SPERRY UNIVAC

PUBLICATIONS UPDATE

Operating System/3 (OS/3)

Data Utilities

User Guide/Programmer Reference

UP-8069 Rev. 9–B

This Library Memo announces the release and availability of Updating Package B to "SPERRY Operating System/3 (OS/3) Data Utilities User Guide/Programmer Reference", UP-8069 Rev. 9.

This update describes the MILOAD fast loader utility program that provides users with an efficient means of loading large MIRAM files.

Copies of Updating Package B are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry representative. To receive only the updating package, order UP-8069 Rev. 9–B. To receive the complete manual, order UP-8069 Rev. 9.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A02, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 75, 75U, 76 and 76U (Package B to UP-8069 Rev. 9, 31 pages plus Memo)	Library Memo for UP-8069 Rev. 9–B
		RELEASE DATE: June, 1983

SPERRY UNIVAC

PUBLICATIONS UPDATE

Operating System/3 (OS/3)

Data Utilities

User Guide/Programmer Reference

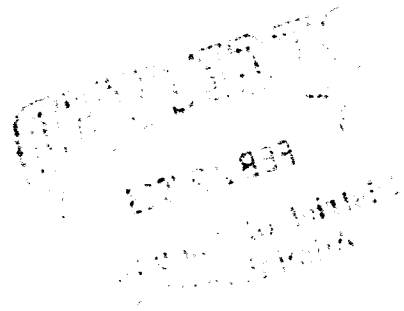
UP-8069 Rev. 9–A

This Library Memo announces the release and availability of Updating Package A to "SPERRY UNIVAC" Operating System/3 (OS/3) Data Utilities User Guide/Programmer Reference", UP-8069 Rev. 9.

This release 8.0 update emphasizes the importance of understanding various workstation considerations when running the DATA routine interactively:

Copies of Updating Package A are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry Univac representative. To receive only the updating package, order UP-8069 Rev.9–A. To receive the complete manual, order UP-8069 Rev. 9.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ,CZ and MZ	Mailing Lists A00,A02,18,18U,19,19U,20,20U,21,21U, 75,75U,76 and 76U (Package A to UP-8069 Rev. 9, 6 pages plus Memo)	Library Memo for UP-8069 Rev.9–A
		RELEASE DATE: December, 1982



SPERRY UNIVAC
COMPUTER SYSTEMS

ATTN: CHARLIE GIBBS

00759
CAV208M45541 8069 R9

UAS

SPERRY UNIVAC
1 - 1818 CORNWALL STREET
VANCOUVER B C

V6J 1C7

##

PUBLICATIONS REVISION

Operating System/3 (OS/3)

Data Utilities

User Guide/Programmer Reference (Series 90)
For System 80 see UP-8834

UP-8069 Rev. 9

This Library Memo announces the release and availability of "SPERRY UNIVAC® Operating System/3 (OS/3) Data Utilities User Guide/Programmer Reference", UP-8069 Rev. 9.

This revision includes:

- format change of I@DATA command;
- new menu selection screens in the sample interactive dialog;
- embedded card data in the batch processing environment;
- clarification of FV=(nnnnn), H=(nnnnnnn), X=(r,s), and YI keyword parameter descriptions;
- UDT subparameter change from OK to OX; and
- OM=([,n] [,V] [,R]) options deleted.

