left out, the comma must be specified. A blank must follow the last parameter. Parameters file-id through eod are required, parameters B through V are optional.

All required and optional information for one file can be contained in one card or card image (columns 1 - 72). Continuation is not allowed.

␣DS	Identifies the file descriptor statement.
nn␣	Indicates the file number. Must be a two-digit number corresponding to the number of the erroneous label found on the SYSLST output of the previous run. DSnn must be followed by at least one blank.
file-id	The file ID identifies the logical file. Can be from one to eight alphameric characters.
nnnnn	Indicates the block length. Must consist of one to five decimal digits. Contains the length of the data records on the file (1 - 128). Leading zeros may be omitted.

`boe`

Indicates the beginning of the extent (BOE). Must be five decimal digits in length and must be a valid diskette address.

cc = cylinder or track number
h = head number (always zero)
rr = record (sector) number

cchrr (lowest)	**cchrr (highest)**
01001	73026

The range covered by BOE and EOE must not overlap any other file.

`eoe`

Indicates the end of the extent (EOE). Must be five decimal digits in length and must be a valid diskette address.

cc = cylinder or track number
h = head number (always zero)
rr = record (sector) number

cchrr (lowest)	**cchrr (highest)**
01001	73026

The range covered by BOE and EOE must not overlap any other file.

EOE must not be lower than BOE.

`eod`

Indicates the end of data (EOD).

cc = cylinder or track number
h = head number (always zero)
rr = record (sector) number

cchrr (lowest)	**cchrr (highest)**
01002	74001

The EOD address must be at least one higher than the BOE address and not more than one higher than the EOE address.

`B`

Indicates that a file is to be bypassed.

`S`

Indicates that additional qualifications must be supplied in order to access the file.

If an S is specified for an *input* file, the operator must reply YES to access the file.

`P`

Indicates that a file may not be overwritten or deleted after expiration.

E		Indicates the exchange indicator, signifying fixed, unblocked, sequential records ≤ 128 bytes in length.
m		Indicates whether the whole file is contained on this volume, continued on another volume, or completed on this volume.
	m=C	The file is continued on another volume.
	m=L	The file did not start but ends on this volume. If the parameter is omitted, the file starts and ends on this volume.
dd		Indicates the order of a volume in a multivolume file. It must be two decimal digits in length. The first or only volume is 01. The default value is 01.
crdate		Indicates the creation date of the file. The format is yymmdd. The default creation date is the system date.
exdate		Indicates the expiration date of the file. The format is yymmdd. The default expiration date is the system date plus seven days.
V		Indicates that the data has been subjected to a verification procedure.

END Statement

The END statement must be supplied to signal the end of the utility control statements.

The format is:

```
// END
```

/* Statement

The /* statement must be supplied to close the files.

The format is:

```
/*
```

Control Statement Stream

An example of a control statement stream that is used to copy an error-free diskette to another diskette on a different I/O unit:

```
// JOB DISKETTE
// ASSGN SYS004,X'001'                    (Note 1)
// ASSGN SYS005,X'002'                    (Note 1)
// EXEC CRDR
// UCD TV,VOL=123456                      (Note 2)
// END
/*
/&
```

Note 1. Assign the input diskette to SYS004 and the output diskette to SYS005.

Note 2. The utility modifier statement is optional. In this case (an error-free input diskette), the VOL parameter is used to make sure that the correct input diskette has been mounted on SYS004.

The above job stream copied an input diskette with faulty labels to another diskette. If, however, the input diskette has one or more faulty labels, no copy operation will take place because you have not supplied descriptor statements and therefore the faulty labels cannot be replaced. The track 0 table is printed instead:

```
                                VOLUME NUMBER IS 222222
   (1)    (2)       (3)   (4)    (5) (6) (7) (8) (9) (10)  (11)  (12)    (13)   (14)  (15)
  DS01  VTL TC 1  00080  01001  03004                  C     01  730101  730101        03005
    02  VTL TC 3  00080  08001  08026                  L     02  730101  781224        08024
*** 03  VTL TC 4  00080  10004  10026
    04  VTL TC 2  00080  05001  06008                  C     01  730101  781224        06008
    05  VTL TC 5  00080  12001  12009                  C     01  730101  781224        12009
    06  VTL TC 6  00080  14001  15003                        01  730101  731224        15004
*** 07
    08
    09
    10
    11
    12
    13
    14
    15
    16
    17
    18
    19

(1) DSNN                    (6) BYPASS INDICATOR        (11) VOLUME SEQUENCE #
(2) DATA SET IDENTIFIER     (7) DATA SET SECURITY       (12) CREATION DATE YYMMDD
(3) BLOCK LENGTH            (8) WRITE PROTECTION        (13) EXPIR. DATE YYMMDD
(4) BEGIN OF EXTENT CCHRR   (9) EXCHANGE INDICATOR      (14) VERIFY INDICATOR
(5) END OF EXTENT CCHRR    (10) MULTI VOLUME INDICATOR  (15) END OF DATA CCHRR
```

Files 3 and 7 are marked defective by means of **. So you need a second run of the Copy and Restore Diskette program with a job stream containing file descriptor statements for files 3 and 7.

File 3: A read check occurred after the end-of-extent field was read. The lost information must be re-created. The information for four of the five required parameters is still present in the track 0 table:

```
ƀDS03 VTL TC 4,80,10004,10026
```

EOD is the only required parameter to be added. If this diskette is *not* the output of a previous copy and restore job, EOD points to the first record beyond EOE, which is the first record on the next track:

```
ƀDS03 VTL TC 4,80,10004,10026,11001
```

If the diskette is the output of a previous copy and restore job, it is possible that EOD is an address lower than 11001 and subsequently the new copy will contain too many records in file 3. The superfluous records may be deleted after the copy and restore job. All required parameters are present, and it is up to you to choose the optional parameters. In the example given below, the only optional parameter is the expiration date.

File 7: No information at all has been passed on to the track 0 table. Therefore, the read check occurred in the first bytes of the label.

You may now either

- Delete the file by coding ƀDS07.
  The output diskette will then contain six files.

or

- Re-create file 7. The problem is to determine the extent limits and the record length. In this case you can use the Copy and Restore Diskette program to make all records between file 6 and the end of the diskette accessible by coding:

```
ƀDS07 ANYNAME,128,15004,73026,74001
                  BOE   EOE   EOD
```

128 is the maximum record length.

Second Run: If you want to use an intermediate disk in the second run, and X'190' has been added to the system as a 2314 disk drive, the following job stream may be used:

```
// JOB DISKETTE
// ASSGN SYS004,X'001'                              (Note 1)
// ASSGN SYS005,X'190'                              (Note 2)
// DLBL UTEMP,,01/01,SD
// EXTENT SYS005,222222,1,1,200,62                  (Note 3)
// EXEC CRDR
// UCD TV
 DS03 VTL TC4,80,10004,10026,11001,,,,,,,,761231
 DS07
// END
/*
/&
```

Note 1. Diskette (input and output).

Note 2. If an intermediate disk file is used, assign the disk that is to contain the temporary file UTEMP (DLBL and EXTENT statements must be provided) to SYS005.

Note 3. The extent size depends on the disk device used. A full diskette requires disk storage as follows

Device	No. of tracks
IBM 2311	124
IBM 2314	62
IBM 3330	38
IBM 3340	68
IBM 3350	32

CHAPTER 9
COPY FILE AND MAINTAIN OBJECT MODULE (OBJMAINT)

Description

Purposes:

- To perform file-to-file copying of card image on card, diskette, sequential disk, and tape.
- To list card image files and programs in object format.
- To update object modules and phases.
- To deblock and/or update selected program temporary fixes (PTFs) from a blocked PTF file.

Functions

OBJMAINT is a multi-purpose utility program with three major functions: file-to-file copying of card image files, maintenance of programs in object format and processing PTFs.

File-to-File Utility

The first function is of particular importance (but not limited) to cardless system or diskette users. OBJMAINT can be used, for example, to create card image files on diskette; copy card image diskette files to another diskette, disk, or tape; and list card image files (including SYSIN) from the same devices.

File-to-file utility functions that can be performed with OBJMAINT include:

- Copying of card image files from and to the following:

  card
  tape
  diskette
  sequential disk.

- Blocking and deblocking of files on tape or sequential disk.

- Selection or exclusion of specific jobs while copying a SYSIN file.
- Listing of data
  - including normally unprintable characters
  - including JOB statements along with the total count of statements in each job of a SYSIN file
  - in 80/80 format (one line per logical input record).

Object Program Maintenance

The second function provides for updating object modules and phases. The object modules may be SYSPCH output from language translators or punched from the relocatable library (using RSERV). The phases may be punched from the core image library (using CSERV). The steps involved in updating cataloged object programs are discussed later in this chapter.

PTF Processing

IBM provides program temporary fixes (PTFs) to correct errors in IBM supplied programs. These PTFs normally consist of object modules (including user REP statements containing the code for updating the module) that will replace existing modules on the relocatable or core image library. The desired PTFs, which normally are included in a PTF file in blocked SYSIN format, must be deblocked prior to application. A PTF may be updated in the same manner as other programs in object format.

Functions provided for maintenance of both object modules and PTFs include:

- Selective copying of object modules or PTF jobs
- Deblocking PTFs from a blocked PTF file
- Selective updating of object modules, phases, or PTFs via user REP statements
- Removal of previously added user REP statements
- Expansion or truncation of a control section within an object module
- Combined object module expansion or truncation and user REP addition within the same job step
- Comprehensive listing options:
  - listing of normally unprintable characters, for example, EDECKs
  - suppression of listing of non-object module jobs
  - formatted listing of TXT or RLD statements
  - listing of a TXT statement on a single line
  - listing of JOB statements with total count of statements within each job
  - 80/80 listing of all statements.

Additionally, you may provide a program to further process input to OBJMAINT. This program phase must be defined in the EXIT statement and must adhere to the coding conventions as described in *User Exit Phase Considerations*.

If OC=YES was included in your supervisor, you may dynamically inquire as to the number of records already read from SYS004, written to SYS005 and printed on SYSLST. Also if a user exit is active, the number of records deleted and added will be shown. OBJMAINT will write these statistics on the console upon receiving an external interrupt via the INTERRUPT key in BG, or the MSG Fn operator command in the foreground partitions. In addition, if the input file is in SYSIN format, for example a PTF tape, the job name of the current job being processed will be shown.

Input/Output

OBJMAINT control statements are read from SYSIPT.

Input data files, object programs, and PTF jobs are assigned to SYS004 and may be on tape (see note), sequential disk, card or diskette. However, if OBJMAINT control statements and data are contained within the same input file, SYSIN must be assigned (see *ACTION Statement).*

Logical record length can be 80 or 81 bytes, and a physical block on tape or disk may contain up to 43 records. Both record and block length are calculated internally.

For disk or diskette input, use filename 'UIN' on the DLBL statement. Exception: For SYSIN files, filename 'IJSYSIN' must be used.

Data output is to SYS005 on tape (see note), sequential disk, card, or diskette and has a record length of 80 bytes. Tape and disk output BLOCKs may contain up to 43 records (see *Block Statement).*

For disk or diskette output, use filename 'UOUT' on the DLBL statement.

All OBJMAINT control statements and messages will be listed on SYSLST. The number of lines per page is determined by the 'SYSLINE' value in COMREG.

When performing a list function only, SYSLST is used.

On SYSLST, OBJMAINT will list

- The program name OBJMAINT and the date of run
- All control statements
- SYS004 and SYS005 file address, device, block size, and record size
- The total number of SYS004 statements by job
- OBJMAINT run statistics for statements on SYS004, SYS005, and SYSLST

- Requested ./ LIST statement data
- OBJMAINT messages.

Note: Tape Label processing is not performed. However, if present, the tape label will be displayed on the console for operator verification.

Job Control Statements

`// JOB ...`	Required.
`// ASSGN SYSLST,...`	Required. Output device for logging of control statements, program messages, and ./ LIST output.
`// ASSGN SYSIPT,...`	Required. Input device for OBJMAINT control statements.
`// ASSGN SYS004,...`	Required. Input device for OBJMAINT input data; may be card reader, diskette, tape, or DASD.
`// ASSGN SYS005,...`	Required. Output device for OBJMAINT output data; may be card punch, diskette, tape, or DASD.
`// DLBL filename,'file-id',...` `// EXTENT SYSnnn,...`	These statements must be provided if data is on diskette or DASD. The filename operand must be UIN for input files; UOUT for output files. If the OBJMAINT control statements are contained within the input data file on diskette or DASD, filename IJSYSIN must be used. In this case, SYSIPT, rather than SYS004, is assigned (see the ./ ACTION statement).
`// MTC`	Optional. This statement may be used for tape positioning.
`// EXEC OBJMAINT`	Required.
`,SIZE=AUTO`	Optional. OBJMAINT does not dynamically acquire storage, and therefore, has a static storage requirement.

/*	Required. For card input, an alternate EOF delimiter may be used (see ./ CARD statement).
/&	Required.

Utility Control Statements

OBJMAINT functions are defined using utility control statements. The control statements, read from SYSIPT, fit into two basic categories and are summarized in the table below.

Group 1 - those which provide descriptive information about the desired functions to be performed by group 2 statements during execution.

Group 2 - those which define the desired program function.

Multiple group 1 statements may be used, followed by only one group 2 statement. A group 2 statement is required for each OBJMAINT execution, except with the LIST and SELECT statements.

Control Statement	Description
Group 1:	
ACTION	Input characteristics; for example, job type, multiple volumes
BLOCK	Physical record sizes of tape or disk output
CARD	Specifies alternate EOF delimiter for card input
EXCLUDE	Indicates jobs to be excluded from processing
EXIT	User phase to process each input record read
LIST	Selective listing and formatting of input data
SELECT	Indicates jobs to be processed by OBJMAINT
UNREP *	Allows removal of previously added user REP statements

	Group 2:	
	COPY	File-to-file copy with deblocked output (BLOCK overrides)
	DEBLOCK	File-to-file copy with deblocked output always
	END	Delimits card input during a list only function
*	EXPAND	Expands or truncates a control section (CSECT)
*	EXPAND/REP	Single statement combines EXPAND and REP functions
*	REP	Allows updating object module via user REP statements

* These control statements are only used for the update of object programs and PTFs.

If OBJMAINT is invoked either from the console or without utility control statements, the ./ LIST function will be assumed.

All group 2 control statements, except END, cause the SYS005 file to be created.

Refer to the section *Control Statement Stream* for OBJMAINT setup.

All control statements must have this general format (starting with column 1):

./ operation parameters

DOS/VS comment statements may be used throughout.

ACTION Statement

The statement specifies input file characteristics. Its format is:

```
./ ACTION [JOBTYPE=DOS][,DATA=SYS004|SYSIPT]
          [,MULTIVOL=NO|YES][,FILES=nnn]
```

JOBTYPE=	Optional.
DOS	DOS/VS jobs are to be processed on SYS004 or SYSIPT. This parameter may be omitted.
DATA=	Identifies the source of input data.
SYS004	Input file is on the device assigned to SYS004. Files in SYSIN format may be processed on SYS004 provided that OBJMAINT control statements are on another device assigned to SYSIPT. Card input data inline with OBJMAINT control cards should be read from SYS004, which must be assigned to the same card device as SYSIPT.
SYSIPT	Input file is on the device assigned to SYSIPT. If the OBJMAINT control statements are embedded in the input data on DASD, tape, or diskette, SYSIPT must be specified. To eliminate end-of-file conditions caused by /* and / &, use ./EOF and ./EOJ respectively (no embedded blanks). As these data statements are read from SYSIPT they will be changed to /* and / &. If a user exit is active, control is given to the user routine prior to alteration of either the ./EOF or ./EOJ. Refer to the ./ CARD statement for the specification of an alternate EOF delimiter. If the parameter is omitted, SYS004 is assumed.
MULTIVOL=	Multiple volumes tape input parameter.
YES	Tape input file consists of two or more volumes.
NO	Tape input is on a single volume. NO is the default.
FILES=	Multiple files tape input parameter.
nnn	Specifies the number of files on the input tape. This entry must be a decimal number of one to three digits not exceeding 255.

This operand allows the concatenation of multiple tape files into one file by specifying the total number of files on all tapes involved. The concatenated files are processed as one input file.

All tape files must have a record length of 81 or 80 bytes; however, block sizes may vary.

If the parameter is omitted, 1 is assumed.

BLOCK Statement

This statement specifies the SYS005 output block size for tapes and DASD. Its format is:

```
./ BLOCK [BLKSIZE=nnnn]
```

BLKSIZE= Block size parameter.

nnnn The block size value may be submitted as a decimal number in multiples of 80 bytes up to 3440, except for a 2321. If the specified value is not a multiple of 80, it will be adjusted downward to the next multiple of 80.

Device type 2321 has a maximum block size of 2000.

If the parameter is omitted, a BLKSIZE of 3440 is assumed.

If the BLOCK statement is omitted, output is unblocked.

CARD Statement

This statement specifies an alternate delimiter for card input. Its format is:

```
./ CARD DELIM=xx
```

DELIM= Alternate spellings of the delimiter parameter include DLM and DELIMITER.

xx Specifies two characters other than /* that will indicate end-of-file on SYSIPT. Card input data should be placed immediately following the group 2 type control statement used. If the optional delimiter is omitted, /* is assumed to indicate EOF.

This statement must be included when SYSIN files are processed on SYS004 (refer to *ACTION Statement* above).

COPY Statement

The statement identifies the file-to-file copy function. Its format is:

```
./ COPY
```

It may be used with group 1 functions to produce a blocked SYS005 file (with a BLOCK statement). Without the BLOCK statement, the input file will be copied as is, if it is unblocked, and it will be de-blocked if blocked.

DEBLOCK Statement

The statement identifies the input file deblocking function. Its format is:

```
./ DEBLOCK
```

The copied SYS005 output file will contain 80 byte blocks. Any ./ BLOCK control statement will be ignored.

END Statement

The statement identifies an end-of-card input function. Its format is:

```
./ END
```

This control statement is used to mark the end of SYS004 card input during a LIST function, when no other group 2 control statement is present.

EXCLUDE Statement

The statement identifies the exclusion function. Its format is:

```
./ EXCLUDE jobname,jobname(n),...
```

`jobname` Represents the name of a job to be excluded or deleted from the input job stream. The job names need not be specified in the same sequence as the jobs in the input file.

A maximum of 120 jobs may be processed in one pass (via multiple EXCLUDE statements).

Since no statement continuation is allowed, all 80 columns of the EXCLUDE statement may be used.

(n) Specifies the nth job of multiple jobs with the same job name.

If used, the EXCLUDE statement may not be used in the same job step as the SELECT and must precede any group 2 statement.

EXIT Statement

The statement identifies the user exit interface function. Its format is:

```
./ EXIT NAME=phasename
```

NAME= User phase name parameter.

phasename Specifies the name of the user routine to be given control during execution of OBJMAINT. This phase must be cataloged and must not exceed 6K bytes in length.

If used, the EXIT statement must precede any COPY, DEBLOCK, EXPAND, or REP control statements.

The named phase will be loaded into a predefined area of OBJMAINT for execution, and will receive control after each SYS004 input record is read. At this point, records may be modified or new records added to SYS005. SYSLST may also be used. At end-of-file additional records may be added.

Refer to section *User Exit Phase Considerations* for coding conventions.