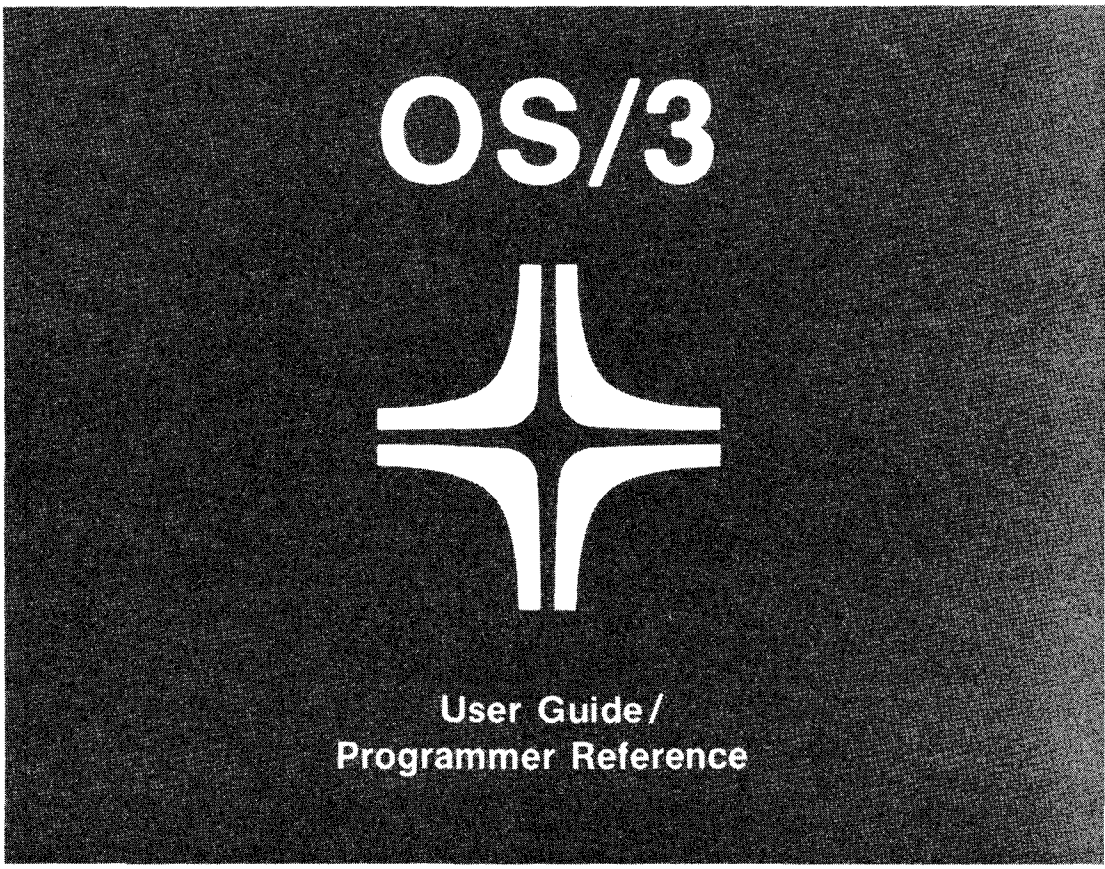
Destruction Notice: If you are going to OS/3 release 8.0, use this revision and destroy all previous copies. If you are not going to OS/3 release 8.0, retain the copy you are now using and store this revision for future use.

Copies of UP-8069 Rev. 8 and UP-8069 Rev. 8–A will be available for 6 months after the release of 8.0. Should you need additional copies of this edition, you should order them within 90 days of the release of 8.0. When ordering the previous edition of a manual, be sure to identify the exact revision and update packages desired and indicate that they are needed to support an earlier release.

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

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A02, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 75, 75U, 76, and 76U (Cover and 170 pages)	Library Memo for UP-8069 Rev. 9
		RELEASE DATE: September, 1982

Data Utilities



Environment: 90/25, 30, 30B, 40 Systems

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

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

Sperry Univac is a division of the Sperry Corporation.

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

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

PRINTED IN U.S.A.

PAGE STATUS SUMMARY

ISSUE: Update B – UP-8069 Rev. 9
RELEASE LEVEL: 8.1 Forward

Part/Section	Page Number	Update Level
Cover/Disclaimer		Orig.
PSS	1	B
Preface	1	B
Contents	1 thru 3	Orig.
	4	B
	5	B*
PART 1		
	Title Page	Orig.
1	1 thru 6	Orig.
PART 2		
	Title Page	Orig.
2	1 thru 19	Orig.
3	1 thru 72	Orig.
4	1 thru 9	Orig.
5	1 thru 30	Orig.
PART 3		
	Title Page	Orig.
6	1, 2	A
	2a	A
	3 thru 13	Orig.
PART 4		
	Title Page	B
7	1 thru 16	B*
PART 5		
	Title Page	B
Appendix A	1 thru 3	Orig.
Index	1 thru 6	B
User Comment Sheet		

Part/Section	Page Number	Update Level

Part/Section	Page Number	Update Level

*New pages

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

Preface

This manual instructs the programmer in the use of the SPERRY UNIVAC Operating System/3 (OS/3) data utilities program.

The manual is divided into the following parts:

- PART 1. INTRODUCTION

  Introduces the data utilities program and briefly describes its uses, characteristics, processing methods, and operating environments.

- PART 2. FILE PROCESSING IN A BATCH ENVIRONMENT

  Describes how to use the data utilities program in a batch environment. The data utilities control statements, sample job control streams, and jprocs are discussed.

- PART 3. FILE PROCESSING IN AN INTERACTIVE ENVIRONMENT

  Describes the interactive execution of the data utilities program. The dialog initiation, file processing operations, output listings, and error messages are described. A sample interactive dialog with step-by-step instructions is included.

- PART 4. LOADING LARGE, MULTIKEYED MIRAM FILES IN A BATCH ENVIRONMENT

  Introduces and describes how to use MILOAD, a special purpose utility that enhances performance when loading large, multikeyed MIRAM files.

- PART 5. APPENDIXES

  Describes the statement conventions.

Contents

3. DATA ROUTINE STATEMENTS

FIGURES

TABLES

PART 1. INTRODUCTION

1. Data Utilities Program

1.1. DATA ROUTINE

The data utilities program, commonly known as the DATA routine, gives you a straightforward, easy-to-use method for reproducing and maintaining your data files. It provides the capabilities for transferring files between various peripheral devices, editing or correcting data files, and comparing files.

The DATA routine is your file maintenance routine. All the information concerning file processing is submitted either via a job control stream (batch method) or via a question and answer session known as a dialog (interactive method). Both the batch and interactive methods describe your files to the DATA routine and inform it of the type of processing to be accomplished. The DATA routine enables you to compare files or selected areas of a file, delete or insert records, copy existing files to any storage device available in your system, rearrange records, or produce a printed copy of any file.

1.2. DATA ROUTINE USE

The following are a few examples of possible applications of the DATA routine:

- You might want to transfer data to a different type of device for the purpose of long range storage, or to take advantage of a faster access device.
- Data from one program could be used in another program by reformatting the records and transcribing them to a storage medium compatible with the receiving program.
- You may select or delete specific areas of a file for testing purposes or report preparation.
- If, for some reason, you need to know the contents of a file, it is readily available to you through the use of the print option.
- To make certain that no discrepancies have occurred during a copy operation, you can compare the input file to the output file.

1.3. OPERATING ENVIRONMENT

The operating environment for the DATA routine is established at system generation time. The operating environment that you choose governs the types of files that you can process and how you can execute the DATA routine. You have the following options:

- Basic Data Management

  The basic data management environment allows you to perform operations on card, card image diskette-data set label, tape, printer, and SAM, DAM, NI, ISAM, IRAM, or MIRAM disk files.

- Consolidated Data Management

  The consolidated data management environment allows you to perform operations only on consolidated data management files (card, printer, tape, MIRAM disk, MIRAM diskette, or data set label diskette). Note that "card image" diskette files can be used only as input files in this environment. Also, you can transfer data on diskette from System 80 to Series 90 in data set label mode by using consolidated data management.

- Mixed Data Management (Basic and Consolidated)

  The mixed data management environment allows you to perform operations on basic data management files (card, card image diskette-data set label, tape, printer, MIRAM disk, or MIRAM diskette). In the basic data management environment, data-set label diskette files are created as unkeyed MIRAM data files.

Thus, the operating environment also governs how the DATA routine can be executed. Table 1-1 lists the operating environments and how the DATA routine can be executed in each case.