EXPAND Statement

The statement identifies the module expansion or truncation function. Its format is:

```
./ EXPAND [NM=phasename],SD={csect|(esid)},
          LENGTH=cslength[,INITIMG=nn]
```

NM=	Phase name parameter. This should be used if duplicate CSECT names appear within the input stream.
phasename	Specifies the name of the phase containing the CSECT to be expanded or truncated (see *Updating Cataloged Object Modules*).
SD=	CSECT name parameter.
csect	Specifies the name of the control section to be expanded or truncated.
(esid)	Specifies the external symbol identification, which must be a hexadecimal value from one to three digits within parentheses.
LENGTH=	Module length parameter.
cslength	Specifies total length after the expansion or truncation of the named CSECT. In case of truncation, no TXT statements are removed; an error during a subsequent link-edit may result. Length value may be from one to six decimal or hexadecimal digits. Hexadecimal values must be in the form x'nnn', for example, x'800'.
INITIMG=	Expansion data parameter.
nn	Specifies the fill character to be used in the TXT statements to be added during expansion. This field must be expressed as hexadecimal data in the form x'nn' or nn. If this operand is omitted, OBJMAINT will expand the length without adding TXT statements. Truncation of a module expanded with an INITIMG value may lead to an unsuccessful link-edit because TXT statements are not removed during module truncation.

EXPAND/REP Statement

The statement identifies the function that combines object module expansion or truncation with user REP statement additions. Its format is:

```
./ EXPAND/REP  [NM=phasename],
               {SD= csect|(esid)|(PC)},
                LENGTH=cslength,[INITIMG=nn]
```

Refer to the REP and EXPAND control statement descriptions for use of each operand.

LIST Statement

The statement identifies the list function. Its format is:

```
./ LIST [PARM=option,option,...]
```

PARM=	Keyword; must be present if parameter options are used.
BINARY	Lists normally unprintable statements (i.e. EDECKs) in hexadecimal format.
TXT|TEXT	Prints TXT statements in dump format showing both character and hexadecimal representation.
LIMIT	Suppresses listing of non-object module jobs.
RLD	Formats all RLD statements.
JOB	Lists only JOB statements and a total count of statements within the job.
SHORTTXT	Formats TXT statements on a single line.
80/80	Lists all statements without formatting.

Omission of the LIST statement or use of the LIST statement without parameters will exclude listing of TXT statements, RLD statements, and statements containing excessive binary data, for example, EDECKs.

If OBJMAINT is invoked from the console, the LIST function will be assumed.

To aid in identifying jobs, OBJMAINT writes a job separator line on SYSLST for each JOB statement encountered by the program. All ESD and END statements are formatted. The first statement of the SYS004 file will be listed regardless of the LIST parameters.

The LIMIT parameter indicates to OBJMAINT that if a control statement is not encountered within ten statements, listing should be suppressed until another control statement is found. A control statement is defined by:

1. a slash (/) in column 1
2. x'02' (12-9-2 punch) or a plus symbol (+) in column 1
3. ./ in the first two columns
4. CATALS, CATALP, CATALR, or UPDATE Librarian statements.

REP Statement

The statement identifies the function that allows the addition of user REP statements to an object module. Its format is:

```
./ REP [NM=phasename][,SD={csect|(esid)|(PC)}]
```

NM=	Phase name parameter.
phasename	Specifies the name of the phase containing the target CSECT (see *Updating Cataloged Object Modules).*
SD=	CSECT name parameter.
csect	Specifies the name of the control section to be modified.
(esid)	Specifies the external symbol identification, which must be a hexadecimal value from one to three digits within parentheses.
(PC)	Specifies that the first section of private code encountered should be processed.
	If both parameters are omitted, the first object module encountered will be acted upon.

User REP statements should be inserted immediately following the ./ REP statement. The ./ REP statement may be used without user REP statements to bypass CSECTS having duplicate names in the input stream. As OBJMAINT processing is sequential, control statements must be in the same sequence as the SYS004 input stream.

User REP statements must be in a form acceptable to the linkage editor, with one exception: that is, in order to handle systems with 96-column card devices, either a plus symbol (+) or the standard 12-9-2 punch (x'02') in card column 1 is allowed.

SELECT Statement

The statement identifies the job selection function. Its format is:

```
./ SELECT jobname,jobname(n),...
```

jobname	Represents the names of the jobs desired for processing. Operands need not appear in the same sequence as the jobs on the input file.
	A maximum of 120 jobs may be processed in one pass (via multiple SELECT statements).

Since no statement continuation is allowed, all 80 columns of the SELECT statement may be used.

(n) — Specifies the nth job of multiple jobs with the same job name.

A SELECT statement must precede any group 2 statement. When used alone, the SELECT statement produces a listing of the selected jobs.

UNREP Statement

The statement identifies the function required for removing existing REP statements from an object module. Its format is:

```
./ UNREP  [NM=phasename][,SD=csect]
          [,DATA={'xxxxxx nnn'|ALL}]
```

NM= — Phase name parameter.

phasename — Specifies name of affected phase (see *Updating Cataloged Object Modules).*

SD= — CSECT name parameter.

csect — Specifies the name of the control section within the object module to be updated. Any CSECT within a phase may be named, as OBJMAINT will scan the entire module.

DATA= — Identifies REP search argument parameter.

'xxxxxxbnnnn' — Indicates 10 bytes of hexadecimal characters which should match columns 7 through 16 of the user REP statement to be eliminated.

ALL — All user REP statements will be removed from the named control section.

If the DATA parameter is omitted, ALL is assumed. Up to 50 unique UNREP statements are permitted in one job step.

If all operands of the UNREP statement are omitted, all user REP statements will be removed from the first object module containing one or more user REP statements.

If it is only desired to remove existing user REP statements from an object module, the group 2 COPY function should also be used to get the updated output on SYS005.

Updating Cataloged Object Modules

Updating cataloged object modules requires three steps:

1. Use RSERV or CSERV to punch the desired object modules or phases to SYSPCH. SYSPCH may be assigned to tape, disk, or diskette.
2. After assigning the SYSPCH file to SYS004, execute OBJMAINT with the necessary control statements on SYSIPT. Output will be on SYS005 in SYSIPT format.
3. After assigning the SYS005 file to SYSIPT, execute MAINT or LNKEDT to recatalog the updated object modules, or phases.

When OBJMAINT control statements require a phase name, either the name on the CATALR statement or on the PHASE statement may be used for relocatable object modules.

When core image library phases are punched out, CSERV generates an ESD statement with an ESID of 001 and a CSECT name equal to the phase name.

User Exit Phase Considerations

You may desire to process records from the SYS004 file in a manner not supported by OBJMAINT. To do so, a program up to 6K in size may be written and cataloged into the core image library. When named on the EXIT statement, this phase will be loaded by OBJMAINT. See also the discussion of the ./ EXIT statement.

Your exit phase will receive control for each record read. You may then modify the current record as well as add or delete statements. Also, if the SELECT function is active and the current job was not selected for inclusion on the SYS005 file, you may, for example, choose to bypass statement processing.

Following are the coding conventions for a user exit.

When your program is entered, the following registers will contain the values as indicated:

Registers

R13	Pointer to an 18 fullword save area
R14	Linkage register; normal return to OBJMAINT
R15	Entry to your exit routine
R1	Pointer to a 44-byte parameter list (see below)
R2	Pointer to the current data record or zero if end-of-file

The 44-byte parameter list pointed to by Reg. 1 contains addresses as follows:

```
Displacement
(in decimal)
   0       DC A(INPUT1)        I/O area for SYS004 data record (80 bytes).
   4       DC A(RETCODE)       Return code byte.
   8       DC A(PRTLINE)       I/O area for SYSLST (133 bytes).
  12       DC A(PUTSYSL)       SYSLST write routine.
  16       DC A(OUTPUT1)       I/O area for SYS005 file (80 bytes).
  20       DC A(PUTSYS5)       SYS005 write routine.
  24       DC A(SELSWT)        SELECT switches (2 bytes).
                               SELSWT+0: 00 = SELECT is not active.
                                         FF = SELECT is active.
                               SEKSWT+1: 00 = current job is not selected.
                                         FF = current job is selected.
  28       DC A(CURRJOB)       Pointer to current job name (8 bytes).
  32       DC A(TYPESTMT)      Type statement indicator (1 byte).
                               X'80' if current statement is a JOB statement.
  36       DC A(*)             Reserved.
  40       DC A(*)             Reserved.
```

When your exit is entered, register 15 points to your program's entry point. Register 13 points to an 18 fullword save area. You should save the registers in this save area on entry to your program. Then establish an 18 fullword save area of your own if you want to use either of the routines PUTSYS5 or PUTSYSL. These routines may be used to add records to the SYS005 file and to list these added or modified records.

When you expect to do any I/O within your routine or use the PUTSYSL or PUTSYS5 write routines and expect to return to OBJMAINT you must save register 14. The return register for PUTSYSL and PUTSYS5 is register 14. Register 15 must be loaded with the address of the routine to be executed: PUTSYS5 for OBJMAINT's SYS005 write routine, PUTSYSL for OBJMAINT's SYSLST write routine.

The first byte of the SYSLST I/O area is the CCW command code used for the SYSLST PUT operation. You may use any of the following:

x'09'	space 1 line after writing
x'11'	space 2 lines after writing
x'19'	space 3 lines after writing
x'0B'	space 1 line immediately
x'13'	space 2 lines immediately
x'1B'	space 3 lines immediately
x'8B'	skip to channel 1

After return from PUTSYSL and PUTSYS5, the associated I/O areas are cleared to blanks so there is no need for you to do this.

The byte at TYPESTMT can be tested to determine if the current statement is a JOB statement. This is indicated by bit 0 of that byte. If the current statement is a JOB statement, the job name can be found at CURRJOB. This is an 8 byte field, left justified.

Before returning to OBJMAINT, indicate the disposition of the statement being processed. The byte at RETCODE determines whether the current statement is to be kept or deleted. If the return code is 00 the statement is kept and copied to SYS005. If the return code is 04 the statement is delet-

ed from the SYS005 file. RETCODE is initialized to 00 prior to each entry to your exit so you need to alter it only when you wish to delete a statement.

Example 3 in the section *Control Statement Stream* is a user exit example.

Control Statement Stream

Example 1. Create SYSIN file on diskette

This example copies SYSIN card input (// JOB TEST) to a diskette. In order to copy the /* and / & statements, an alternate delimiter of $$ is used to indicate EOF for SYS004 input. The second job step simply lists the created diskette file.

```
// JOB CDTDSKT
// DLBL UOUT,'PTFWORK3',75/300,DU
// EXTENT SYS005,OBJMNT
// ASSGN SYS004,SYSIPT
// ASSGN SYS005,3540
// EXEC PGM=OBJMAINT
./ CARD  DLM=$$
./ COPY          CARD TO DISKETTE
// JOB TEST
*  TEST 1.....TEST 1.....TEST 1 )
*  TEST 2.....TEST 2.....TEST 2 }  input for the
*  TEST 3.....TEST 3.....TEST 3 )  COPY statement
/*
/&
$$
/*
// PAUSE   START DISKETTE FOR SECOND TIME
// RESET SYS005
// DLBL UIN,'PTFWORK3',,DU
// ASSGN SYS004,3540
// EXEC OBJMAINT
./ LIST
/*
/&
```

Example 2. Use of ./ END for a LIST only function

This example shows the required use of the ./ END statement when doing a LIST only function from the card reader.

```
// JOB CDLIST
// ASSGN SYS004,SYSIPT
// EXEC PGM=OBJMAINT
./ LIST
./ CARD DLM=$$
./ END
// JOB TEST
*  TEST 1.....TEST 1.....TEST 1
*  TEST 2.....TEST 2.....TEST 2
*  TEST 3.....TEST 3.....TEST 3
/*
/&
$$
/*
/&
```

Example 3. User exit phase

This example shows how a user exit phase may be used in conjunction with OBJMAINT to process SYS004 input. The phase changes // PAUSE statements to comment statements in PTFs selected for system application. The job USEREXIT is set up to assemble and catalog the phase NOPAUSE just prior to being invoked by OBJMAINT (in the third job step). The SYS005 output tape will be in deblocked SYSIN format. In this example ./ DEBLOCK could be replaced by a ./ COPY statement with the same effect.

```
// JOB USEREXIT
// OPTION CATAL,NODECK
  PHASE NOPAUSE,+0
// EXEC PGM=ASSEMBLY,SIZE=44K
OBJEXIT CSECT
R0        EQU   0
R1        EQU   1
R2        EQU   2
R3        EQU   3
R4        EQU   4
R10       EQU   10
R11       EQU   11
R12       EQU   12
R13       EQU   13
R14       EQU   14
R15       EQU   15
          SPACE 1
PARMLIST  DSECT                    ADCON LIST AS IN OBJMAINT
INPUT1    DS    F
RETCODE   DS    F
PRTLINE   DS    F
PUTSYSL   DS    F
OUTPUT1   DS    F
PUTSYS5   DS    F
SELSWT    DS    F
CURRJOB   DS    F
TYPESTMT  DS    F
          SPACE 1
OBJEXIT   CSECT
          STM   R14,R12,12(R13)    SAVE REGS
          LR    R10,R15            LOAD BASE REG
          USING OBJEXIT,R10
          USING PARMLIST,R12       SET ADDRESSABILITY TO ADCONS
          LR    R3,R13             COPY OBJMAINT SAVE REG
          LA    R13,SAVE           POINT TO MY SAVE AREA
          ST    R13,8(0,R3)        STORE FORWARD POINTER
          ST    R3,4(0,R13)        STORE BACKWARD POINTER
          LR    R12,R1             COPY PARAMETER LIST POINTER
          LTR   R2,R2              TEST FOR END-OF-FILE EXIT
          BZ    EXITEOF            HERE, ADD ASSGN SYSIN STMT
          L     R1,INPUT1          LOAD POINTER TO DATA RECORD
          L     R3,SELSWT          LOAD POINTER TO SELECT SWITCHES
          CLC   0(2,R3),=X'FF00'   TEST STATUS OF SELECT SWITCHES
* IF SELECT FUNCTION IS USED, AND CURRENT JOB IS NOT SELECTED,
* THEN EXIT BACK TO OBJMAINT.
          BE    EXITN
          CLC   0(2,R1),=C'//'     IS THIS A JCL STMT
          BNE   EXITN
          LA    R3,74              LOAD COUNTER
          LA    R4,2(0,R1)         POINT PAST //
EXIT1     CLI   0(R4),C'Ƀ'         SCAN FOR OPERAND
          BNE   EXIT2
```

```
          LA    R4,1(0,R4)          BUMP TO NEXT BYTE
          BCT   R3,EXIT1            CONTINUE SCAN
          B     EXITN               TAKE NORMAL RETURN
EXIT2     CLC   0(6,R4),=C'PAUSE '  IS THIS PAUSE STMT
          BE    EXIT4               IF SO, TAKE ACTION
EXITN     L     R13,4(0,R13)        LOAD BACKWARD SAVE AREA POINTER
          LM    R14,R12,12(R13)     RESTORE REGS
          BR    R14                 TAKE NORMAL RETURN
          SPACE 1
EXITEOF   EQU *
          L     R4,OUTPUT1          POINT TO SYS005 I/O AREA
          MVC   0(L'REASSGN,R4),REASSGN MOVE ASSGN STMT TO I/O
          L     R15,PUTSYS5         POINT TO SYS005 WRITE ROUTINE
          BALR  R14,R15             GO WRITE LAST STMT
          L     R4,PRTLINE          POINT TO PRINT I/O AREA
          MVI   0(R4),11            CAUSE SPACE IMMEDIATE
          L     R15,PUTSYSL         POINT TO PRINT PUT ROUTINE
          BALR  R14,R15             LINK TO PUTLST ROUTINE
          MVI   0(R4),17            CAUSE WRITE AND SPACE 2
          MVC   1(L'REASSGN,R4),REASSGN SHOW RE-ASSGN STMT ON LST
          MVC   81(L'MSG2,R4),MSG2  IND AN ADDED STMT
          L     R15,PUTSYSL         POINT TO PRINT PUT ROUTINE
          BALR  R14,R15             GO WRITE MSG
          B     EXITN               EXIT BACK TO CLOSE ROUTINES
          SPACE 1
EXIT4     EQU *
* PAUSE STATEMENTS WILL BE ALTERED TO COMMENT STATEMENTS
* PAUSE STATEMENTS OFTEN CONTAIN VALUABLE COMMENT INFORMATION
* AND THEREFORE WILL NOT BE DELETED.
          MVC   0(2,R1),=C'* '      CHANGE // TO COMMENT
          B     EXITN
          SPACE 2
          DS    0F
SAVE      DC    18F'0'              REG SAVE AREA
MSG2      DC    C'*** STMT ADDED BY USER EXIT ***'
REASSGN   DC    C'ASSGN SYSIN,X''00C'''
          LTORG
          END
/*
// EXEC PGM=LNKEDT,SIZE=64K
/*
// ASSGN SYS004,X'283'             INPUT PTF TAPE
// ASSGN SYS005,X'281'             OUTPUT TAPE FOR SELECTED PTFS
// MTC REW,SYS004
// MTC REW,SYS005
// MTC WTM,SYS005
// MTC REW,SYS005
// EXEC PGM=OBJMAINT,SIZE=AUTO
./ SELECT NO2582#3,NO4574#3
./ EXIT  NAME=NOPAUSE
./ DEBLOCK FOR SYSIN
/*
// MTC REW,SYS005
// RESET SYS005
// MTC REW,SYS004
// ASSGN SYS004,X'281'
// EXEC OBJMAINT
./ LIST PARM=JOB
/*
// MTC REW,SYS004
/&
```

Example 4. Copy Selected PTFs

This example excludes unwanted PTF jobs from the distribution tape and blocks the output PTF jobs to tape.

```
// JOB PTFBLOCK
// ASSGN SYS004,X'283'
// MTC REW,SYS004
// EXEC PGM=OBJMAINT
./ LIST PARM=JOB
/*
// MTC REW,SYS004
// ASSGN SYS005,X'281'     OUTPUT TAPE
// MTC REW,SYS005
// MTC WTM,SYS005
// MTC REW,SYS005
// EXEC PGM=OBJMAINT
./ BLOCK  BLKSIZE=3440
./ EXCLUDE  NO2592#3,NO3700#A,NO9706#A
./ COPY    TAPE TO TAPE
/*
// MTC REW,SYS004
// MTC REW,SYS005
// RESET SYS005
// ASSGN SYS004,X'281'
// EXEC PGM=OBJMAINT
./ LIST PARM=JOB
/*
// MTC REW,SYS004
/&
```

Example 5. Update an object module

This example writes the object module OBJTEST from the relocatable library to disk; removes previously added user REP statements; expands the size of the module, while adding a new user REP statement; and finally, in the last job step, lists the updated object module (on diskette). The INITMG parameter causes TXT statements containing hexadecimal AA to be added to the expanded module.

```
// JOB XPANDMOD
// DLBL IJSYSPH,'PTF.WORK.FILE.1',0,SD
// EXTENT SYSPCH,111111,,,7161,11
// ASSGN SYSPCH,X'160',PERM,VOL=111111,SHR
// EXEC RSERV
  PUNCH OBJTEST
/*
CLOSE SYSPCH,X'04D'
// DLBL UIN,'PTF.WORK.FILE.1',0,SD
// EXTENT SYS004,111111
// DLBL UOUT,'PTFWORK3',75/300,DU
// EXTENT SYS005,OBJMNT
// ASSGN SYS004,3340,TEMP,VOL=111111,SHR
// ASSGN SYS005,3540
// EXEC PGM=OBJMAINT
./ UNREP  NM=OBJTEST,DATA=ALL
./ EXPAND/REP NM=OBJTEST,SD=(001),LENGTH=X'100',INITIMG=AA
+REP  000010 001EEEE,EEEE,EEEE,EEEE,EEEE,EEEE,EEEE,EEEE
/*
// PAUSE  START DISKETTE FOR SECOND TIME, NOW AS INPUT
// DLBL UIN,'PTFWORK3',,DU
// ASSGN SYS004,X'04F'
// EXEC OBJMAINT
./ LIST  PARM=TXT
/*
/&
```

CHAPTER 10
DEBLOCK

Description

Purposes:

- To block an 80/81-byte record file to a 3440-byte record file.
- To deblock a blocked 3440-byte (IBM distribution) file in order to create an 80-byte SYSIN file.
- To copy files.
- To print (list) job control statements and comments from a blocked input file.
- To select records (or a group of records) from a blocked 3440-byte file in order to create an 80-byte SYSIN file.

This program does not support the IBM 1442 Card Read Punch or the IBM 2520 Card Read Punch.

Functions

In general, all functions of the Deblock utility (plus additional ones) are also available with the Copy File and Maintain Object Module (OBJMAINT) utility.

Block: To block an 80 or 81-byte record file to a 3440-byte record file.

Deblock: To deblock the blocked 3440-byte file in order to create an 80-byte SYSIN file.

Copy: The card-to-card copy function includes 80-column to 96-column conversion for the IBM 5425 Multi-Function Card Unit.

List: To determine the contents of a file with blocked 3440-byte records. Note, however, that only the first 73 characters of job control statements (starting with two slashes) and comment statements are listed. Furthermore, listing between a // PAUSE statement and a / & statement is suppressed.

Select: To deblock selected PTFs from a blocked PTF file. The function can also be used for any other 3440-byte blocked sequential file.

Input/Output

The devices used for input and output are defined by assigning the input device to SYS004 and the output device to SYS005. For the list function the output device is SYSLST.

When a disk is assigned, DLBL and EXTENT statements are required. The entries used for the DLBL statement are:

```
// DLBL UIN,'file-id'      (for input)
// DLBL UOUT,'file-id'     (for output)
```

Tape labels and the UPSI byte are not supported, except for deblocked output tapes.

To create a deblocked tape, a TLBL statement or an UPSI statement is required. If label checking and creation is desired, a TLBL statement is required. If label checking and creation is to be suppressed, the UPSI statement is required instead. The entries that are used for the TLBL and UPSI statements are:

```
// TLBL UOUT,'file-id'
// UPSI 00100000
```

NO REWIND is always assumed for input/output tapes. Be sure that the tapes are correctly positioned.

Figure 10.1 shows the input/output devices for the block, deblock, select, and list functions, and Figure 10.2 those for the copy function.

		Block		Deblock/Select		List
		Input	Output	Input	Output	Input
	Record Format	80/81 bytes Unblocked	80 bytes 3440 blocked	80 bytes 3440 blocked	80 bytes Unblocked	80 bytes 3440 blocked
Devices	Card	yes	no	no	yes	no
	Tape	yes	yes	yes	yes	yes
	Disk	no	yes	yes	yes	yes

Figure 10.1 Input/output devices for block, deblock, select, and list functions

		Copy			
		Input	Output	Input	Output
	Record Format	80/81 bytes Unblocked*	80 bytes Unblocked*	80 bytes 3440 blocked	80 bytes 3440 blocked
Devices	Card	yes	yes	no	no
	Tape	yes	yes	yes	yes
	Disk	no	yes	yes	yes

* When a card device is assigned to SYS004 or SYS005, the program supports unblocked files; otherwise, blocked files are assumed. Be sure you mount a tape or disk with records of the required length.

Figure 10.2 Input/output devices for the copy function

Utility Modifier Statement

Contains information to run the program and is required for all functions.

The parameters are positional. The format and entries for the block, deblock, copy, and list functions are:

```
// UDS ffff
```

If this statement is omitted, a default value of

```
// U DBL
```

is assumed.

For the select function, the format and parameters are:

```
// UDS SEL,n,'x...x',m1,m2
```

`// U`	Identifies the utility modifier statement.
`DS`	Indicates the Deblock program. Can be omitted.
`ffff`	Indicates the function specification. Can be omitted. The default is DBLb.
`ffff=BLKb`	Block function.
`ffff=COPb`	Copy function.
`ffff=DBLb`	Deblock function.
`ffff=LSTb`	List function.
`ffff=SEL,`	Select function.

The following parameters define the select identifier and are only required for the select function. For an explanation of the select identifier see *Selector Statement.*

`n`	Indicates the starting position (column number) of the fixed part of the select identifier. A decimal number, from 1 to 80, is required.
`'x...x'`	The fixed part of the select identifier. All characters are allowed.
`m1`	Indicates the starting position (column number) of the variable part of the select identifier. A decimal number, from 1 to 80, is required.
`m2`	Indicates the end position (column num-

ber) of the variable part of the select identifier. A decimal number, from 1 to 80, is required.

Note: Apostrophes must not be used in any comment in the utility modifier statement for the select function.

END Statement

Must be supplied. The format is:

```
// END
```

Selector Statement

Selector statements are only required if you request the SEL function of the Deblock program in the utility modifier statement. The selector statements follow the END statement (see example 6 in *Control Statement Stream).*

To identify the records to be selected, any distinctive part of the records (for example column 1 to column 14) can serve as an identifier, and the data contained in these columns constitutes the select identifier. In order to simplify the selection of a group of records the select identifier is divided into a fixed part and a variable part. The fixed part is common to all the records to be selected from the file; it is specified once in the utility modifier statement ('x...x'). The variable part of the select identifier is different for each record to be selected. The record columns to be compared with the variable part of the select identifier are specified in the utility modifier statement (m1 and m2); however, the characters that constitute the search arguments have to be specified in the selector statements, one for each record to be selected.

The format of the selector statement is shown in example 6 in *Control Statement Stream.* Any number of characters, from 1 to 80, is allowed. If the fixed and variable parts overlap, the variable part overwrites the fixed part.

The selected stream of records starts with the record that has the required characters in the positions specified by the select identifier. The select operation is ended by a / & statement from SYS004.

The input file identifiers are searched in the order in which the selector statements are read. Therefore, the sequence of the selector statements should be identical to that of the input file.

Control Statement Stream

Six examples of control statement streams to run the program are given.

1. Blocking from card to tape.

```
// JOB BLOCK CARD TO TAPE
// ASSGN SYS004,X'00C'
// ASSGN SYS005,X'282'
// EXEC DSTRB
// UDS BLK
// END
// .
   .
  data cards
   .
   .
/*EOD      (no embedded blanks)
/&
```

It is required that the last two statements be /*EOD and / & , in that order.

2. Deblocking from tape to disk.

```
// JOB CREATE SYSINFILE
// ASSGN SYS004,X'283'
// ASSGN SYS005,X'132'
// DLBL UOUT,'SYSINFLE',99/365
// EXTENT SYS005,,,,20,780
// EXEC DSTRB
// UDS DBL
// END
/&
```

3. Deblocking from tape to tape.

```
// JOB DEBLOCK TAPE TO TAPE
// ASSGN SYS004,X'180'
// ASSGN SYS005,X'181'
// TLBL UOUT,'DEBLOCKT'
// EXEC DSTRB
// UDS DBL
// END
/&
```

4. Copying card to tape.

```
// JOB COPY CARD TO TAPE
// ASSGN SYS004,X'00C'
// ASSGN SYS005,X'181'
// TLBL UOUT,'COPYCRDT'
// EXEC DSTRB
// UDS COP
// END
// .
   .
  data cards
   .
   .
/*EOD      (no embedded blanks)
/&
```

5. Listing a blocked tape.

```
// JOB LIST BLOCKED TAPE
// ASSGN SYS004,X'181'
// EXEC DSTRB
// UDS LST
// END
/&
```

6. Selecting from tape.

```
// JOB SELECT FROM TAPE
// ASSGN SYS004,X'180'
// ASSGN SYS005,X'00C'
// EXEC DSTRB
// UDS SEL,1,'INPUTREC',9,14
// END
123456
437298
/*
/&
```

Record Limits

During blocking, messages will be generated to inform you that the blocked file, when deblocked, will fit on 90% of a 2400-foot 7 or 9-track or on a 2311 disk. Processing continues. These limits (90%) are:

31,000 records (7-track tape)
40,000 records (9-track 1600 bpi tape)
45,000 records (2311 disk).

Be sure you mount an output tape or disk that can contain the whole file.

CHAPTER 11
FAST COPY DISK VOLUME

Description

Purpose:

- To copy the entire contents of a disk device onto another disk device of the same type in a short time. The pack to be copied may contain any combination of DOS/VS files and system components.

  The contents of this disk may be copied directly to another disk device, or it may be written on magnetic tape, to be restored at a later time.

This program does not support the IBM 2311 Disk Storage Drive or the IBM 2314 Direct Access Storage Facility.

The Fast Copy program copies complete volumes and cannot be used to copy only a part of a disk volume, such as an individual data file.

The program can transfer the volume contents from disk to disk, or from disk to tape and subsequently from tape to disk.

Your choice of one of these three functions must be indicated by an entry in the utility modifier statement. The target volume must be of the same type as the source volume and must have the same or a greater storage capacity.

There are three versions of the Fast Copy program:

- an 'integrated' version which runs in a partition under the DOS/VS supervisor, like the other system utilities
- two stand-alone versions: one being card-resident and the other diskette-resident. Such a version is fully contained in a card deck or on a diskette, respectively, and can be loaded and executed without the DOS/VS SYSRES pack. Using the diskette-resident version requires DOC support.

Magnetic tapes created by any of the program versions, in any specific DOS/VS release, may be restored using any version of the program. For example, tapes written by the disk-to-tape function in the integrated version, may be restored to disk by the stand-alone tape-to-disk function.

The VTOC (volume table of contents) of a disk output volume will be scanned for unexpired files. For any file that is found, the file ID and expiration date are issued on SYSLOG. The operator may then either

cancel the job or ignore the unexpired files and proceed with the copy operation.

The Fast Copy program provides the following options for disk volumes:

- The volume serial number written on an output disk volume may be
  - copied from the original input disk
  - rewritten as specified by you in the utility modifier statement.

Additionally, the volume serial number of the output disk volume, prior to being written on, may be checked against the number specified in the utility modifier statement. This option is provided to ensure that the correct disk pack has been mounted. If not, the operator may exchange disk packs, correct the utility modifier statement, or cancel the job.

These options apply to the disk-to-disk and tape-to-disk functions.

The Fast Copy program provides the following options for tape volumes:

- Two tape label options
  - no labels
  - standard labels (one volume label, one file header label, and one file trailer label per volume).

  The option specified when the tape is created by the disk-to-tape function must also be specified when the tape is restored to disk by the tape-to-disk function.
- Alternate tape drives are supported for both program functions using magnetic tape.

Tape Processing

The only purpose of the tape file created by the disk-to-tape function is to provide input to the subsequent tape-to-disk restore function. You should not use the tape for any other purpose, because the information is uniquely formatted, intermixing control information with data.

The label processing option is specified by the bit settings of the UPSI statement.

Output tape label processing

Bit 2 of the UPSI statement determines output label processing. Therefore, for the disk-to-tape function, an UPSI statement may be submitted in one of two formats:

```
// UPSI 00000000
```

Indicates standard output label checking and creation. In this case

- The UPSI statement is optional, because standard label processing is the default.
- A TLBL statement is required. The first operand must be UOUT.

```
// UPSI 00100000
```

Indicates that no label checking of the output tape is performed and that no file header or trailer labels are created. In this case

- The UPSI statement is required.
- A TLBL statement is not required. If submitted, its contents are not used by the program.

Input tape label processing

Bit 0 of the UPSI statement indicates your option for label checking of input tape volumes. For the tape-to-disk function, an UPSI statement may be submitted in one of two formats:

```
// UPSI 00000000
```

Indicates that input tapes contain standard labels which must be checked. In this case

- The UPSI statement is optional, because standard label processing is the default.
- A TLBL statement is required. The first operand must be UIN.

```
// UPSI 10000000
```

Indicates that no label checking of input tapes is to be performed. In this case

- The UPSI statement is required.
- A TLBL statement is not required. If submitted, its contents are not used by the program.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// Uff [IV=nnnnnn],[OV=mmmmmm],[NV=pppppp]
```

The parameters are not positional. You must not insert a comma for any omitted parameters.

`// U`	Utility modifier statement entry.
`ff`	Indicates the specific copy function to be performed.
`ff=DD`	Disk-to-disk function.
`ff=DT`	Disk-to-tape function.
`ff=TD`	Tape-to-disk function.
`IV=`	Identifies the input volume parameter.
`nnnnnn`	Indicates the 6-character volume serial number of the disk input volume. This parameter is required for the disk-to-disk and disk-to-tape functions. **Note:** The parameter is ignored by the program if it is submitted for the tape-to-disk function.
`OV=`	Identifies the output volume parameter.
`mmmmmm`	Indicates the 6-character volume serial number of the disk output volume before the program writes any information onto it. The parameter is optional. The parameter may be included for the disk-to-disk and tape-to-disk functions to provide a means of ensuring that the correct output disk volume is mounted before any data transfer takes place. **Note:** The parameter is ignored by the program if it is submitted for the disk-to-tape function.
`NV=`	Identifies the new volume parameter.
`pppppp`	Indicates the 6-character new volume serial number to be written on the output disk volume. The parameter is optional. If the parameter is omitted for the disk-to-disk and tape-to-disk functions, the volume serial number read from the input disk (either by the disk-to-disk or disk-to-tape function) is written on the output disk. See item 1 under *Precautions* later in this section. **Note:** The parameter is ignored by the program if it is submitted for the disk-to-tape function.

Examples

Following are examples of various parameter combinations of the utility modifier statement for the disk-to-disk function.

Example 1.

```
// UDD IV=111111
```

The input disk volume is checked for the volume serial number 111111. If this is correct, the program continues and the volume serial number is copied to the output disk volume.

If the number read from the input disk does not match 111111, a message is issued to the operator, who then has the following options:

- Mount the pack with volume serial number 111111 and enter the appropriate response on the console keyboard to continue processing.
- Submit a new utility modifier statement with the correct volume serial number to continue processing.
- Cancel the job.

Example 2.

```
// UDD IV=111111,NV=333333
```

The input disk volume is checked for the volume serial number 111111 as in example 1. The output disk volume will have the volume serial number 333333 written on it. See 1 in the section *Precautions.*

Example 3.

```
// UDD IV=111111,OV=222222
```

The input disk volume is checked for the volume serial number 111111 as in example 1. The output disk volume is checked for the volume serial number 222222 before any data is written upon it. If the number matches, the program continues. The volume serial number of the input disk volume is copied onto the output disk volume.

If 222222 does not match the number on the output disk volume, a message is issued and the operator is given the same options as described in the first example.

Example 4.

```
// UDD IV=111111,OV=222222,NV=333333
```

The input disk volume is checked for the volume serial number 111111 as in example 1. The output disk volume is checked for the volume serial number 222222 before any data is written on it. This is a programmed check to ensure that the proper output disk volume has been mounted.

The output disk volume will have the volume serial number 333333 written on it. This allows you to specify a particular volume serial number for the output pack rather than accepting the default of copying the volume serial number from the input disk volume.

Note that OV and NV can be the same number. This invokes a programmed check for the correct output pack and effectively results in retaining that same volume serial number of the output disk volume. See 1 in the section *Precautions.*

Precautions

1. Use of NV parameter

If this parameter is used, the Fast Copy program inserts the new-volume serial number only into the VOL1 label. The program writes file labels on the output volume exactly as they appear in the VTOC of the disk volume being copied. If the volume being copied is a SYSRES pack, Fast Copy makes no change to the standard label information as it is written on the output disk.

If the output disk pack is to be used directly as DOS/VS SYSRES, libraries, or data files, it is advisable to retain the volume serial number of the input pack. The simplest way of doing this is to omit the NV parameter from the utility modifier statement.

2. Abnormal termination

If, during the disk-to-disk or tape-to-disk functions, the Fast Copy program should abnormally terminate between the issuance of the COPY IN PROGRESS message (8F01I) and the COPY COMPLETED message (8F27I), the Initialize Disk program may have to be run on the output pack before attempting to use it again.

3. IBM 3330: Copying IBM 3330 Model 1 to Model 11

The IBM 3330 Model 1 may be copied to either a Model 1 or a Model 11. When copying to a Model 11, the 404 cylinders of the Model 1 are transferred to the first 404 cylinders of the Model 11. The remaining 404 cylinders of the Model 11 are not overwritten, but any data in that area can be accessed only through physical IOCS. The VTOC copied from the Model 1 will contain labels only for the copied data. The VTOC that existed on the Model 11 prior to the copy operation will no longer be accessible by DOS/VS system programs or the data management access methods.

Caution: Since ISAM does not support the IBM 3330 Model 11, an ISAM file copied onto a Model 11 volume will no longer be accessible.

4. IBM 3340: Copying IBM 3348 Model 35 to Model 70

The IBM 3348 Model 35 may be copied either to a Model 35 or to a Model 70. When copying to a Model 70, the 348 cylinders of the Model 35 are transferred to the first 348 cylinders of the Model 70. The next 348 cylinders of the Model 70 are not physically overwritten, but any residual data in that area is unaccessible except through physical IOCS. The VTOC

copied from the Model 35 will only contain labels for the copied data. The VTOC that existed on the Model 70 prior to the copy operation will no longer be accessible by DOS/VS system programs or the data management access methods.

Integrated Version

The program is contained in the core image library. If it has been deleted from the core image library, it can be re-created from the relocatable library; run the following job to catalog the program into the core image library using the IBM provided procedure LINKFCY:

```
// JOB CATALOG FASTCOPY INTO CIL
// EXEC PROC=LINKFCY
/&
```

Storage Requirement and Execution Mode

The integrated version of the Fast Copy program should be run in real mode to achieve maximum execution efficiency and to minimize impact on supervisor requirements. In real mode, the program requires allocation of real storage as follows:

Device	*Real Storage*
IBM 3330	24K
IBM 3340	20K
IBM 3350	30K

For execution in virtual mode, you must consider the effect that this utility may have on supervisor assembly macro specifications. The BUFSIZE parameter of the VSTAB macro must take into account the theoretical maximum size of the channel programs generated by the Fast Copy program. The number of copy blocks required by the Fast Copy program is device dependent:

Device	*No. of Copy Blocks*
IBM 3330	30
IBM 3340	17
IBM 3350	32

See *DOS/VS System Generation,* GC33-5377, for information on the VSTAB macro.

Control Statement Examples

1. Copy Disk to Disk

```
// JOB COPY 3330 TO 3330
// ASSGN SYS004,X'160'     (input disk)
// ASSGN SYS005,X'161'     (output disk)
// EXEC FCOPY,REAL
// UDD IV=DOSRES
/&
```

2. Copy Disk to Tape

```
// JOB COPY 3340 TO TAPE
// ASSGN SYS004,X'160'      (input disk)
// ASSGN SYS005,X'280'      (output tape)
// ASSGN SYS005,X'281',ALT (alternate tape)
// TLBL UOUT,'BACKUP TAPE'
// EXEC FCOPY,REAL
// UDT IV=111111
/&
```

3. Copy Tape to Disk

```
// JOB RESTORE BACKUP TAPE TO DISK
// ASSGN SYS004,X'280'         (input tape)
// ASSGN SYS004,X'281',ALT     (alternate tape)
// ASSGN SYS005,X'160'         (output disk)
// TLBL UIN,'BACKUP TAPE'
// EXEC FCOPY,REAL
// UTD
/&
```

Stand Alone Versions

The stand-alone versions of the Fast Copy program are distributed in card-image format in the DOS/VS source statement library. They are cataloged in the sublibrary designated Z under the book names FASTCOPY (card-resident version) and FASTCOPD (diskette-resident version). The program DKTTIPL, which is used only to obtain a diskette-resident version, is a member of the core image library.

The phase name, used as the operand of the EXEC job control statement, is FCOPY.

Obtaining the Card-Resident Version

The object program is contained in the source statement library of the DOS/VS distribution system. It can be punched into cards by submitting a DOS/VS job made up of the following statements:

```
// JOB PUNCH STAND ALONE FASTCOPY DECK
// ASSGN SYSPCH,X'00D' (see Note)
// EXEC SSERV
   PUNCH Z.FASTCOPY
/*
/&
// PAUSE REMOVE FIRST 2 AND
   LAST 2 CARDS FROM PUNCHED DECK
```

Note: Assign SYSPCH to a card punch device. Include ASSGN statements for SYSIPT, SYSLST, and SYSLOG if the current assignments are not those required.

The first two cards of the punched deck contain CATALS and BKEND in the first punched positions. The last two cards contain BKEND and /*. All four cards should be removed before the stand-alone card deck is used.

Running the Card-Resident Version

Job control card and the utility modifier statement can now be prepared and inserted into their proper locations in the card deck. The deck is made up of five sections, each identified with a letter punched in card columns 73-76. Within each section, the cards are sequentially numbered, starting with 0001, in columns 77-80. The five sections, and the identification of the first card in each section is as follows:

Card Component	*Columns 73-80*
IPL	SDUAIPC1
Supervisor	SDUB0001
Job Control	SDUC0001
Utility-Phase 1	UUUU0001
Utility-Phase 2	ZZZZ0001

Job control cards are placed in the deck immediately in front of card UUUU0001; the utility modifier statement must immediately precede card ZZZZ0001. (See Figure 11.3.) The complete deck is then placed in the card reader, IPL from the card reader is performed, and the program is loaded and executed.

Obtaining the Diskette-Resident Version

Two programs, namely DKTTIPL and Z.FASTCOPD, are needed to obtain the stand-alone diskette version. DKTTIPL writes only the first two IPL records on sectors 1 and 2 of track 0 of the diskette; Z.FASTCOPD writes the rest of the stand-alone diskette-resident version on the diskette, starting on track 1.

The stand-alone diskette-resident version can be obtained by submitting a DOS/VS job made up of the following statements:

```
// JOB WRITE STAND ALONE FASTCOPY DISKETTE
// ASSGN SYS006,X'cuu'     (see Note 1)
// EXEC DKTTIPL
/*
// DLBL IJSYSPH,[file-id],[yy/ddd]   (see Note 2)
// EXTENT
ASSGN SYSPCH,X'cuu'        (see Note 1)
// EXEC SSERV
   PUNCH Z.FASTCOPD
/*
CLOSE SYSPCH,UA
/&
```

Note 1: Assign SYS006 and SYSPCH to the same diskette I/O unit.

Note 2: yy/ddd is the expiration date of the diskette-resident version.

Running the Diskette-Resident Version

The stand-alone diskette is placed in the diskette I/O unit, and IPL is performed from this unit. When the program enters the WAIT state, press ENTER on the display operator console (DOC), and the Z.FASTCOPD program will be loaded and executed. During execution you will be request-

ed to enter the appropriate job control statements and later the utility modifier statement via the DOC.

Job Control Statements

Some of the stand-alone supervisor job control conventions differ slightly for the Fast Copy program. These differences are:

- The ASSGN statement has a modified format.
- The PAUSE statement may be placed anywhere between the JOB and the EXEC statement but will cause the system to pause only after the EXEC statement has been read.

A list of requirements for valid job control statements is given in Figure 11.1. Detailed information regarding job control statements may be found in *DOS/VS System Control Statements,* GC33-5376.

ASSGN Statement

The format and entries of the ASSGN statement are:

```
// ASSGN SYSxxx,X'cuu',dd,X'ss'[,ALT]
```

SYSxxx	Identifies the logical unit being assigned.
xxx=LOG	Card resident version: Must be a console printer-keyboard or a display operator console. Diskette-resident version: Is automatically assigned to the display operator console. Any ASSGN statement for SYSLOG will be ignored by the program.
xxx=004	Must be the input disk or tape unit.
xxx=005	Must be the output disk or tape unit.
xxx=LST	May be asssigned to the printer. In this case, all messages issued by the Fast Copy program will be written on SYSLST in addition to SYSLOG. This assignment is useful when SYSLOG is assigned to the display operator console and a printer is not attached to it or is not being used.
xxx=RDR	Card-resident version: Is automatically assigned to the card reader from which IPL was performed. This assignment is used only if you wish to switch to another card reader for reading the remaining job control cards and the Fast Copy program deck (cards with UUUU or ZZZZ in columns 73-76).

JOB CONTROL STATEMENT	DISK-TO DISK	DISK-TO-TAPE TAPE-TO-DISK	COMMENTS
// JOB	Required		Operand is name of job as assigned by user; may consist of up to 8 alphameric characters.
// ASSGN SYSLOG	Required for card version; for diskette version, automatically assigned to the DOC.		Required format is described in *ASSGN statement.*
SYS004, SYS005	Required		
SYSLST	Optional		
SYSRDR, SYSIPT	Optional for card version; for diskette version, automatically assigned to the DOC and the diskette I/O unit respectively.		
// LOG	Optional for card version. Not used for diskette version.		NOLOG is effective as default. The LOG statement may be used any time after SYSLOG has been assigned.
// NOLOG	Optional		
// DATE	Required		Format of operand mm/dd/yy
// UPSI	Not required	Required if no label processing for tape. Optional if tape label processing is desired.	Format is described in *Tape Processing.*
// TLBL	Not required	Required if tape labels are to be processed.	Format described in *DOS/VS System Control Statements,* GC33-5376.
// PAUSE	Optional		Causes the system to pause *after the EXEC statement has been read* and before Fast Copy is loaded for execution.
// EXEC	Required		Operand may be FCOPY or left blank.

Figure 11.1 **Summary of job control statements used in the stand-alone Fast Copy Disk Volume program**

Diskette-resident version:
Is automatically assigned to the display operator console. Any ASSGN statement for SYSRDR will be ignored by the program

	xxx=IPT	Card-resident version: Is automatically assigned to the card reader from which IPL was performed. This assignment is used only if you wish to read the utility modifier statement from a different card reader. Diskette-resident version: Is automatically assigned to the display operator console. Any ASSGN statement for SYSIPT will be ignored by the program. **Note:** Normally, with the card-resident version, the entire card deck, including the inserted control cards, is read from one card reader. In this case, assignment cards for SYSRDR and SYSIPT are not required.
X'cuu'		Channel and unit number in hexadecimal representation according to standard DOS/VS notation.
dd		Identifies the device type parameter.
	dd=C1	Console printer-keyboard. (Also specify C1 when using the 5213 console printer with the display operator console in 1052 mode on Model 125).
	dd=CR	Display operator console. No console printer output is issued, even when a 5213 console printer is attached.
	dd=L1	1403 or 1404 Printer.
	dd=L2	1443 or 1445 Printer.
	dd=L3	3203, 3211, or 5203 Printer.
	dd=L4	3800 Printing Subsystem.
	dd=R1	2540 Card Read Punch (reading only).
	dd=R2	2540 Punch/Read Feed Feature.
	dd=R3	1442 Card Read Punch (reading only).
	dd=R4	2501 Card Reader.
	dd=R5	2520 Card Read Punch (reading only).
	dd=R6	3504/3505 Card Reader.
	dd=R7	2560 Multi-Function Card Machine 5425 Multi-Function Card Unit
	dd=T1	7-track 2400/3400 Series Magnetic Tape Unit*.
	dd=T2	9-track 2400/3400 Series Magnetic Tape Unit*.

*The IBM 2495 Tape Cartridge Reader does not belong to the IBM 2400 Series Magnetic Tape Units.

dd=D4 — 3330 (except Model 11).

dd=D5 — 3340 Direct Access Storage Facility Model 35 Data Module.

dd=D6 — 3340 Direct Access Storage Facility Model 70 Data Module.

dd=D7 — 3330 Disk Storage Model 11

dd=D8 — 3350 Direct Access Storage Facility

Note: For a 3350 in 3330-1 compatibility mode, D4 must be specified. For a 3350 in 3330-11 compatibility mode, D7 must be specified.

X'ss' — Identifies the device specification parameter. Optional for all devices except magnetic tape.

ss=00 — For devices other than a magnetic tape unit (optional).

For Printers:

ss=7B — Sense data checks

ss=73 — Block data checks (default)

For 7-track tapes:

ss=10 — for 200 bpi

ss=50 — for 556 bpi

ss=90 — for 800 bpi.

Each of these alternatives handles odd parity with the translate feature off and data convert on.

For 9-track tapes:

ss=D0 — for 6250 bpi on a single or dual density drive

ss=C0 — for 1600 bpi on a single or dual density drive

ss=C8 — for 800 bpi.

ALT — Must be specified if an alternate tape drive is assigned for multiple reel input or output (see examples in Figures 11.3 and 11.4).

Restriction: The information for the alternate drive is stored as for SYS006. It is not allowed to use multiple alternate assignments, or an alternate assignment and an ASSGN to SYS006, in one job step. Only the last assignment is kept; no warning message is issued.

Note: When you need more than two tapes, mount the odd sequence tapes on the first drive after rewind/unload, and do the same for the even sequence tapes on the alternate drive.

Control Statement Stream

Figure 11.2 shows the sequence of operations when the diskette-resident version of the Fast Copy program is used. Illustrated are sample statements for the disk-to-disk function.

Figure 11.3 shows a representation of the Fast Copy deck with the inserted job control statements and the utility modifier statement. Illustrated are sample statements for the disk-to-tape function. Figure 11.4 shows sample statements for the tape-to-disk function.

In each of these two examples

- Standard tape labels are used.
- The UPSI statement could be omitted, because these particular bit settings indicate that standard labels are to be processed, which is the system default.
- Alternate tape drives are used.

After IPL from the diskette has been performed, the following message appears on the screen:

```
*
***** ENTER JOB CONTROL STATEMENTS NECESSARY TO
***** EXECUTE FAST COPY
*
```

You enter the following job control statements one by one:

```
// JOB FAST COPY DISK-TO-DISK
// DATE 05/23/76
// ASSGN SYS004,X'180',D5
// ASSGN SYS005,X'181',D6
// EXEC FCOPY
```

The following message appears on the screen:

```
***** ENTER UTILITY MODIFIER STATEMENT
```

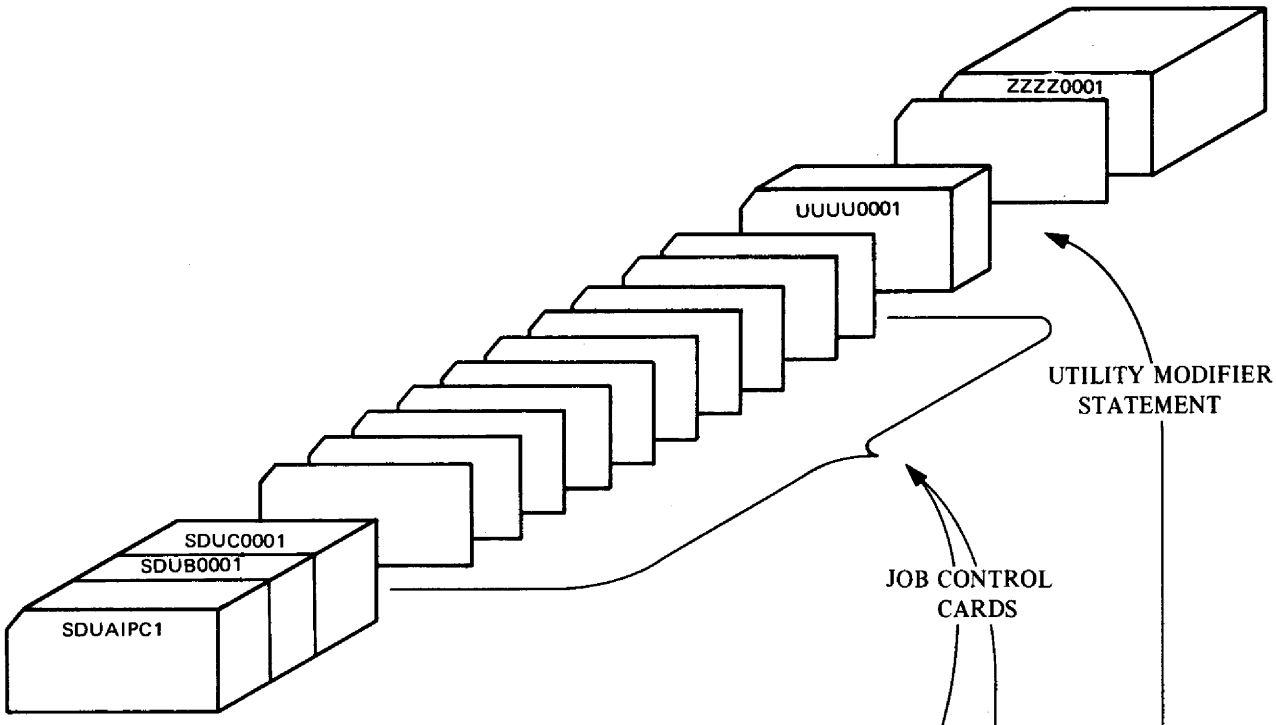
You enter:

```
// UDD IV=111111,OV=333333,NV=222222
```

After the input and output devices have been checked, the following message appears on the screen:

```
8F01I COPY 3340-35 TO 3340-70 IN PROGRESS
```

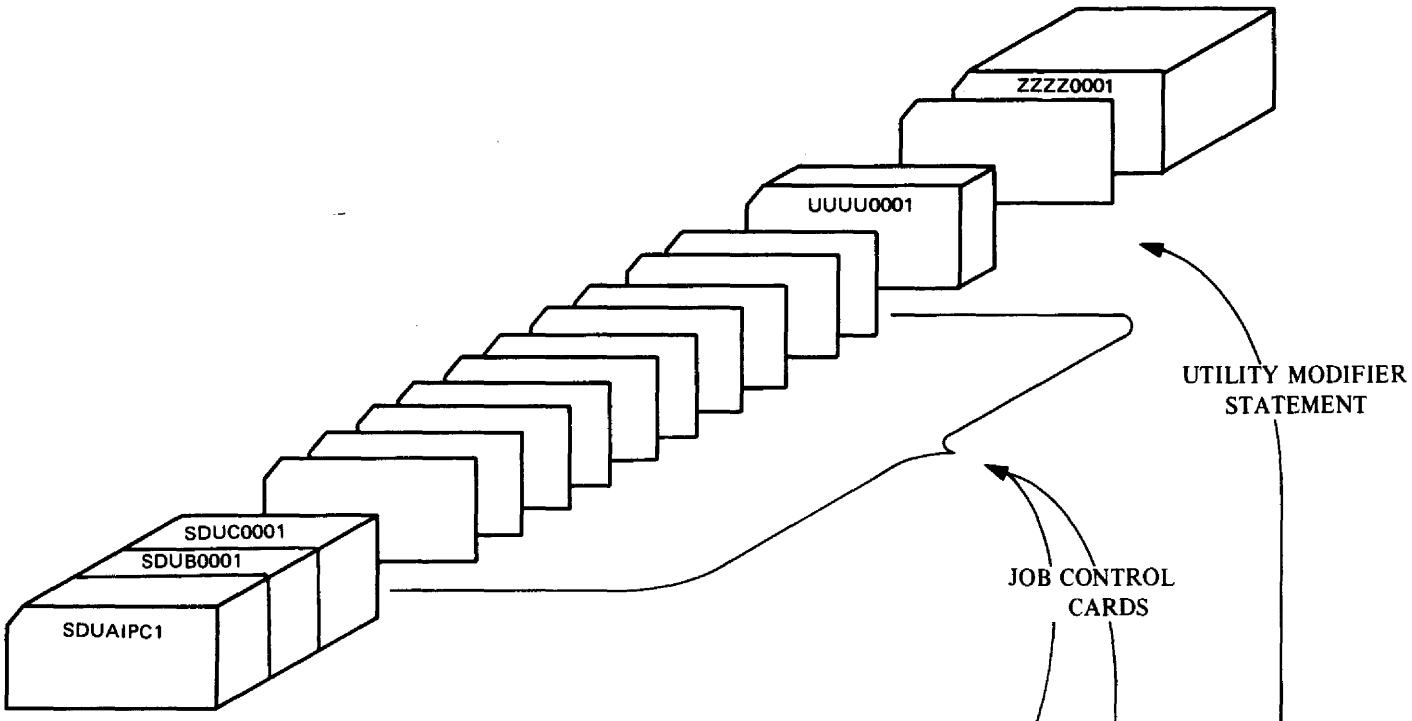
Figure 11.2 **Sample control statements for the disk-to-disk function using the diskette-resident version.**




```
// JOB FAST COPY DISK-TO-TAPE
// ASSGN SYSLOG,X'01F',C1
// LOG
// UPSI 00000000
// TLBL UOUT,'FASTCOPY DUMPRSTR'
// DATE 08/22/73
// ASSGN SYS004,X'231',D4
// ASSGN SYS005,X'180',T2,X'C0'
// ASSGN SYS005,X'181',T2,X'C0',ALT
// EXEC FCOPY


// UDT IV=111111
```

Figure 11.3 **Sample control statements for the disk-to-tape function using the card-resident version**




```
// JOB FAST COPY TAPE-TO-DISK
// ASSGN SYSLOG,X'01F',C1
// LOG
// UPSI 00000000
// TLBL UIN,'FASTCOPY DUMPRSTR'
// DATE 08/23/73
// ASSGN SYS004,X'180',T2,X'C0'
// ASSGN SYS004,X'181',T2,X'C0',ALT
// ASSGN SYS005,X'230',D4
// EXEC FCOPY


// UTD OV=333333
```

Figure 11.4 **Sample control statements for the tape-to-disk function using the card-resident version**

CHAPTER 12
INITIALIZE DATA CELL

Description

Purpose:

- To prepare from one to five new or expired cells for use on an IBM 2321 Data Cell Drive.

Preparation: For each of the cells consists of:

- VTOC (volume table of contents) label checking.
- HA (home address) generation.
- R0 (track descriptor record) generation.
- Surface analysis and initialization verification.
- Volume label creation.
- IPL and VTOC format creation.

VTOC Label Checking

Before a cell is initialized, a check is made to see whether the VTOC is present. If the cell has been previously initialized and the VTOC is present, any labels in the VTOC are checked to see if the files on the data cell have expired. If there are un-expired files, a message is printed. You may either bypass that cell or initialize it in spite of this condition.

Home Address Generation

The program writes, in binary notation a five-byte home address on every track. The five tracks are:

- flag
- subcell number
- strip number
- cylinder number
- track number

Track Descriptor Record Generation

The R0 is the first record following the HA. It consists of a count and a data area, each containing eight bytes.

If an error occurs in the HA or R0 area, the track is flagged defective. An alternate track is assigned by advancing the HA and R0 800 bytes. If an error occurs at this time, the program is terminated.

Surface Analysis and Initialization Verification

Surface analysis is performed first in the alternate track area. When a track in the alternate track area is found defective, the track is flagged defective. It then cannot be assigned as an alternate track. The surfaces of all remaining tracks on the cell are analyzed. If a track is detected with a defective surface area, on which data cannot be written, an alternate track is assigned. Messages are printed on SYSLOG to inform you of the defective tracks, assigned alternate tracks, and their respective locations.

A cell is deleted from further processing when no more tracks are available for assignment.

Volume Label Creation

When you use the volume label control statement VOLn, a volume label is created in the standard format (VOL1) for each cell processed.

The VOL1 label is written on subcell 0, strip 0, cylinder 0, track 0, record 3 of each cell. Seven additional user volume labels (VOL2–VOL8) can be created. They are written on subcell 0, strip 0, cylinder 0, track 0, records 4 through 10. See section *Volume label control statement.*

IPL and VTOC Format Creation

The program formats two IPL records. They are written on subcell 0, strip 0, cylinder 0, track 0, records 1 and 2.

The first record is written with a 24-byte data field of binary zeros.

The second record is written with a 144-byte data field of binary zeros.

The program also formats the VTOC. The location on the cell in which the VTOC must be placed is indicated in the VTOC control statement.

The standard location of the VTOC is on subcell 0, strip 0, cylinder 0, immediately following the volume labels, and extending to the end of the cylinder. The VTOC can appear anywhere on the cell, except for the alternate track area, but cannot exceed cylinder boundaries. See section *VTOC control statement.*

Utility Modifier Statement

Contains information required to run the program.

The format and entries are:

```
// UIM CELLS=(n,n,n,n,n),Ix
```

`// U`	Utility modifier statement entry.
`IM`	Indicates the Initialize Data Cell program. Can be omitted.
`CELLS=`	Identifies the CELLS parameter. Is only used for data cell operations.
`(n,n,n,n,n)`	Each n is a decimal digit from 0 to 9, designating a cell number. From one to five cell numbers may be included.
`Ix`	Identifies the input option parameter.
`x=R`	Indicates that previously flagged tracks must retain their flags.
`x=A`	Indicates that the surface of all tracks must be analyzed, ignoring any previous flags.
	If this parameter is omitted, the default value is IR.

Label Control Set

Three types of label control statements, called a label control set, must be supplied:

- VTOC control statement
- volume label statement
- END statement

VTOC Control Statement

Contains control information necessary to create the VTOC.
The statement may be written in either

- standard format
- nonstandard format

Standard Format

The format and entry are:

```
// VTOC STANDARD
```

// VTOC	Identifies the VTOC control statement.
STANDARD	Indicates that the VTOC must be generated in the standard location.

Nonstandard Format

The format and entries are:

```
// VTOC STRTADR=(ccchhh),EXTENT=(n)
```

// VTOC	Identifies the VTOC control statement.
STRTADR=	Identifies the VTOC beginning address parameter.
$c_1c_2c_2c_3h_1h_2h_2$	Indicates the beginning address. c_1 = cell (0–9) c_2c_2 = subcell (00–19) c_3 = strip (0–9) h_1 = cylinder (0–4) h_2h_2 = track (00–19) These entries must be enclosed in parentheses. *Restriction:* You must not specify subcell 0, strip 0, cylinder 0, track 0, which is used for VOL labels and IPL records.
EXTENT=	Identifies the VTOC track number parameter.
(n)	Indicates the number of tracks (1 to 20) in the VTOC. This entry must be enclosed in parentheses.

Volume Label Control Statement

Contains information to create volume labels. A VOL1 control statement must be supplied for each cell. Up to seven additional volume label control statements (VOL2 through VOL8) for user volume labels can be supplied for each cell. The VOL2 through VOL8 statements must be sequentially entered.

The format and entries are:

`VOLnvolser`	(for columns 11–80 see the description below)

`VOL` — Label identifier. Must contain VOL to indicate that it is a volume label.

`n` — Volume label number. Indicates the relative position (1–8) of a volume label within a group of volume labels.

`volser` — Volume serial number. The identification code is assigned to a volume when it enters an installation. Normally a numeric value from 000001 through 999999, although any or all of the six characters may be alphabetic.

col. 11 — *Volume security.* Indicates the security status of the volume. Not used by DOS/VS.

col. 12–21 — *Data file directory.* The first five bytes contain the starting address of the VTOC, the last five bytes are blank.

col. 22–41 — Reserved.

col. 42–51 — *Owner name and address code.* Indicates a specific customer, installation, and/or system. Used by the VTOC Display program.

col. 52–80 — Reserved.

For more information on the VOLn label, see the publication *DOS/VS Data Management Guide,* GC33–5372.

Columns 5 through 10 are used for the volume serial number. It is recommended that you always specify six characters. If you specify less than six characters, the program pads the fields to the right with blanks (X'40').

Always ensure that all six characters are specified in the job control extent information, since the job control program pads an incomplete serial number to the left with zeros. Therefore, do not use / / EXTENT if the serial number contains any blanks; otherwise, the remainder of the information is ignored and default values are assumed. Instead, use the VOL, DLAB, and XTENT statements. You can use the / / XTENT statement to specify the six-character volume serial number within apostrophes.

END Statement

The last statement of the label control set must be followed by an END statement.

The format is:

```
// END
```

Control Statement Stream

An example of a control statement stream that is used to run the Initialize Data Cell program:

```
// JOB INITIAL
// ASSGN SYS000,X'293'
// EXEC INTDC
// UIM CELLS=(3,5,7)
// VTOC STRTADR=(3033303),EXTENT=(5)
VOL1222222
// END
// VTOC STANDARD
VOL1333333
// END
// VTOC STANDARD
VOL1444444
// END
/&
```

CHAPTER 13
INITIALIZE DISK

Description

Purpose:

- To prepare one complete disk pack for use.
- To change the volume label(s) and the VTOC (volume table of contents) address of a previously initialized disk pack (other than an emulator pack).
- To convert a work pack used for OS into work pack suitable for DOS/VS.

VTOC Label Checking

Before a pack is initialized, it is checked to see whether labels in the VTOC have expired or not. If a file is found which has not expired, a message is printed. If you still want to initialize the disk pack after you have received the message, you can continue processing. Otherwise the job is terminated.

The initialization of disk packs follows a slightly different pattern depending on the device type: initialization of the IBM 3330, 3340, and 3350 is described under *Quick Initialization;* initialization of the IBM 2311 and 2314 under *Initialization.*

Quick Initialization

The IBM 3330, 3340, and 3350 disk packs are preinitialized by IBM: surface analysis is performed, and the HA (home address) and R0 (track decriptor) records are written on each track. The quick initialization prepares:

- Two IPL records
- One or more volume labels
- The VTOC.

For details on these see the sections *Volume Label Creation* and *IPL and VTOC Format Creation.*

For 3340 and 3350 devices, flagged defective primary tracks will be analyzed and reclaimed if no errors are detected. This does not apply to 3350 devices in 3330-1 or 3330-11 compatibility mode.

The quick initialization is invoked if you specify IQ for the input option parameter in the utility modifier statement.

Initialization

The following actions are performed during the initialization of an IBM 2311 or 2314 disk pack depending on the specified options:

- Home Address (HA) generation
- Surface analysis
- Tack Descriptor (R0) generation
- Volume label creation
- IPL and VTOC format creation.

Home Address Generation

The program writes the HA as follows:

- flag (1 byte)
- cylinder number (2 bytes)
- head number (2 bytes).

The flag byte is transmitted to the flag byte of each record on the track. It indicates the condition of the track:

	0	0	0	0	0	0	0	0
	–	–	–	–	–	–	1	1
bit	0	1	2	3	4	5	6	7

Bits 0-5 are 0.

Bit 6: 0 if it is a good track,
1 if it is a defective track.

Bit 7: 0 if it is not an alternate track,
1 if it is an alternate track.

HA generation requires that the input option parameter in the utility modifier statement be IA or IR.

The job is cancelled when it is impossible to write an HA on a pack.

Surface Analysis

Surface analysis is performed depending on the specified option. You may specify in the utility modifier statement:

- IA –Surface analysis of all tracks, including those already flagged defective.
- IR –Tracks already flagged defective without surface analysis.
- IQ –Surface analysis is skipped.
- IS –Surface analysis, HA and R0 generation are skipped.

Surface analysis is first performed on the alternate tracks (cylinders 200–202) and then on the remaining tracks. Tracks found to be defective are flagged defective. The program flags a track defective by setting bit 6 in the flag byte of the HA to 1. See section *Home Address Generation.*

Alternate tracks are assigned to all defective tracks, except defective alternate tracks (defective alternate tracks are not assigned). The program assigns an alternate track by writing:

- The address of the alternate track in the R0 count field of the defective track.
- The address of the defective track in the R0 count field of the alternate track.

The program terminates processing when a defective track is found and no more alternate tracks are available. The defective tracks are logged to provide a record of the condition of each pack processed.

Track Descriptor Record Generation

R0 is the first record that follows the HA. It is divided into two parts:

- count
- data

The count field, eight bytes long, is written as follows:

- cylinder number (2 bytes)
- head number (2 bytes)
- record number (1 bytes)
- key length (1 byte)
- data length (2 bytes).

The data field, eight bytes long, is written as follows:

- cylinder number (2 bytes)
- head number (2 bytes)
- record number, which is zero (1 byte)
- number of bytes remaining on the track (2 bytes)
- binary zeros (1 byte).

If R0 cannot be written on a track, a message is printed to identify the error and the job is cancelled.

Volume Label Creation

A volume label is created on each disk pack. It consists of a VOL1 label and seven additional (optional) labels (VOL2–VOL8).

The VOL1 label is written on cylinder 0, track 0, record 3 of the disk pack. Seven additional user volume labels (VOL2–VOL8) can be created. They are written on cylinder 0, track 0, records 4 through 10. See section *Volume label control statement.*

IPL and VTOC Format Creation

The program formats two IPL records. They are written on cylinder 0, track 0, records 1 and 2.

The first record is written with a 24-byte data field of binary zeros.

The second record is written with a 144-byte date field of binary zeros.

The program also preformats the VTOC. The location on the disk in which the VTOC must be placed is indicated in the VTOC control statement.

The standard location of the VTOC (for non-SYSRES packs) is on cylinder 0, track 0, immediately following the volume label(s), and extending to the end of the cylinder (see Note). The VTOC can appear on any cylinder, except for the alternate cylinders, but cannot exceed a cylinder boundary.

A VTOC placed anywhere other than in the standard location can occupy any number of tracks you desire on that cylinder. The first record begins in the first record of the first track. The last record appears as the last record of the last track specified.

Each record of the VTOC contains a 44-byte key field and a 96-byte data field, written as binary zeros. The first two records of the VTOC are reserved for two specific records:

- The data set control block label (format 4) of the VTOC (VTOC–DSCB).
- The space management label (format 5).

The first four bytes of the key field of the space management label are written as hexadecimal fives (05). The first byte of the data field of the space management label is written in hexadecimal representation F5.

Note: Problem programs may not access cylinder 0, track 0.

Converting a Work Pack

If you specify IS in the input option parameter of the utility modifier statement, surface analysis, HA (home address), and R0 (track descriptor record) generation are bypassed. This option assumes that a valid VTOC is present. A work pack used for OS can thus be converted into a work pack suitable for use by DOS/VS.

Utility Modifier Statement

Contains information required to run the program.

The format and entries are

for the IBM 2311 and 2314:

```
// UID Ix,Cn,R=(ccchhh)
```

for the IBM 3330, 3340, and 3350:

```
// UID Ix
```

If this statement is omitted, the following default values are assumed:

```
// U IR,C1
```

`// U`	Utility modifier statement entry.
`ID`	Indicates the Initialize Disk program. Can be omitted.
`Ix`	Identifies the input option parameter.
`x=R`	Previously flagged tracks retain their flags. You should give preference to IR over IA. IA might remove the flag from a marginal track and cause errors later or on another disk drive. *Restriction:* You cannot use this option for the IBM 3330, 3340, and 3350.
`x=A`	The surface of all tracks must be analyzed. Previous flags are ignored. *Restriction:* You cannot use this option for the IBM 3330, 3340, and 3350.

x=S	Surface analysis, HA, and RC generation are skipped. IPL records, volume label(s), and VTOC are written if a valid VTOC is already present. For initialized work packs, this entry can be used to change the volume label(s) and VTOC address and/or to convert a work pack used for OS into a work pack suitable for DOS/VS. When the program does not find a VOL1 label or a format-4 label, the job is cancelled. The pack is assumed not to be initialized. *Restriction:* This entry cannot be used for an emulator pack.
x=Q	Quick initialization. R0, the IPL records, volume label(s), and the VTOC are written. *Restriction:* You can use this option only for the IBM 3330, 3340, and 3350.
Cn	Identifies the count option parameter. This parameter is ignored if IS or IQ is specified.
n	Enter a decimal value of 1 to 255. This number indicates the number of times that the surface of the track must be analyzed. Program run time increases in proportion to the value of n. *Restriction:* You cannot use this option for the IBM 3330, 3340, and 3350.
R=	Identifies the replace option parameter. This parameter is ignored if IS or IQ is specified.
(ccchhh)	Indicates the physical location of a track that must be unconditionally flagged defective. cccc = cylinder number (decimal) hhh = head number (decimal). These entries must be enclosed in parentheses. The replace option parameter may be repeated any number of times up to the total number of available alternate tracks. The way to repeat this option is:

```
R=(ccchhh),R=(ccchhh),
R=(ccchhh),...
```

This option may be continued on the subsequent statement. To do this, enter a character other than blank in column position 72 and begin you continuation in column position 16.

Note: Each card or card image must contain a full R=(ccchhh) parameter.

Label Control Set

Three types of label control statements, called a label control set, must be supplied:

- VTOC control statement
- volume label statement
- END statement.

VTOC Control Statement

Contains control information necessary to create the VTOC. The statement may be written in either

- standard format
- nonstandard format.

Standard Format

The format and entry are:

```
// VTOC STANDARD
```

// VTOC	Identifies the VTOC control statement.
STANDARD	Indicates that the VTOC must be generated in the standard location, on cylinder 0, immediately following the volume label(s). This option must not be used for SYSRES packs. If you intend not to process the VTOC with the standard OPEN macro, the VTOC should be generated in a location other than the standard location. Problem programs may not access cylinder 0, track 0.

Nonstandard Format

The format and entries are:

```
// VTOC STRTADR=(ccccchhh),EXTENT=(n)
```

`// VTOC` — Identifies the VTOC control statement.

`STRTADR=` — Identifies the VTOC beginning address parameter.

`(ccccchhh)` — Indicates the beginning address.
cccc = cylinder number
hhh = head number.

Cylinder and head number must be entered in decimal representation: in the standard location, on cylinder 0.

cylinders:	for device:
0-199	IBM 2311, 2314
0-403	IBM 3330 (except Model 11)
0-807	IBM 3330 Model 11
0-347	IBM 3340 35MB
0-695	IBM 3340 70MB
0-554	IBM 3350

tracks/ heads:	for device:
0-9	IBM 2311
0-19	IBM 2314
0-18	IBM 3330
0-11	IBM 3340
0-29	IBM 3350

Restriction: Problem programs may not access cylinder 0, track 0.

These entries must be enclosed in parentheses.

`EXTENT=` — Identifies the VTOC track number parameter.

`(n)` — Indicates the number of tracks in the VTOC. This number must be given in decimal representation.

The maximum number of tracks is
10 for the IBM 2311
20 for the IBM 2314
19 for the IBM 3330
12 for the IBM 3340
30 for the IBM 3350

This entry must be enclosed in parentheses.

Note: The number of tracks specified must not be greater than the number of tracks that exist between the beginning address (STRTADR) and the end of the cylinder.

Volume Label Control Statement

Contains information to create volume labels. A VOL1 control statement must be supplied. Up to seven additional volume label control statements (VOL2-VOL8) for the volume labels can be supplied. The VOL2 through VOL8 statements must be entered sequentially.

The format and entries are:

```
VOLnvolser  (for columns 11-80 see the description below)
```

VOL	Label identifier. Must contain VOL to indicate that it is a volume label.
n	Volume label number. Indicates the relative position (1-8) of a volume label within a group of volume labels.
volser	Volume serial number. Identification code assigned to a volume when it enters an installation. Normally a numeric value from 000001 through 999999, although any or all of the six characters may be alphabetic.
col. 11	*Volume security.* Indicates the security status of the volume. Not used by DOS/VS.
col. 12-21	*Data file directory.* The first five bytes contain the starting addresses of the VTOC, the last five bytes are blank.
col. 22-41	Reserved.
col. 42-51	*Owner name and address code.* Indicates a specific customer, installation, already in dataset and/or system. Used by the VTOC Display program.
col. 52-80	Reserved.

For more information on the VOLn label, see the publication *DOS/VS Data Management Guide,* GC33-5372.

Columns 5 through 10 are used for the volume serial number. It is recommended that you always specify all six characters. If you specify less than six characters, the Initialize Disk program will add blanks on the right. Job control pads incomplete volume serial numbers with zeros, on the left. As a result, the volume serial numbers do not match.

The Initialize Disk program also accepts VOL, DLAB, and XTENT statements, except for the IBM 3330, 3340, and 3350.

END Statement

The last statement of the label control set must be followed by an END statement. The format is:

```
// END
```

Control Statement Stream

An example of a control statement stream that is used to run the Initialize Disk program for non-SYSRES volumes:

```
// JOB INITIAL
// ASSGN SYS000,X'191'
// EXEC INTDK
// UID IR,C1,R=(0027003) (not valid for 3330, 3333, and 3340)
// VTOC STANDARD
VOL1222222
// END
/&
```

To initialize a SYSRES volume, the // VTOC statement should specify the location, such as:
// VTOC STRTADR=(0695000), EXTENT=(12).

When the disk being initialized is an IBM 3330, 3340, or 3350, the // UID statement in the above example should read: // UID IQ.

Note for 1401/1440 System/370 Emulation: When you initialize an IBM 2311 or 2314 disk pack to be used as a stacked disk pack by the 1401/1440 System/370 Emulators, you must include an UPSI statement before the EXEC statement in the control statement stream. This UPSI statement must have the following format:

```
// UPSI 00000001
```

This statement allows cylinder 200 to be used for emulator data instead of being part of the alternate track area.

Restriction: You cannot use the UPSI statement for the IBM 3330, 3340, and 3350.

CHAPTER 14
INITIALIZE TAPE

Description

Purposes:

- To write one to eight IBM standard tape volume labels in numerical sequence, followed by one dummy header label and one tapemark on EBCDIC tapes.
- To write one American National Standard Label, followed by one dummy header label and one tapemark on ASCII tapes. ASCII tapes use only VOL1.

  (ASCII = American National Standard Code for Information Interchange).

In order to perform standard label checking on tapes, standard volume labels must be present. These labels are prepared by this program. The labels are checked when they have been prepared. The format of the two standard labels is described in the publication *DOS/VS Data Management Guide*, GC33–5372.

There are two options to create standard labels on tape:

- You must use a single control statement to provide the starting volume serial number, the owner name and address identification, and the protection code. This information is written on the first tape supplied. The same volume label is written on each succeeding tape supplied with the job, except that the volume serial number is incremented by one.
- You may write unique volume labels on each tape by using a volume label image statement. You may supply up to eight volume label image statements for EBCDIC tapes. Only one volume label image statement may be supplied for ASCII tapes.

  This statement is the exact image of the 80-character label.

  You must use this option when the volume serial number field contains any alphameric values.

The program also prepares a dummy header label, immediately followed by a tapemark.

As each tape is initialized, its volume label(s) and tape unit number are printed on SYSLOG. This means that the initialization is completed. The tape may then be either rewound or rewound and unloaded as specified by you.

You must assign the tape units in ascending order, starting with SYS000. After the last unit has been assigned, the next sequential logical unit should be unassigned (or assigned to a unit other than a tape drive) to ensure proper termination. In this way, unintentional initialization is avoided.

If there are additional tape units on line (the maximum number is sixteen), the tape on the next higher unit (SYS001, SYS002, etc.), is initialized. If there are no more additional units, one of the following two steps is taken:

- If you specified the rewind option, the job is terminated.
- If you did not specify the rewind option, the program waits until a new tape is mounted on the first output unit (SYS000). When a new tape has been mounted, tape initialization begins a new cycle of the output units.

If the last statement is read before a cycle of assigned units has been read, the tape on the current unit is completed and the job is terminated.

Job Control Statements

Once you have defined these items, they cannot be changed during the execution of the program. If the required units for the program are not defined, the program terminates.

The following job control statements are used for system assignment:

`// JOB`	Required.
`// ASSGN`	Required (if not permanently assigned).
`SYSIPT`	Must be assigned.
`SYSLOG`	Must be assigned for output messages.
`SYS000`	Must be assigned as the first output tape unit.
`SYS001-SYS015`	Optional. May be assigned as additional output tape units.
`// EXEC`	Required.

Utility Control Statement

Contains information required to run the program.

The format and entries are:

```
// INTT CARD,REWIND,A,SERIAL=(xxxxxx),P=(x),
    CODE=(xxxxxxxxxxxxxx)
```

The parameters must be specified in the order stated. If an optional parameter is not used, you must *not* insert a comma for the omitted parameter. For example:

```
// INTT REWIND,SERIAL=(xxxxxx),
    CODE=(xxxxxxxxxxxxxx)
```

`// INTT`	Identifies the initialize tape control statement. This entry is required.
`CARD`	Indicates the volume label image option parameter. If specified, volume label image statements (only one for ASCII tapes) must be supplied. If omitted, the program assumes that volume information for every initialized tape must be taken from the INTTP control statement. This parameter is optional.
`REWIND`	Indicates that each tape must be rewound at completion of initialization. If omitted, the program assumes that the tapes must be rewound and unloaded at completion of initialization. This parameter is optional.
`A`	Indicates that all tape labels must be recorded in the ASCII mode and will observe the ANSI standards (ANSI= American National Standard Institute). If omitted, the program assumes EBCDIC mode and IBM standards. This parameter is optional. It can only be specified if the utility is to be run under a supervisor with ASCII support.

Volume Label Image Option Not Specified

The following parameters are examined if you did *not* specify the volume label image option:

`SERIAL`	Identifies the volume serial number parameter.
`(xxxxxx)`	Indicates a six-character numeric field that represents the starting value for the volume serial number. This number is incremented

by one for each additional tape initialized after the first.
This entry must be enclosed in parentheses.

`P=` — Identifies the protection option parameter.

`(x)` — For *EBCDIC* tapes, this parameter indicates that the volume security fields of all volumes must be given the specified protection value.
A value of 1 indicates protection. If omitted, each volume is given a value of 0, indicating no protection.

For *ASCII* tapes, this parameter indicates that the accessibility fields of all volumes must be given the specified protection value.
A value of 1 indicates no accessibility. If omitted, a space is written, indicating unlimited accessibility.
This entry must be enclosed in parentheses.
This parameter is optional.

`CODE=` — Identifies the CODE parameter.
This parameter is required.

`(xxxxxxxxxx)` — For *EBCDIC* tapes, this is a *ten*-character alphameric value, representing the owner's name and address code.
This entry must be enclosed in parentheses.

`(xxxxxxxxxxxxxx)` — For *ASCII* tapes, this is a *fourteen*-character value, representing the owner's name and address code.
This entry must be enclosed in parentheses.

Volume Label Image Option Specified

If you specified the volume label image option, the following control statements are required:

- With *EBCDIC* tapes. From one to eight volume label image statements, which are an exact image of the 80-character label. They must be in ascending order (VOL1, VOL2, etc.).
  With *ASCII* tapes. One volume label image statement. This statement is an exact image of the 80-character label. This statement must contain VOL1 in the first four columns.

- `//`␣END statement. This statement is used to separate the volume label image statements for the different tapes. When the program reads this statement, it stops reading statements until it has finished initializing the tape on the current output unit. If eight volume label image statements are supplied for a given EBCDIC tape, you may omit the END statement.

The VOLn label consists of the following fields:

col. 1 – 3 *Label identifier.* Must contain VOL to indicate that it is a volume label.

col. 4 *Volume label number.* Indicates the relative position (1–8) of a volume label within a group of volume labels.

col. 5–10 *Volume serial number.* An identification code assigned to a volume when it enters an installation. Normally a numeric value from 000001 through 999999, although any or all of the six bytes may be alphabetic

col. 11 *Volume security.* Indicates the security status of the volume. Not used by DOS/VS.

col. 12–21 *Data file directory.* The first five bytes contain the starting address of the VTOC, the last five bytes are blank. Used for DASD only.

col. 22–41 Reserved.

col. 42–51 *Owner name and address code.* Indicates a specific customer, installation, and/or system. Used by the VTOC Display program.

col. 52–80 Reserved.

For more information on the VOLn label, see the publication *DOS/VS Data Management Guide,* GC33–5372.

Columns 5 through 10 are used for the volume serial number. It is recommended that you always specify six characters. If you specify less than six characters, the program pads the fields to the right with blanks (X'40'). Always ensure that all six characters are specified in the job control extent information, since the job control program pads an incomplete serial number to the left with zeros. The six-character volume serial number can be specified within apostrophes in the TLBL statement.

Control Statement Stream

Two examples of control statement streams that are used to run the Initialize Tape program are given.

This job stream is used to initialize an ASCII tape without the volume label image option:

```
// JOB INITIAL
// ASSGN SYS000,X'181'
// ASSGN SYS001,UA (no checkpoints)
// EXEC INTTP
// INTT REWIND,A,SERIAL=(000001),P=(1),
   CODE=(AB COMPANY NYC)
/&
```

This job stream is used to initialize an ASCII tape with the volume label image option:

```
// JOB INITIAL
// ASSGN SYS000,X'181'
// ASSGN SYS001,X'182'
// ASSGN SYS002,UA (no checkpoints)
// EXEC INTTP
// INTT CARD,A                    (column 80)
VOL1000001      AB COMPANY NYC         1
// END
VOL1000002      AB COMPANY NYC         1
// END
/&
```

It is assumed that, in each example, SYSLOG is permanently assigned.

CHAPTER 14A
LIST SYSTEM HISTORY (HISTLIST)

Description

Purposes:

- To provide a complete printout of either of the history books Y.PTFSCP and Y.PTFPP, which are generated and maintained by the utility program Maintain System History (PTFHIST), or of any other history book with the same format.
- To provide edited and sorted cross-reference lists of APARs, local fixes, PTFs, and affected library members, with pointers to the entries in the book printout.
- To provide an edited list of lost APARs and an error report.

Storage Requirements

The List System History utility can be executed in any real or virtual partition. The storage requirements depend on the number of entries in the history book to be processed, because the program requires sufficient space for the complete history book and additional working storage. The minimum virtual partition size of 64K is sufficient for processing a book with approximately 1200 entries. If the book to be processed is larger, the partition size will have to be increased accordingly.

Program Output

The List System History utility produces its output in three parts: the book list, the cross-reference lists, and the lost APAR and error report. The program allows you to request specific parts of the output as well as the complete output (see the section *Program Execution).*

Book List

This is a complete listing of the selected history book with its entries in the original sequence. This part of the output is printed as soon as the request-

ed book is found in the library, but it is suppressed if // UPSI 01 or // UPSI 11 was specified.

The following sample shows the format of this list.

```
********* PTF'S AND LOCAL FIXES APPLIED TO DOS/VS REL 32.0                00000000
********* ------------------------------------------------------------- 00000001
********* CUSTOMER---------- EVA CORPORATION                             00000002
********* ADDRESS----------- 2 MAIN STREET, ANYWHERE                     00000003
********* PHONE NO---------- 017-723-977                                 00000004
********* SYSTEM PROGRAMMER-- TOM SMITH                                  00000005
********* ENVIRONMENT--------                                            00000006
********* ------------------------------------------------------------- 00000007
********* TO ADD A NEW ENTRY USE FOLLOWING CONTROL STATEMENTS:           00000008
*********          // EXEC MAINT                                         00000009
*********           UPDATE Y.PTFSCP                                     00000010
*********          ) ADD 0024                                           00000011
*********          NEW ENTRY                                            00000012
*********          ) END                                                00000013
*********          /*                                                   00000014
*********          /&                                                   00000015
*********                                                               00000016
********* ------------------------------------------------------------- 00000017
*********                                                               00000018
********* IDENTIFIERS IN COLUMN ONE ARE AS FOLLOWS:                      00000019
********* P=PTF, A=APAR FIX, L=LOCAL FIX, I=INSTALLATION FIX, C=COMMENT  00000020
********* B=BACKOUT PTF                                                 00000021
*********                                                               00000022
* 0        1                                                         7  00000023
* 3------- 2-------------------------------------------------------------1 00000024
B N10099:1 5745PDA AR=320                                  09/22/76 00000025
B          E10001                                                       00000026
B          R-IJBDMPGN                                                   00000027
P N10084:B 5745BTM AR=310,320 SUP=N10083,N10081,N10080 PRE=N10261,     *00000028
P          N10158                                          09/27/76 00000029
P          E06010 E06061 E06062 E06082 E06087 E06102 E06110 E06114      00000030
P          E06132 E07210 E07211 E07212 E07217 E07218 E07219 E07220      00000031
P          E07221                                                       00000032
P          E.BTMODIH2 E.BTCSE E.BTBTFIX E.DTFBT C-$$ABERP5 C-$$ABERP6   00000033
P          C-$$ABERP7 C-$$ABERP8 C-$$BCTC01 C-$$BOTC02                  00000034
P N10099:1 5745PDA AR=320                                  09/22/76 00000035
P          E10001                                                       00000036
P          R-IJBDMPGN                                                   00000037
P N11190:3 5745UTL AR=330                                  09/22/76 00000038
P          E11313                                                       00000039
P          Z.FASTCOPY                                                   00000040
I          5745SUP    ***   FT TRACE INSTALLED   ***      09/22/76 00000041
I          E.SGDFCH                                                     00000042
A E07220   5745BTM AR=32.0    MSGOP18I                      09/22/76 00000043
A          E.DTFBT                                                      00000044
P N10083:3 5745BTM AR=31.0,32.0 SUP=N10081,N10080 PRE=N10158            00000045
P          E06010 E06061 E06062 E06082 E06087 E06102 E06110 E06114      00000046
P          E06132 E07210 E07211 E07212 E07217 E07218                    00000047
P          E.BTMOD1H2 E.BTCSE E.BTBTFIX E.DTFBT C-$$ABERP5 C-$$ABERP6   00000048
P          C-$$ABERP7 C-$$ABERP8 C-$$BCTC01                             00000049
P N10081:A 5745BTM AR=320                                  09/22/76 00000050
P          *** WARNING *** NO VALID APAR NUMBERS FOUND IN PTF           00000051
P          E.BTNCKID E.BTONLOA E.BTMOD E.BTMODIH2 E.DTFBT R-IJL03Z      00000052
P          C-$$BCTC01 C-$$ABERP9                                        00000053
P N10158:2 5745SUP AR=320                                  09/22/76 00000054
```

```
P           E.IOINTER E.SGEND E.FOPT E.SGCCWT E.SGSUP E.SGMAIN E.ALLOC   00000055
P           E.ALLOCR E.DVCGEN                                            00000056
P           E09050 E09171 E09172 E10123 E10134 E10155 E10159 E10161      00000057
P           E10165 E10170                                                00000058
A E06011    5745BTM       LOOP IN BTBTFIX                  08/13/76 00000059
A           E.DTFBT                                                      00000060
L           5745BTM       CICS ABEND WITH PROGRAM CHECK    08/13/76 00000061
L           BTBTFIX                                                      00000062
```

Cross-Reference Lists

This part of the output is produced when you specify the XREF function. It is also produced for book Y.PTFSCP when the program is executed without console communication.

Fix/PTF Cross-Reference

This is a cross-reference list of all APAR fixes, local fixes, and PTFs installed on the system. The list has four columns with the following meanings:

- FIX
  Installed APAR, local fix, or PTF. APARs and PTFs are identified by APAR and PTF numbers, respectively. A local fix is identified by its entry sequence number.
- COMP
  Component identifier of the APAR fix or PTF.
- LAST INST. DATE
  Last installation date of the APAR fix or PTF.
- ENTRIES
  The sequence numbers of the corresponding history book entries, as an aid to finding these entries in the first part of the program output.

APAR Cross-Reference

This is a sorted list of all installed APARs. The list has four columns with the following meanings:

- APAR
  The numbers of all APARs for which PTFs or APAR fixes are installed.
- COMP
  The component identifier of the affected component.
- LAST FIX
  Either the number of the last PTF containing the APAR or the indication 'A-FIX' for APAR fix.
- ENTRIES
  The sequence numbers of the corresponding history book entries as an aid to finding these entries in the first part of the output.

Module Cross-Reference

A sorted list of all affected library members. The list is produced for the XREF function and has four columns with the following meanings:

- MODULE
  The names of all library members affected by APAR fixes, local fixes, or PTFs. Names of phases are identified by the prefix 'C-'. Names of modules have the prefix 'R-'; for names of books the sublibrary identifier is used (for example, E.PIOCS).
- COMP
  The identifier of the affected component.
- LAST CHNG. DATE
  The date of the last change installed by means of a fix or PTF.
- FIXES
  This column contains one of the following:
  - The PTF number
  - If no PTF is installed, the APAR number of the fix
  - If the fix has no APAR number, the sequence number of the history book entry.

The following sample shows the format of the three cross-reference lists.

```
***   CROSS REFERENCE OF ALL INSTALLED PTF'S   ***

                      LAST
  FIX        COMP     INST.DATE          ENTRY SEQUENCE NUMBERS
--------- --------- ---------       -------------------------------
E06011    5745BTM   08/13/76         0059
E07220    5745BTM   09/22/76         0043

* INST. FIXES W/O APAR NO. *         0041

* LOCAL FIXES W/O APAR NO. *         0061  0062

N10081:A  5745BTM   09/22/76        S0050
N10083:3  5745BTM                   S0045
N10084:B  5745BTM   09/27/76         0028
N10099:1  5745PDA   09/22/76        B0025 B0035
N10158:2  5745SUP   09/22/76         0054
N11190:3  5745UTL   09/22/76         0038

TOTAL FIX COUNT 00010

A 'B' IN FRONT OF A SEQUENCE NUMBER INDICATES A BACKOUT PTF.
A 'S' INDICATES THIS PTF HAS BEEN SUPERSEDED.
```

*** CROSS REFERENCE OF ALL APAR NUMBERS ***

APAR	COMP	LAST FIX	ENTRY SEQUENCE NUMBERS
E06010	5745BTM	N10084:B	0030 S0046
E06011	5745BTM	A-FIX	0059
E06061	5745BTM	N10084:B	0030 S0046
E06062	5745BTM	N10084:B	0030 S0046
E06082	5745BTM	N10084:B	0030 S0046
E06087	5745BTM	N10084:B	0030 S0046
E06102	5745BTM	N10084:B	0030 S0046
E06110	5745BTM	N10084:B	0030 S0046
E06114	5745BTM	N10084:B	0030 S0046
E06132	5745BTM	N10084:B	0031 S0047
E07210	5745BTM	N10084:B	0031 S0047
E07211	5745BTM	N10084:B	0031 S0047
E07212	5745BTM	N10084:B	0031 S0047
E07217	5745BTM	N10084:B	0031 S0047
E07218	5745BTM	N10084:B	0031 S0047
E07219	5745BTM	N10084:B	0031
E07220	5745BTM	N10084:B	0031 0043
E07221	5745BTM	N10084:B	0032
E09050	5745SUP	N10158:2	0057
E09171	5745SUP	N10158:2	0057
E09172	5745SUP	N10158:2	0057
E10001	5745PDA	N10099:1	B0026 B0036
E10123	5745SUP	N10158:2	0057
E10134	5745SUP	N10158:2	0057
E10155	5745SUP	N10158:2	0057
E10159	5745SUP	N10158:2	0057
E10161	5745SUP	N10158:2	0057
E10165	5745SUP	N10158:2	0058
E10170	5745SUP	N10158:2	0058
E11313	5745UTL	N11190:3	0039

TOTAL FIX COUNT 00030

A 'B' IN FRONT OF A SEQUENCE NUMBER INDICATES A BACKOUT PTF.
A 'S' INDICATES THIS PTF HAS BEEN SUPERSEDED.

*** CROSS REFERENCE OF ALL CHANGED MODULES ***

MODULE	COMP	LAST CHNG.DATE	FIXES OR ENTRY SEQUENCE NO'S
C-$$ABERP5	5745BTM	09/27/76	N10084:B N10083:3
C-$$ABERP6	5745BTM	09/27/76	N10084:B N10083:3
C-$$ABERP7	5745BTM	09/27/76	N10084:B N10083:3
C-$$ABERP8	5745BTM	09/27/76	N10084:B N10083:3
C-$$ABERP9	5745BTM	09/22/76	N10081:A
C-$$BCTC01	5745BTM	09/27/76	N10084:B N10083:3 N10081:A
C-$$BOTC02	5745BTM	09/27/76	N10084:B
E.ALLOC	5745SUP	09/22/76	N10158:2
E.ALLOCR	5745SUP	09/22/76	N10158:2
E.BTBTFIX	5745BTM	09/27/76	N10084:B N10083:3
E.BTCSE	5745BTM	09/27/76	N10084:B N10083:3
E.BTMOD	5745BTM	09/22/76	N10081:A
E.BTMODIH2	5745BTM	09/27/76	N10084:B N10083:3 N10081:A
E.BTNCKID	5745BTM	09/22/76	N10081:A

```
E.BTONLOA    5745BTM  09/22/76   N10081:A
E.DTFBT      5745BTM  09/27/76   N10084:B  E07220    N10083:3  N10081:A  E06011
E.DVCGEN     5745SUP  09/22/76   N10158:2
E.FOPT       5745SUP  09/22/76   N10158:2
E.IOINTER    5745SUP  09/22/76   N10158:2
E.SGCCWT     5745SUP  09/22/76   N10158:2
E.SGDFCH     5745SUP  09/22/76   **0042**
E.SGEND      5745SUP  09/22/76   N10158:2
E.SGMAIN     5745SUP  09/22/76   N10158:2
E.SGSUP      5745SUP  09/22/76   N10158:2

R-IJL03Z     5745BTM  09/22/76   N10081:A

Z.FASTCOPY   5745UTL  09/22/76   N11190:3

TOTAL COUNT OF CHANGED MODULES 00026
```

Lost APAR and Error Report

This report provides a summary of all PTFs for which one of the following conditions was detected:

- Incorrect syntax or format,
- Prerequisite PTF(s) missing.

It also provides a list of lost APARs. These are APAR or local fixes which were lost due to the installation of PTFs.

This part of the output is produced when you specifiy the CHECK function. It is also produced for book Y.PTFSCP when the program is executed without console communication.

The following sample shows the format of this report.

```
******   LOST APAR AND ERROR REPORT   ******

                          PHASE       REFERENCED ENTRY OF
  PTF      COMP     DATE   MODULE       LOCAL-FIX OR APAR          DATE   SEQ.NO
-------- ------- -------- ------- ------------------------------ -------- ------
N10084:B 5745BTM 09/27/76         ***THIS PTF HAS NO PREREQUSITE
                                     PTF N10262 INSTALLED                  0028
                           E.DTFBT   A E06011 5745BTM
                                     LOOP IN BTBTFIX            08/13/76   0059
                           E.DTFBT   L        5745BTM CICS ABEND
                                     WITH PROGRAM CHECK         08/13/76   0061
                           E.DTFBT   L        BTBTFIX                      0062
N11190:3 5745UTL 09/22/76         ***THIS PTF IS NOT APPLICABLE
                                     TO THE INSTALLED RELEASE              0038
N10081:A 5745BTM 09/22/76         ***SYNTAX ERROR ON PTF,
                                     ENTRIES BYPASSED                      0050


NOTE: LOCAL AND INSTALLATION FIXES OR ENTRIES WITH PRE APAR
      NO'S WILL BE DISPLAYED IF THE APAR NO. IS MISSING.