Table 1—1. Operating Environments and Execution Methods

Operating Environment	Execution Method
Basic data management	Batch
Mixed (basic/consolidated)	Batch and interactive
Consolidated data management	Batch and interactive

1.4. DEVICE UNIT COMBINATIONS

In the process of copying data from one device to another of the same type, you can move data from:

- Card to card
- Tape to tape
- Disk to disk

You may also transcribe data from one type of storage medium to another. The combinations of devices you can use are:

- Card to tape
- Card to disk
- Card to printer
- Tape to card
- Tape to disk
- Tape to printer
- Disk to card
- Disk to tape
- Disk to printer

In the basic data management environment, input and output data from the SPERRY UNIVAC 8413 Diskette Subsystem is processed the same as for the card reader, punch, or read/punch device. Therefore, any reference to card reader or punch in the DATA routine can also apply to the 8413 diskette subsystem.

In the consolidated data management environment, input and output data from an 8413 diskette subsystem is processed the same as for the disk. Therefore, any reference to a disk in the DATA routine can also apply to the 8413 diskette.

As in all areas of OS/3 data utility programming, any of the operations can be performed with a minimum of programming effort by the use of a few utility input and output statements. All the necessary interfaces with you and with the system operator, including related software components and your data files, are accomplished with software components of the OS/3 disk operating system.

1.5. FILE ORGANIZATION

When operating in the basic data management environment, your devices are assigned via the job control stream. However, when you use either the mixed or the consolidated data management environment, your devices are assigned via either the interactive dialog or the job control stream.

In the basic data management environment, your disk files can be organized in one of the following access methods:

- Sequential Access Method (SAM)

  SAM files are constructed sequentially. Records are accessed in the same way, starting at the first record and searching through the file until the required record is detected. SAM files are used on magnetic tape units, card readers, card punches, printer files, and disks. SAM is the only access method that can be used for card, tape, and printer files.

- Indexed Sequential Access Method (ISAM)

  Disk files can use the ISAM organization. Records are written sequentially either by record identification, or by key, and then indexed. Records can be retrieved randomly by key.

- Indexed Random Access Method (IRAM)

  Records are written on an IRAM file in the order they are presented. The file may be indexed or nonindexed, depending on your requirements. In addition, indexed files may be created by means of an orderly load (records submitted in ascending key order) or a disorderly load (record keys in no particular order). An IRAM file can be accessed in many ways; however, data utilities will access the records consecutively only by key or by relative record number. (See the basic data management user guide, UP-8068 (current version) for details on other ways to access records in an IRAM file.)

- Multiple Indexed Random Access Method (MIRAM)

  A MIRAM file is similar to an IRAM file in that records are written on the file in the order they are presented and the file can be indexed or nonindexed. Disorderly load is permitted with indexed files, and it can also be accessed in other ways. However, as with an IRAM file, the DATA routine will access records consecutively only by key or by relative record number.

  The MIRAM file differs from the IRAM file in that variable-length records, multiple keys, duplicate keys, and logical deletion of records from the file are permitted. In addition, if multiple keys are specified, a separate index is created for each key type.

- Direct Access Method (DAM)

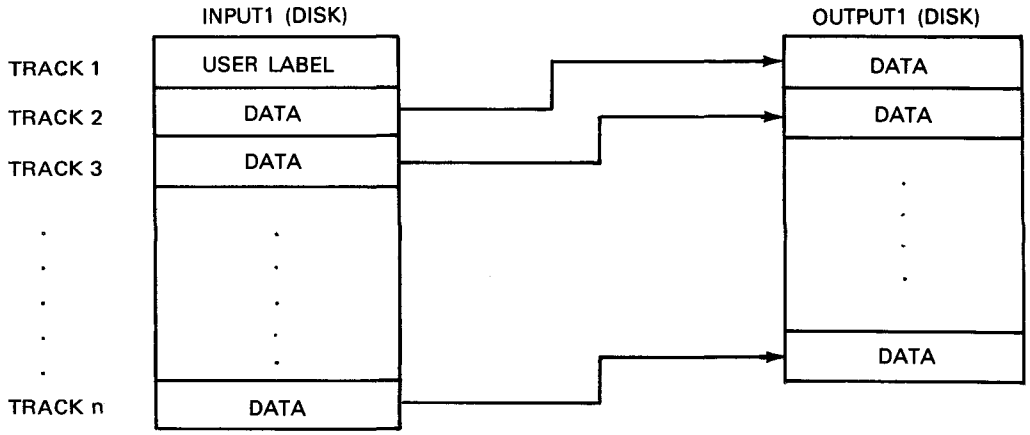
  The DAM organization, for use by disk files, constructs a file by relative track addressing. Records are accessed by requests for a particular record number.