  *** SYNTAX ERRORS FOUND ON ENTRIES   0051 0053 0061 0062
```

Program Execution

If you need a complete printout of the history book Y.PTFSCP, you simply invoke the HISTLIST utility program by submitting a // EXEC HISTLIST statement. In this case, the program uses the default book name Y.PTFSCP, lists the contents of the book, and produces the complete output shown in the section *Program Output*.

If the list of the history book contents is not required, // UPSI 01 should be specified. This suppresses the listing of the history book: the remaining output consists of the cross-reference lists and the lost APAR and error report.

If you require a printout of any history book other than Y.PTFSCP, selection must be carried out by means of the operator console. For this purpose, // UPSI 1 must be specified in order to activate console communications.

When the program starts, it issues message 8901D ENTER BOOKNAME OF Y.SUBLIBRARY. You may now specify the required history book, or take the default (Y.PTFSCP) by pressing END/ENTER. The program searches for the requested book and lists its contents.

Printing of the history book contents can again be suppressed by means of the UPSI byte. In this case, // UPSI 11 must be specified.

If the requested book cannot be found, the program issues message 8902I BOOK XXXXXXXX NOT FOUND IN Y.SUBLIBRARY, and re-issues message 8901D. If the book name was simply mistyped, it may be corrected. If the book name was entered correctly, but does not exist, the operator must enter 'END' in order to terminate the job.

If the program finds a sublibrary element with the specified name, but there are no entries, or the contents of the element do not agree with the expected format of a history book (because, for example, the specified element is something other than a history book), the program issues message 8907I NO VALID ENTRIES IN HISTORY BOOK, followed by message 8912I END OF HISTLIST. The program is then canceled.

After the book has been listed, the program issues message 8903D ENTER FUNCTION (XREF, CHECK, END). A response of XREF causes the program to print the three cross-reference lists. A response of CHECK results in a printout of the lost APAR and error report. A response of END terminates the job. Depression of END/ENTER in response to this message causes the default function XREF to be executed.

After each execution of the XREF or CHECK function, message 8903D is re-issued to permit selection of another function or termination of the job.

Job Control Statements

The HISTLIST utility runs in a batch partition of an operational DOS/VS system and is controlled by the following job control statements:

`// JOB jobname`	Required.
`// UPSI`	
`1`	Optional. Indicates that console communication is desired.
`01`	Optional. Suppresses listing of the history book.
`// ASSGN SYSSLB,X'cuu'`	Required if the requested book is stored in a private source statement library.
`// EXEC HISTLIST`	Required.
	Note: It is assumed that SYSLOG, SYSLST, and SYSRES are already asigned.

CHAPTER 15
MAINTAIN SYSTEM HISTORY (PTFHIST)

Description

Histories of installed Program Temporary Fixes (PTFs) are maintained in the system. The Maintain System History utility is designed to simplify this maintenance, and performs the following tasks:

- Selects specified PTFs from a PTF file.
- Generates job control statements to punch a backout PTF.

  A backout PTF consists of control statements that can be used to remove the PTF at a later time, if this should be necessary, and to restore the libraries to their pre-PTF condition.
- Generates job control statements to update the system history.
- Lists the PTF index of a master file or the job control statements within a PTF file.

The program can also be used for the installation of some Independent Distributed Components (IRs) and Program Products (PPs). The description of the control statements in this chapter includes the parameters needed to process such IRs and PPs. However, for information on how to use the program for IR and PP installation, refer to the individual IR or PP.

The System History

The system history is kept in the form of two books in the source statement library, namely:

a) the system control program history Y.PTFSCP
b) the program product history Y.PTFPP

These history books are provided in skeleton format. An update run to insert customer information is required at installation time. This initialization job is generated with the aid of the macro HIST, and it can be used to initialize both books simultaneously. The macro HIST and its use are described in *DOS/VS System Generation,* GC33-5377.

During each PTF application the SCP or PP history is updated using information retrieved from the JCL comment statements and/or from the

catalog control statements. The names of the affected modules are retrieved from the catalog statements, and the numbers of the fixed APARs from the JCL comment statements.

Executing the Maintain System History Program

Operation of the Maintain System History utility is controlled with the aid of the utility modifier (UDS) statement. The basic format of this statement is:

```
// UDS {SEL|LST}
```

where SEL indicates that the select function of the utility is required, and LST requests the list function. The following section is accordingly divided into two parts that describe the operation of these two functions and provide information about further parameters needed for the execution of the utility.

The Select Function (SEL)

The control statements used by the select function are the utility modifier (UDS) statement, the PTF select statement, and the comment statement.

Selection is controlled by PTF select statements. These may be in any order, but they must follow the utility modifier statement. Up to 100 select statements may follow the utility modifier statement.

Each PTF is identified by a PTF number. This number is present as jobname in the JOB statement of the PTF. The JOB statement is used as the select argument.

In order to simplify the selection of PTFs, the select argument is considered to consist of a fixed part and a variable part. The fixed part of the select argument is common to all select arguments within a PTF file; it may be specified once in the utility modifier statement instead of in each select statement. The variable part of the select argument is the unique identifier for each PTF within the PTF file; it has to be specified in the respective select statement.

The following table summarizes the effect of submitting or not submitting the utility modifier statement (UDS) in combination with PTF select statement(s).

// UDS SEL,... submitted	select statement(s) submitted	will cause
YES	YES	selected PTFs to be processed.
NO	NO	all PTFs on input file to be processed.
YES	NO	no PTFs to be processed.
NO	YES	error message to be issued. Invalid.

Comment statements may be submitted for each selected PTF, after the select statements, and will be inserted into the history entry for that PTF. A maximum of 20 comments may be submitted for one PTF, and the limit for one selection run is 100 comment statements. Comments can be submitted even if no utility modifier statement has been supplied.

Utility Modifier Statement

Contains information required to run the program. The parameters are positional. The first parameter (SEL) must be supplied.

The format and entries are:

```
// UDS SEL,[(ww,'x...x',yy,zz)]
         SINGLE|MASS|MASSB|SUM [,name]
```

The specification of a utility modifier statement without one or more parameters (// UDS SEL) results in the generation of the standard default:

```
// UDS SEL,(1,'// JOB N',9,13),SINGLE
```

which is suitable for the selection of DOS/VS SCP PTFs for system components, which are in the form:

```
// JOB Nnnnnn
```

where nnnnn is the PTF identifier which must be specified in the select statement.

// U	Utility modifier statement entry.
DS	Indicates the Maintain System History program. Can be omitted.
SEL	Requests the select-PTF function.
ww,'x...x'	These two operands describe the fixed part of the select argument.

ww specifies the starting position of the fixed part of the argument within the statement in the PTF file.

x...x is the character representation of the fixed part of the argument. The length of the fixed part may be up to twenty characters. Any apostrophes within the character string must be coded as double apostrophes.

`yy,zz`

These two operands specify the starting and ending positions of the variable part of the argument within the statement in the PTF file. The contents of the field delimited by these positions are compared with the parameter n...n in the select statements. The length of the variable part may be up to eight characters.

`SINGLE`
`MASS`
`MASSB`
`SUM`

This parameter controls backout PTF creation and history updating.

If SINGLE is specified, statements for a backout PTF are generated for each selected PTF. The history is updated at the end of each PTF job.

In the case of mass application of PTFs, or the application of tested PTFs, backout PTFs are often unnecessary. The source statement library may also be overloaded by the various update steps for the history.

If MASS is specified, creation of backout PTFs is suppressed and the history entries are collected in a table. A history update job is generated at the end of the selection run or between two PTFs if the table becomes full.

If MASSB is specified, job control statements for a backout PTF are generated for each selected PTF. A history update job is generated at the end of the selection run or between two PTFs if the table becomes full.

SUM is used only for IR and PP installation.

`name`

Used only for IR and PP installation.

Select Statement

This statement has the following format:

```
n...n[,C=kkkkkkk][,CL=S|P][,RL=S|P][,SL=S|P]
```

The first parameter must be supplied. The parameters are not positional except for the first one.

`n...n` — Represents the variable part of the select argument for the PTF to be selected; it may be up to eight characters long. The length of this parameter must match the length specified by the parameters yy and zz in the utility modifier statement.

`C=kkkkkkk` — Specifies the component which is affected by this PTF. This parameter is optional, but if specified is transferred to the history. Its length must be seven bytes (for example, 5745UTL for component 5745-SC-UTL). A parameter with an incorrect length is ignored.

`CL=S|P`
`RL=S|P`
`SL=S|P`

Used only for IR and PP installation.

Comment Statement

The comment statement is read in from SYSIPT and merged with the selected PTF. It is treated as a JCL comment within the PTF jobstream. A comment may also appear within the PTF itself. A maximum of 20 comment statements can be processed for one PTF.

The comment statement has the following format:

```
* 'n...n' [tt...tt]
```

`*` — An asterisk in column 1 followed by at least one blank identifies a comment statement.

`'n...n'` — Identifies the PTF into which the comment is to be inserted. This parameter may be up to eight characters long and must be the complete identifier of the required PTF.

`tt...tt` — Up to 60 text characters, separated from the preceding parameter by one or more blanks.

Control Statement Examples

```
// UDS SEL,(1,'// JOB ',8,13),MASS
N04066
N04039,C=5745LBR
N04069,C=5745PWR
* 'N04069' APAR E05058 NOT FIXED
```

The utility modifier statement requests the select function. The fixed part of the select argument starts at column 1, is seven bytes long, and contains '// JOB'. The variable part of the argument extends from column 8 to 13.

PTFs N04066, N04039, and N04069 are requested. The second and third select statements also contain the component ID.

The comment supplied for PTF N04069 will appear in the history entry.

```
// UDS SEL
04577
* 'N04577' LOCAL FIX 21 IS ALSO INSTALLED
```

This utility modifier statement defaults to the 'standard' version:

```
// UDS SEL,(1,'// JOB N',9,13),SINGLE
```

As the fixed part of the select argument is now '// JOB N', the N in the PTF ID need not be specified in the select statement.

The comment statement, however, needs the complete ID of the PTF.

Job Control Statements

// JOB	Required.
// ASSGN SYSIPT	Required. Input device for file 'IJSYSIN', which contains: Utility modifier statement, PTF select statements, comment statements.
// ASSGN SYS004	Optional. Input unit for the master PTF file for normal PTF processing. The unit can be a card reader, tape drive, disk drive, or diskette unit.
// ASSGN SYSSLB	Used only for IR and PP installation.
// ASSGN SYS001	Optional. Workfile (filename: IJSYS01). This file is needed only if backout PTFs are to be generated. Supported devices: tape or disk.
// ASSGN SYSLOG	Required. Accounting information input and message output device.
// ASSGN SYSPCH	Required. Output jobstream (filename: IJSYSPH).

`// ASSGN SYSLST`	Required. Printout of control statements, selected PTFs, and statistics.

`// DLBL IJSYSIN,'file-id',...`
`// EXTENT SYSIPT,...`
`// DLBL PTF,'file-id',...`
`// EXTENT SYS004,...`
`// DLBL IJSYSSL,'file-id',...`
`// EXTENT SYSSLB,...`
`// DLBL IJSYS01,'file-id',...`
`// EXTENT SYS001,...`
	These statements must be used if the devices used for SYSIPT, SYS004, SYSSLB, or SYS001 are disk drives (2314, 3330, 3340, or 3350) or, for SYSIPT or SYS004, a diskette I/O unit.

`// EXEC PTFHIST`	Required. The program name is PTFHIST.
`SIZE=nnk`	Optional. The minimum size is 64K.

Notes: If the input is on a multi-file tape, the tape must first be positioned to the correct file. The MTC command may be used for this.

If the input is the form of a card deck, selection of specific PTFs is not possible. SYSIPT has to be assigned UA in that case.

User job accounting information is requested via SYSLOG at the start of the program. The user may enter information of up to 56 bytes. This information will be transferred to the JOB statement(s), starting at column 17.

The system input is printed on SYSLST as it is read. During selection, the PTF ID is printed together with its card count (or 'NOT FOUND' if the PTF could not be found). A summary at the end of the program shows how many PTFs were selected.

To conclude processing, SYSPCH must be closed. The output on SYSPCH is in a form suitable for input via SYSIN for application of the PTFs to the system.

Sample Select Job

```
// JOB SELECT
// ASSGN SYS004,TAPE
// ASSGN SYSPCH,TAPE
// ASSGN SYS001,DISK,VOL=111111
// DLBL IJSYS01,...
// EXTENT SYS001,...
// EXEC PTFHIST
// UDS SEL
04054,C=5745RMS
04076
/*
// CLOSE SYSPCH,...
/&
```

The resulting output on SYSPCH is:

```
// JOB N04054#3
// OPTION NOLOG
// EXEC PTFREPRO (see note)
* // JOB N04054#3
* * C A U T I O N THIS BACKOUT-PTF REMOVES THE APARS LISTED BELOW
      .
      .
    JCL for BACKOUT-PTF generation
      .
* * * N04054#3 * * *
```

```
* APPLICABLE REL ...
      .
      .
      .
     PTF fetched from input tape
      .
      .
RLD
END
/*
// EXEC LNKEDT
// ASSGN SYSSLB,UA
// EXEC MAINT
   UPDATE Y.PTFSCP
      .
      .
   Control statements for history update step
      .
      .
) END
/*
/&
```

Note: The auxiliary program PTFREPRO transfers the job control statements for the backout PTF to SYSPCH. The program PTFHIST itself generates the calls for PTFREPRO in the correct positions in the job stream. No user intervention is necessary for the execution of PTFREPRO.

The resulting output on SYSLST is:

First page

```
// JOB SELECT
// ASSGN SYS004,TAPE
// ASSGN SYSPCH,TAPE
// ASSGN SYS001,DISK,VOL=111111
// DLBL IJSYS01,...
// EXTENT SYS001,...
// EXEC PTFHIST
```

Second page

```
*** PTFHIST *** CONTROL-STMTS, SELECT-ARG. AND COMMENTS SUPPLIED
INCOMPLETE SELECT CONTROL STMT DEFAULTS TO:
// UDS SEL,(1,'// JOB N',9,13),SINGLE

// UDS SEL
04054,C=5745RMS
04076

PTFS SELECTED          CARD-CNT
N04054                 146
N04076     NOT FOUND

** SELECTED FROM INPUT FILE:

**       1  PTFS CONTAINING
**       3  PHASES
**          MODULES
**          MACROS
**          PROCEDURES
**          INVALID NAMES
```

The List Function (LST)

The list function permits the user to print a PTF file index or all the job control statements contained in a PTF file.

The operation of the list function is controlled with the aid of the utility modifier (UDS) statement, as described below.

Utility Modifier Statement

Contains information to run the program. The parameters are positional. The first parameter must be supplied.

The format and entries are:

```
// UDS LST,{INDEX|JCL}[,name]
```

If you specify only

```
// U LST
```

the resulting default is

```
// UDS LST,INDEX
```

`// U`	Utility modifier statement entry.
`DS`	Indicates the Maintain System History program. Can be omitted.
`LST`	Requests the list function.
`INDEX` `JCL`	Specifies whether the index of a PTF the JCL contained in a PTF file is to be listed.
`name`	Used only for IR and PP installation.

The two list functions are described separately below.

List-Index Function

The list-index function is requested with the statement:

```
// UDS LST,INDEX[,name]
```

The utility modifier statement is read from SYSIPT, and the PTF file is read from SYS004, which may be a disk, a tape, or a diskette, or from SYSSLB in the case of IR and PP installation. The input record length may be 80 or 81 bytes, or the records may be blocked into 3440-byte blocks.

The output of the function is printed on SYSLST, and contains the following information:

- the utility modifier statement read.

- the PTF names, their card counts, and the word BACKOUT if this is a backout PTF (backout PTFs produced by previous PTF application runs may exist in the user environment).

List-JCL Function

A list of all job control statements is requested with the statement:

```
// UDS LST,JCL[,name]
```

A list of all job control statements found in the file (including CATALR, CATALS, CATALP, INCLUDE, PHASE, COPYC, COPYR, and COPYS statements) is printed on SYSLST.

I/O Device Assignments

The following I/O device assignments are required for the list function of the Maintain System History utility:

SYSIPT	Control statement input
SYSLST	System output
SYS004	PTF input file (The filename is PTF.)
or	
SYSSLB	IR or PP installation history input.

Sample List-Index Job

```
// JOB PRINT PTF-INDEX
// ASSGN SYS004,X'281'   PTF-TAPE
// EXEC PTFHIST
// UDS LST,INDEX
/*
/&
```

The resulting output from the above job is:

First page

```
// JOB PRINT PTF-INDEX
// ASSGN SYS004,X'281'
// EXEC PTFHIST
```

Second page

```
*** PTFHIST *** CONTROL-STMTS, SELECT-ARG. AND COMMENTS SUPPLIED

// UDS LST,INDEX


INDEX TO PTF-FILE    CARD-CNT     21/10/75

  N04007#3             215
  N04009#3             338
  N04012#3             165
  N04014#3              34
     .                   .
     .                   .
     .                   .
```

Sample List-JCL Job

```
// JOB PRINT JOB-CONTROL-STMTS
// ASSGN SYS004,DISK,VOL=DOSNNN
// DLBL PTF,...
// EXTENT SYS004,DOSNNN,...
// EXEC PTFHIST
// UDS LST,JCL
/*
/&
```

The resulting output from this job is:

First page

```
// JOB PRINT JOB-CONTROL-STMTS
// ASSGN SYS004,DISK,VOL=DOSNNN
// DLBL PTF,...
// EXTENT SYS004,DOSNNN
// EXEC PTFHIST
```

Second page

```
*** PTFHIST *** CONTROL-STMTS, SELECT-ARG. AND COMMENTS SUPPLIED

// UDS LST,JCL




// JOB N04007#3
* APPLICABLE REL ...
* APARS FIXED E05100,E05118,E05133
* MOD/MAC AFFECTED IPW$$RR
* COMMENT REPLACED PHASES IPW$$LR,IPW$$PR,IPW$$SC
* COMMENT UNASSIGN SYSCLB IF POWER/VS PHASES MUST BE ON CIL
// OPTION CATAL
// PAUSE END/ENTER OR CANCEL
 INCLUDE
     PHASE IPW$$PR,+0,NOAUTO
           INCLUDE,(PRCS)
     PHASE IPW$$LR,+0,NOAUTO
           INCLUDE,(LRCS)
     PHASE IPW$$SC,+0,NOAUTO
           INCLUDE,(SCCS)
/*
// EXEC LNKEDT
/&
// JOB N04009#3
     .
     .
     .
```

Updating the System History

Each PTF fetched from the master file is followed by JCL statements which update the appropriate history book.

Figure 15.1 shows an example of the SCP history after application of two PTFs and subsequent removal of the first one. Only columns 1-71 of the history are shown; columns 77-80 contain sequence numbers. The example assumes that the history header was initialized at installation time.