- Non-Indexed (NI)

  NI allows for multiple partition files and combines the characteristics of SAM and DAM.

When using the consolidated data management environment, only the MIRAM access method is applicable.

In both the basic and consolidated data management environments, the DATA routine copy functions are not designed to copy files with user labels. However, you can use a disk file with user labels as input, and the data portion can be an output to another disk, card, tape, or printer. User labels occupy the first track of the file. When you implement data management functions, the DATA routine will bypass the first track and start processing the data. For example:



You can find detailed information concerning data and file structures in the current versions of the basic data management user guide, UP-8068, or the consolidated data management concepts and facilities, UP-8825.

1.6. FILE PROCESSING MODES

1.6.1. Batch Processing

Batch processing consists of coding a set of instructions telling the DATA routine your processing requirements. These instructions (the job control stream) assign the necessary input/output devices and data utilities statements that describe what you want done.

After you have completed your coding, the job control stream must be transcribed onto punched cards or a diskette. The card deck or diskette is then submitted to the system operator for subsequent execution. For details, see Part 2.

1.6.2. Interactive Processing

Interactive processing consists of a dialog (question and answer session) that you conduct with the DATA routine via a workstation. During the dialog, questions appear on the workstation screen. The answers given will assign the necessary input/output devices and tell the DATA routine what you want done.

Interactive processing achieves the same results as batch processing; however, it is much simpler to use because:

- you do not need to be familiar with the job control statements or the DATA routine to use it; and
- the DATA routine is executed immediately (that is, you do not have to wait for the system operator to execute it for you).

For more details on interactive processing, see Part 3.

PART 2. FILE PROCESSING IN A BATCH ENVIRONMENT

2. Batch Processing Considerations

2.1. JOB CONTROL STREAM REQUIREMENTS

You must code a job control stream when you execute the DATA routine in a batch environment. The control stream consists of:

- // JOB statement
- Device assignment sets (statements that describe the files you intend to use or create)
- // EXEC DATA statement
- Start-of-data statement (/$)
- Data utilities control statements
- End-of-data statement (/*)
- End-of-job statement (/&)
- // FIN statement
- PARAM statements

You can have your job control stream define your input file in a device assignment set or as embedded card data. Embedded card data is inserted between the data delimiters (/$ and /*). Jprocs are described in 2.1.11 and a sample job control stream is provided in 2.1.12.

2.1.1. // JOB Statement

The // JOB statement is the first statement in the job control stream. It names the job and specifies the amount of main storage required for the job.

2.1.2. Device Assignment Set

The device assignment sets you use in your job control stream consist of the following statements that you must use in this order:

DVC
VOL
EXT
LBL
LFD

1. DVC

   Designates the device you require.

2. VOL

   Identifies a tape or disk volume by serial number.

3. EXT

   Needed only when you allocate disk space.

4. LBL

   Identifies a tape or disk file by file identifier.

5. LFD

   Specifies the logical file name and links your file description with the corresponding data management file definition.

The DVC and LFD statements are always required in your device assignment set. The VOL and LBL statements are needed only when a tape or disk file is identified in your job control stream. When you are copying to a disk, be sure to allocate sufficient space on the disk by using the EXT statement. A device assignment set is not required for the card reader if your primary (input) file is embedded card data.

To describe your input/output files to the DATA routine, use the following logical file names as they apply to the devices you assign to your file:

Logical File Name	Use
// LFD PRNTR	Needed to allocate a printer to the job. If a printer is allocated, the DATA routine prints out the start-of-data (/$) statement, the Uio statement, any utility modifier statements, and the end-of-data (/*) statement as each is read from the control stream. Error messages, if applicable, are also printed. If a printer is not allocated to the job, no headings or control stream listings are generated. If an error should occur, a data management DMxx error message and a data utilities DU fatal error message are displayed on the system console. If the functions specified require a printer and none is allocated, a fatal error condition terminates the job. A descriptive listing of all data management and data utilities error messages is contained in the system messages programmer/operator reference, UP-8076 (current version).
// LFD INPUT1	Needed for all runs except embedded data; defines the input file for a copy generation or the primary input file for a compare operation. If not present, DATA assumes INPUT1 is embedded card data.
// LFD INPUT2	Needed only when a compare operation is specified; defines the secondary input.
// LFD OUTPUT1	Needed for a copy operation only when the printer is not the primary output.
// LFD OUTPUT2	Needed for a copy operation only when the dual output option (DC) specifies card output; defines the card punch.

The DATA routine also accepts file characteristics from a // DD job control statement. When this // DD statement is present within the device assignment set for a file, the DATA routine uses the information specified in the // DD statement rather than the information provided in the data utilities statements; that is, the // DD statement overrides the data utilities statements. For disk input files, the file characteristics specified in the format labels override the information in both the data utilities statements and the // DD statement.

The EXEC statement follows the device assignment sets and initiates execution of the DATA routine. The parameter for this statement must be DATA.

2.1.3. // EXEC DATA Statement

The // EXEC DATA statement follows the device assignment set and starts the execution of the DATA routine.

2.1.4. Start-of-Data Statement (/$)

This statement follows the // EXEC DATA statement and indicates the start of the data utilities control statements.

2.1.5. Data Utilities Control Statements

These statements (the Uio statement and the modifier statements) specify the file processing to be performed. A description of the data utilities control statements is found in Section 3.

2.1.6. End-of-Data Statement (/*)

This statement follows the last data utilities control statement and indicates the end of the control statements.

2.1.7. Embedded Card Data

The INPUT1 file for data utilities can be entered in your control stream as embedded card data. This method eliminates the need for assigning your input files in your job control stream. The INPUT1 file and the Uio statement are inserted between two sets of data delimiters (/$ and /*).

2.1.8. End-of-Job Statement (/&)

This statement indicates the end of the job.

2.1.9. // FIN Statement

This statement indicates that no more statements are read for this job control stream and turns off the card reader.

2.1.10. // PARAM Statements

There are four // PARAM statements you can supply to the DATA routine to control operation. They are: // PARAM CONTROL, // PARAM MODE, // PARAM DISPLAY, and // PARAM EOJ control statements. When supplied, they must appear immediately following the // EXEC DATA statement but before the /$ statement. They are supported in all operating environments.

2.1.10.1. // PARAM CONTROL

Use this statement to print data utilities control statements on the final output.

Format:

```
// PARAM CONTROL={YES}
                 {NO }
```

Parameters:

YES

Specifies that PARAM statements and Uio statements are printed on the final output.

NO

Specifies that PARAM statements and Uio statements are not printed on the final output.

2.1.10.2. // PARAM MODE

This statement is used to specify the debug or OS/4-to-OS/3 data conversion mode.

- Debug Mode

  Provides snap dumps of the job region at critical points during execution and is made operative by including the following PARAM statement in the control stream immediately following the // EXEC DATA statement:

  ```
  // PARAM MODE=DBG
  ```

  This mode of operation should be used only to provide documentation in reporting a software user report (SUR).

- OS/4-to-OS/3 Data Conversion Mode

  Used to convert OS/4 disk files to OS/3 disk files. Before using this mode, you must use the OS/4 disk data conversion utility (DCON4) to dump the OS/4 files and their characteristics onto a tape file. Then you use the DATA routine to read the tape and create OS/3 disk files. (See the UTD statement in 3.2.2.2.) To do this, include the following PARAM statement in your control stream immediately following the // EXEC DATA statement:

  ```
  // PARAM MODE={OS4 }
                {OS4D}
  ```

  The MODE=OS4 specification produces MIRAM files in a mixed or consolidated data management environment. To obtain DTF files, add a D to OS4 (MODE=OS4D).

  You can also convert OS/4 disk files to OS/3 disk files interactively in a mixed or consolidated data management environment. The input file is created by using DCON4, and the output file is always MIRAM.

  For a detailed description of DCON4 use and the conversion mode, see the OS/4 to OS/3 disk data conversion utility user guide/programmer reference, UP-8606 (current version), and the OS/4 to OS/3 conversion guide user guide/programmer reference, UP-8553 (current version).