```
        BKEND   Y.PTFSCP

********* PTFS AND LOCAL FIXES APPLIED TO DOS/VS REL NN.N
********* ------------------------------------------------------------
********* CUSTOMER-----------EVA CORPORATION
********* ADDRESS------------2 MAIN STREET, ANYWHERE
********* PHONE NO-----------017-723-977
********* SYSTEM PROGRAMMER--TOM SMITH
********* ENVIRONMENT--------
********* ------------------------------------------------------------
********* TO ADD A NEW ENTRY USE FOLLOWING CONTROL STATEMENTS:
*********           // EXEC MAINT
*********           UPDATE Y.PTFSCP
*********           ) ADD 0024
*********           NEW ENTRY
*********           ) END
*********           /*
*********           /&
********* ------------------------------------------------------------
*********
********* IDENTIFIERS IN COLUMN ONE ARE AS FOLLOWS:
********* P=PTF, A=APAR FIX, L=LOCAL FIX, I=INSTALLATION FIX, C=COMMENT
********* B=BACKOUT PTF
*********
*********
* 0         1                                                  7
* 3-------- 2-------------------------------------------------1
B N04074    5745TAP                                  21/01/76
B           E05335 E05908
B           E.DTFMT
P N04089    5745IOX AR=NNN                           21/01/76
P           E04348 E04906 E05372 E05630 E05786
P           C-$$BOPEN C-$$BOPEN0 C-$$BOPEN1
P           C-$$BOPEN3 C-$$BOPIGN C-$$BOSD01
P N04074    5745TAP                                  19/01/76
P           E05335 E05908
P           E.DTFMT
       BKEND
```

Figure 15.1 **Sample of the SCP history**

The date in the history book records, which reflects the date on which a PTF was selected, is extracted from the Partition Communication Region. It is entered in the records in the format MM/DD/YY or DD/MM/YY, depending on the DATE parameter in the supervisor generation macro STDJC.

PTFs N04074 and N04089 were selected from a PTF file by the Maintain System History utility and applied to the system. During this process, backout PTFs were generated and these may be used later to remove a PTF, if it is unsuitable or no longer required. The APAR numbers shown in the history were retrieved from the comment statement included in the JCL or the PTF. The names of the macros and modules affected were retrieved from the PHASE statement or the CATAL statement.

As PTF N04074 caused other problems, it had to be removed. To do this, backout PTF N04074 was selected from the backout PTF file and applied to the system. The application of a backout PTF is indicated by a warning message on the system console, and by a B in column 1 of the history entry.

Each new entry in the history is inserted at the top of the history book, and the previous entries are shifted downwards. This means that the application sequence is from bottom to top.

CHAPTER 16
PRINT HARDCOPY FILE (PRINTLOG)

Description

Purpose:

- To print on SYSLST the harcopy file IJSYSCN or selectively print groups of messages contained in it.

For central processing units that are equipped with a display operator console, each line that appears on the screen is written to the hardcopy file IJSYSCN. This recorder file resides on SYSREC and is defined as a disk extent. The procedure for creating the hardcopy file is described in *DOS/VS Operating Procedures,* GC33-5378.

Apart from printing the entire hardcopy file, the PRINTLOG program allows you to select specific groups of messages to be printed. These groups are the message types, messages pertaining to a particular job or issued on a particular day. You may select and print these groups either for one partition or for all partitions. Or you may print the messages issued by the attention routine. Moreover, you can specify whether you want the messages, or message groups to be selected from the entire hardcopy file or just from that part that accumulated since the last printing of all messages.

Options

The utility modifier options are entered via the display operator console (see *Executing the PRINTLOG Program).* The available options are:

```
[ALL/NEW] [AR/BG/Fn] [A] [D] [I] [E] [U]

[JOBNAME=name] [mm/dd/yy]
```

The options may be specified in any order. If multiple options are used, each option must be separated from the next by a comma.

ALL	Print messages contained in the hardcopy file. If you specify no further option, all messages contained in the hardcopy file will be printed.
NEW	Print messages that have accumulated since the last time PRINTLOG was executed with ALL or NEW as the only option specified. If you specifiy no further option, all messages accumulated will be printed. ALL is the default.
AR	Print messages issued by the attention routine.
BG Fn	Print messages pertaining to the one specified partition. n in Fn can be from 1 up to number of foreground partitions supported in the system. The operator may enter only one of the options AR, BG, Fn. If more than one is entered, only the first is processed; the other options are ignored and *not* redisplayed as erroneous options.
	The operator may enter any combination of the following options. They must be separated by a comma.
A	Print action messages.
D	Print decision messages.
I	Print information messages.
E	Print eventual-action messages.
U	Print undefined messages.
JOBNAME=name	Print messages pertaining to the job identified by name. The operator must substitute the name of the job (from the // JOB statement).
mm/dd/yy	Print messages issued on a particular day.

The date should be entered in the format month, day, year. Leading zeros may be omitted.

If the operator omits the first Y of YY, the program inserts the proper digit as contained in the system date in the communications area.

The program also checks whether the date entry contains any nonnumeric characters. If the operator, in correcting a previous date entry, enters a date that differs from the first entry, the corrected (second) entry is processed.

Executing the PRINTLOG Program

The PRINTLOG program is selfrelocating and may be run in any partition. The job control statements to execute the program are:

```
// JOB
// EXEC PRINTLOG
```

You initiate the PRINTLOG program by pressing ENTER. The program issues a message, which displays all options and prompts you to specify one or more of the available options. The program performs syntax checking of the options entered. If you enter an option that does not exist, or if you do not adhere to the prescribed format, the incorrect options are displayed and you can correct them.

If a printout of all messages is required, you can bypass the syntax-checking function by simply pressing ENTER instead of entering the option ALL.

CHAPTER 17
VTOC DISPLAY

Description

Purpose:

- To display the file labels contained in the VTOC of a disk pack or data cell.

The VTOC Display utility helps you to keep track of the files and their extents. The program lists the file labels in the VTOC plus certain information from the standard volume label (VOL1), namely the label identifier, the starting address of the VTOC, the volume serial number, and the owner name and address code. The file labels are identified by their location within the VTOC and by their format types; all major fields are displayed together with appropriate headings.

The output of the VTOC Display program is on SYS005, which can be assigned to a printer, tape file, or disk pack.

The first time a label of a data secured file is encountered, message 8V96D

```
FORMAT 1 LABEL OF DATA SECURED FILE
```

is issued to the operator. Reply YES if you want the label information of all data secured files included in the output listing. A data secured file is identified in the listing by the letters DSF, immediately following the format identification. If you reply NO, the label information for data secured files is not printed.

Control Statement Stream

There is no utility modifier statement for the VTOC Display program.

An example of a control statement stream that is used to run the VTOC Display program:

```
// JOB VTOC
// ASSGN SYS004,X'191'
// ASSGN SYS005,X'00E'
// PAUSE REPLY NO IF MSG 8V96D IS ISSUED
// EXEC LVTOC
/&
```

The // PAUSE statement shown in the example indicates to the operator that label information for data secured files should not be included in the printed output.

When SYS005 is assigned to a disk, DLBL and EXTENT statements are required. When SYS005 is assigned to a tape, a TLBL statement is required.

Glossary

The explanations of the following terms relate to their use in IBM DOS/VS. These explanations may differ from those in other publications.

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the American National Standard Vocabulary for Information Processing (Copyright © 1970 by American National Standards Institute, Incorporated), which was prepared by Subcommittee X3K5 on Terminology and Glossary of American National Standards Committee X3. ANSI definitions are preceded by an asterisk.

A

Access method — A technique for moving data between virtual storage and input/output devices.

Address constant — A value, or an expression representing a value, used in the calculation of storage addresses.

Alternate track — One of a number of tracks set aside on a disk pack or data cell for use as alternatives to any defective tracks found elsewhere on the disk pack or data cell.

APAR — Authorized program analysis report. A request for correction of a problem caused by a defect in a current unaltered release of a program. A PTF or corrected code is issued to the customer and the correction is incorporated into subsequent releases of the program.

B

Background partition — A partition to which a background job is assigned.

Batched job

- A job that is grouped with other jobs as input to a computing system.
- A job whose job control statements are grouped with job control statements of other jobs as input to a computing system.

Blocking — To group logical records physically for the purpose of saving storage space in external storage, or increasing the efficiency of access or processing.

Book — A group of source statements written in any of the languages supported by DOS/VS and stored in a source statement library.

C

Checkpoint — A point in a program at which sufficient information can be stored to permit restarting the job step from that point.

Checkpoint record — Records that contain the status of the job and the system at the time the records are written by the checkpoint routine. These records provide the information necessary for restarting a job without having to return to the beginning of the job.

Checkpoint/restart facility — A facility for restarting execution of a program at some point other than at the beginning, after the program was terminated due to a program or system failure. A restart may begin at a checkpoint or from the beginning of a job step, and uses checkpoint records to reinitialize the system.

Core image library — A library of phases that have been produced as output from link-editing. The phases in the core image library are in a format that is executable either directly or after processing by the relocating loader in the supervisor.

D

Data file — A collection of related data records organized in a specific manner. For example, a payroll file (one record for each employee, showing his rate of pay, deductions, etc.), or an inventory file (one record for each inventory item, showing the cost, selling price, number in stock, etc.). See also *File.*

Deblocking — The action of making the first and each subsequent logical record of a block available for processing one record at a time.

Default — The choice among alternatives made by the system when no explicit choice is specified by the user.

Deleted record — Deleted records on an IBM 3540 Diskette Input/Output Unit are flagged 'special record' in the ID-portion of that record. They are marked X'C4' ('D'), in the first byte of the data portion of the deleted record. This situation occurs when deleting a record from an existing file.

Direct access

- Retrieval or storage of data by a reference to its location on a volume, other than relative to the previously retrieved or stored data.
- * Pertaining to the process of obtaining data from, or placing data into, storage where the time required for such access is independent of the location of the data most recently obtained or placed in storage.
- * Pertaining to a storage device in which the access time is effectively independent of the location of the data.

***Dump**

- To copy the contents of all or part of virtual storage.
- The data resulting from the above-mentioned process.

E

Ellipsis — A series of three periods, indicating that a variable number of items may be included.

Emulator

Emulator — The combination of programming techniques and special machine features that permits a given computing system to execute programs written for another system.

***Error message** — An indication that an error has been detected.

Executive routine — A routine that controls the execution of other routines.

Extent — A continuous space on a direct access storage device, occupied by or reserved for a particular file.

F

***File** — A collection of related records treated as a unit.

Fixed-length record — A record having the same length as all other records with which it is logically or physically related. Contrasted with *variable-length records.*

Foreground partition — A partition to which a foreground job is assigned.

H

Hard copy — A printed copy of machine output in a visually readable form, for example, printed reports, listings, documents, and summaries.

Home address — An address written on a direct access volume, denoting a track's address relative to the beginning of the volume.

I

Indexed-sequential organization — The records of an indexed-sequential file are arranged in logical sequence by key. Indexes to these keys permit direct access to individual records. All or part of the file can be processed sequentially.

***Initial program loader (IPL)** — The procedure that causes the initial part of an operating system to be loaded such that the program can then proceed under its own control.

I/O (Input/Output) area — A portion of virtual storage into which data is read or from which data is written.

J

Job control — A program that is called into storage to prepare each job step to be run. Some of its functions are to assign I/O devices to certain symbolic names, to set switches for program use, to log (or print) job control statements, and to fetch the first program phase of each job step.

***Job control statement** — A statement in a job that is used in identifying the job or describing its requirements to the operating system.

JOB statement — The control statement that identifies the beginning of a job. It contains the name of the job.

L

Label — Identification record for a tape or disk file.

Library — A collection of files or programs, each element of which has a unique name, that are related by some common characteristic. For example, all phases in the core image library have been processed by the linkage editor.

Linkage editor — A processing program that prepares the output of language translators for execution. It combines separately produced object or load modules; resolves symbolic cross references among them, and generates overlay structures on request; and produces executable code (a load module).

M

***Module** — A program unit that is descrete and identifiable with respect to compiling, combining with other units, and loading, for example, the input to, or output from, an assembler, compiler, linkage editor, or executive routine.

Multiprogramming system — A system that controls more than one program simultaneously by interleaving their execution.

O

Operator message — A message from the operating system or a problem program directing the operator to perform a specific function, such as mounting a tape reel, or informing him of specific conditions within the system, such as an error condition.

Overlap — To do something at the same time that something else is being done. For example, to perform I/O operations while instructions are being executed by the central processing unit.

P

***Parameter** — A variable that is given a constant value for a specific purpose or process.

Phase — The smallest complete unit that can be referenced in the core image library.

Private library — A user-owned library that is separate and distinct from the system library.

PTF — Program temporary fix. A temporary solution or bypass of a problem diagnosed by IBM field engineering as the result of a defect in a current unaltered release of the program.

R

Real storage — The storage of a System/370 computing system from which the central processing unit can directly obtain instructions and data, and to which it can directly return results.

Record zero — The first record on every track. Record zero (R0) is used primarily to facilitate the use of an alternate track, when the original track is found to be defective.

Relocatable library — A library of relocatable object modules required by various computers. It allows the user to keep frequently used modules available for combination with other modules without recompilation.

Restart — See *Checkpoint/restart facility.*

S

Selfrelocating — A programmed routine that is loaded at any doubleword boundary and can adjust its address values so as to be executed at that location.

Sequential organization — Records of a sequential file are arranged in the order in which they will be processed.

Sequentially relocated record — Sequentially relocated records on an IBM 3540 Diskette Input/Output Unit are records flagged 'special record', in the ID-portion, of that record. They are marked X'C6' ('F') in the first byte of the data portion of the sequentially relocated record. This situation occurs when a record of a file (at creation time) cannot be written due to a 'bad spot' on a diskette (gives a permanent I/O error).

Special record — Special records on an IBM 3540 Diskette Input/Output Unit are records that are flagged in the ID-field preceding each record. When a record is flagged 'special', the first byte of the data portion gives information on the nature of the record (see *Deleted record* and *Sequentially relocated record*).

Stand-alone dump — A program that displays the contents of the registers and part or all of virtual storage and that runs independently and is not controlled by DOS/VS.

Standard label — A fixed-format identification record for a tape or disk file. Standard labels can be written and processed by DOS/VS.

Supervisor — A component of the control program. It consists of routines to control the functions of program loading, machine interruptions, external interruptions, operator communications, and physical IOCS requests and interruptions. The supervisor alone operates in the privileged (supervisor) state. It coexists in real storage with problem programs.

Symbolic I/O assignment — A means by which problem programs can refer to an I/O device by a symbolic name. Before a program is executed, job control can be used to assign a specific I/O device to that symbolic name.

T

***Track** — The portion of a storage medium, such as a tape, or disk, that is accessible to a given reading head position.

Track descriptor record (R0) See *Record zero.*

Transient area A virtual storage area used for temporary storage of transient routines, such as nonresident supervisor call or error-handling routines.

Transient routine Selfrelocating routines, permanently stored on the system residence device and loaded by the supervisor into the transient area when needed for execution.

U

Unrecoverable error A hardware error which cannot be recovered from by the normal retry procedures.

User label An identification record for a tape or disk file; the format and contents are defined by the user, who must also write the necessary processing routines.

Utility program A problem program designed to perform a routine task, such as transcribing data from one storage device to another.

V

Variable-length record A record having a length independent of the length of other records with which it is logically or physically associated. Contrasted with *Fixed-length records*

Virtual storage Addressable space that appears to the user as real storage, from which instructions and data are mapped into real storage locations. The size of virutal storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available, rather than by the actual number of real storage locations.

Volume

- That portion of a single unit of storage media which is accessible to a single read/write mechanism, for example, a disk pack, or part of a disk storage module.
- A recording medium that is mounted and dismounted as a unit, for example, a reel of magnetic tape, a disk pack, a data cell.

Volume table of contents A table on a direct access volume that describes each file on the volume. Abbreviated VTOC.

Index

D

K

L

M

N

Q

R

S

T

DOS/VS System Utilities Printed in U.S.A. GC33-5381-2

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

Technical Newsletter

This Newsletter No. GN33-9211
Date April 29, 1977

Base Publication No. GC33-5381-2
File No. S370-32

Previous Newsletters None

DOS/VS System Utilities

This Technical Newsletter, a part of Release 34 of the IBM Disk Operating System/Virtual Storage, DOS/VS, provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent DOS/VS releases unless specifically altered. Pages to be inserted and/or removed are:

Cover, Summary of Amendments	10.01, 10.02
VII - XII	10.05, 10.06
1.01 - 1.08	11.01 - 11.14
2.01, 2.02	13.01, 13.02
3.01 - 3.06	13.05 - 13.10
4.05 - 4.10	14.03, 14.04
6.01 - 6.04	14A.01 - 14A.08 (added)
7.01, 7.02	15.01 - 15.14
7.05, 7.06	16.01 - 16.04
8.07, 8.08	X.01, X.02
9.17 - 9.20	X.07 - Back page

A change to the text or to an illustration is indicated by a vertical line to the left of the change.

Summary of Amendments

This Technical Newsletter documents:

- List System History (HISTLIST) utility
- Full support of
  - IBM 3350 Direct Access Storage
  - IBM 3330-11
- Support of IBM 3277 Display Station as operator console
- Support of IBM 3540 Diskette Unit as IPL communication device.

In addition, the Technical Newsletter contains minor technical corrections.

Note: *Please file this cover letter at the back of the manual to provide a record of changes.*

IBM Laboratory, Programming Publications Department, Boeblingen, Germany

Printed in U.S.A.

IBM **Technical Newsletter**

This Newsletter No.	GN33-9245
Date	November 28, 19
Base Publication No.	GC33-5381-2
File No.	S370-32
Prerequisite Newsletters	GN33-9211

DOS/VS System Utilities

This Technical Newsletter, a part of the independent component release (ICR) of support for the IBM 3800 Printing Subsystem under Release 34 of the IBM Disk Operating System/Virtual Storage, DOS/VS, provides replacement pages for your publication. Information contained on these pages applies only if the ICR is installed on your system. You need not insert the pages if it is not installed. These replacement pages remain in effect for subsequent DOS/VS releases unless specifically altered. Pages to be replaced are:

Cover, edition notice
v
1.01-1.04
11.11, 11.12
X.09, X.10

A technical change to the text or to an illustration is indicated by a vertical line to the left of the change.

Summary of Amendments

Changes to the system are summarized under "Summary of Amendments" on the back of the front cover

For a complete list of publications that support the DOS/VS IBM 3800 Printing Subsystem ICR, see the *DOS/VS IBM 3800 Printing Subsystem Programmer's Guide*, GC26-3900.

Note: Please insert this page in your publication to provide a record of changes.

IBM Laboratory, Programming Publications Department, Boeblingen, Germany

Printed i

DOS/VS
System Utilities

GC33-5381-2

READER'S COMMENT FORM

This sheet is for comments and suggestions about this manual. We would appreciate *your* views, favorable or unfavorable, in order to aid us in improving *this* publication. This form will be sent directly to the author's department. Please include your name and address if you wish a reply. Contact your IBM branch office for answers to technical questions about the system or when requesting additional publications. Thank you.

Name

Address

What is your occupation?

How did you use this manual?

As a reference source

As a classroom text

As a self-study text

Your comments* and suggestions:

*** We would especially appreciate your comments on any of the following topics:**

Clarity of the text	Accuracy	Index	Illustrations	Appearance	Paper
Organization of the text	Cross-references	Tables	Examples	Printing	Binding

YOUR COMMENTS, PLEASE . . .

This manual is part of a library that serves as a reference source for systems analyst , programmers and operators of IBM systems. Your answers to the questions on th. back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Fold Fold

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM Corporation
1133 Westchester Avenue
White Plains, N.Y. 10604

Attention: Department 813 BP

Fold Fold

IBM®

CUT ALONG THIS LINE

DOS/VS System Utilities Printed in U.S.A. GC33-5381-2