2.1.10.3. // PARAM DISPLAY

This statement specifies the files where the termination information is written.

Format:

```
// PARAM DISPLAY = {P   }
                   {L   }
                   {C   }
                   {NONE}
```

Parameters:

P
Indicates the termination information is listed on the printer.

L
Indicates the termination information is written to the system log file.

C
Indicates the termination information is displayed on the system console.

NONE
Indicates no termination information is written.

NOTE:

Any combination (PLC) can be used and commas are not required. If omitted, PLC is assumed; however, detail statistics will be written to printer and log files and not displayed on the system console.

2.1.10.4. // PARAM EOJ

This statement indicates what occurs when the DATA routine terminates.

Format:

```
// PARAM EOJ = {CANCEL}
               {UPSI  }
```

Parameters:

CANCEL
Indicates that the DATA routine terminates with error code 0 if a fatal or serious error is encountered and the UPSI byte is not modified.

UPSI
Indicates that the DATA routine terminates normally in all cases. At job termination, the UPSI byte settings (Table 2-1) are listed on the printer.

Table 2—1. UPSI Byte Settings

Setting	Category
X'00'	Informative
X'20'	Warning
X'40'	Serious
X'80'	Fatal

2.1.11. Job Control Procedure Use

Each time you use the DATA routine in a batch environment, you must provide a job control stream. As time goes on, you will note that you are coding the same sequence of job control statements over and over. This repetitious coding can be avoided by using a single statement (a job control procedure call statement) in your job control stream in place of the job control statements you normally would code. This jproc call statement will generate the proper job control statement sequence for you. (For more details, see Section 5.)

2.1.12. Sample Job Control Stream

Figure 2–1 is an example of a DATA routine control stream, showing the device assignment set, the EXEC statement, the Uio statement, and the placement of data cards.

```
     1         10         20         30
1.  // JOB TESTFIL,,A000
2.  // DVC 20        // LFD PRNTR
3.  // DVC 30        // LFD INPUT1
4.  // DVC 51        // VOL DSP012
5.  // EXT SQ,C,,CYL,8
6.  // LBL WRKFIL3    // LFD OUTPUT1
7.  // EXEC DATA
8.  /$
9.  UCD keyword parameters
10. /*
11. /&
12. // FIN
13.       data cards
14. /*
```

Figure 2—1. Sample Job Control Stream for Card-to-Disk Operation

Line 1 shows the job beginning, job name of TESTFIL, and the main storage requirements. Line 2 assigns the printer to the job. Line 3 assigns the card reader as the input file. Line 4 assigns the disk file with the volume serial number DSP012 to this job. Line 5 obtains disk space, specifies SAM format, and allocates contiguous cylinder space on eight cylinders. Line 6 identifies the output file by file name. Line 7 requests the loading and execution of the DATA routine. Line 8 designates the start of control cards. Line 9 is the Uio statement that identifies this job as a card-to-disk operation and includes any necessary keyword parameters. Line 10 designates the end of control cards. Line 11 indicates the end of job. Line 12 terminates the card reader. Line 13 shows the location of your data cards in the control stream. If your input file is to be spooled, you will precede your data cards with a // DATA control statement and follow your data cards with a // FIN statement to terminate the card reader. Line 14 designates the end of data.

If you require more detailed information concerning job control statements and their parameters, refer to the job control user guide, UP-8065 (current version).

Figure 2–2 is an example of a DATA routine control stream using embedded card data.

```
 1. // JOB TESTFIL,,A000
 2. // DVC 20          // LFD PRNTR
 3. // DVC 51          // VOL DSP012
 4. // EXT MI,C,,CYL,8
 5. // LBL WRKFIL3     // LFD OUTPUT1,,INIT
 6. // EXEC DATA
 7. /$
 8. UCD keyword parameters
 9. /*
10. /$
11. .
12. .
13. .
14. embedded card data
15. .
16. .
17. .
18. /*
19. /&
20. // FIN
```

Figure 2—2. Sample Job Control Stream Using Embedded Card Data

Notice that with embedded card data, the device assignment set for the card reader is not needed. Line 10 designates the start-of-data. Lines 11 through 17 show the location of your embedded card data. Line 18 signifies the end of data. Line 19 indicates the end of job. Line 20 terminates the card reader.

If you require more detailed information concerning embedded card data, refer to the job control user guide, UP-8065 (current version).

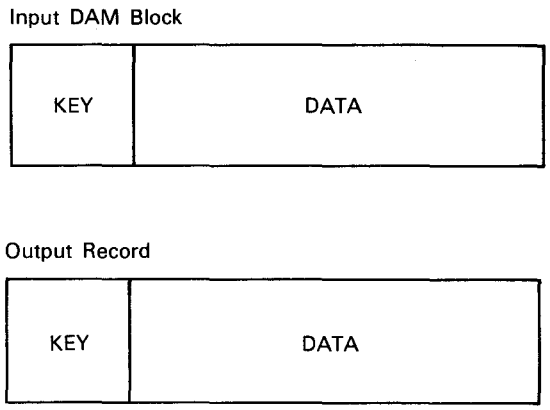
2.2. FILE COPYING CONSIDERATIONS

File copying considerations are described for the basic data management and mixed operating environments. Since most OS/3 file formats are compatible with each other, the DATA routine copies between these files without change. However, in certain cases, the DATA routine converts the format of the INPUT1 file to conform to the format of the OUTPUT1/OUTPUT2 file.

These conversions are not performed if you specify field select. When you use field select, the DATA routine assumes that you are performing all necessary conversions. For file compare (K2), the DATA routine converts the input record before the comparison is made.

2.2.1. Keyed DAM Fixed Unblocked File to Non-DAM File

The DATA routine treats the whole keyed DAM input block as a logical record. When defaulting output record length, the DATA routine computes a record length equal to the DAM record length plus the DAM key length. The output record is logically equal to the input block.

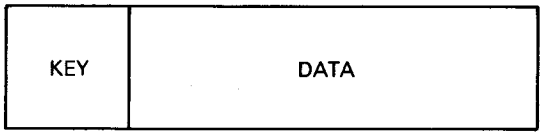


The KEY in the output record is used to show only the portion of the output record that represents the DAM input block key and has no relation to the actual record key in the output file (if one exists).

2.2.2. Non-DAM File to Keyed DAM Fixed Unblocked File

The DATA routine creates a keyed DAM block from each input record. When defaulting the output record length, the DATA routine computes a record length equal to the input record length minus the output key length.

Input Record



Output DAM Block

The KEY in the input record is used to show only the portion of the input record that is used as the DAM input block key and has no relation to the actual record key in the output file (if one exists).

2.2.3. Variable Blocked ISAM File to Non-ISAM File (Except DAM)

Since the variable record descriptor word for ISAM records is two bytes shorter than the record descriptor word for the other OS/3 file types, the DATA routine creates an extra two bytes in the input record descriptor word. These bytes contain X'00'.

Input ISAM Record

Output Record

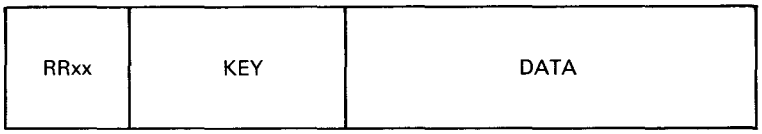
2.2.4. Non-ISAM (Except DAM) to Variable Blocked ISAM File

In this case, the conversion is the reverse of that described in 2.2.3.

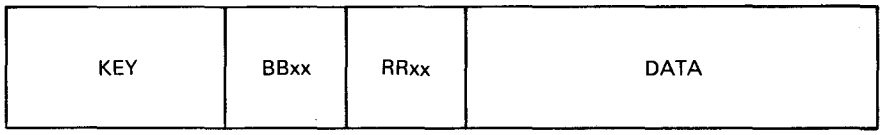
2.2.5. Variable SAM/MIRAM/Tape File to Variable Unblocked DAM File

The DATA routine considers each input record as a DAM block; consequently, the following conversion is made.

Input Record



Output DAM Block



BBxx denotes the block descriptor word that must be allowed for in the output block area. The BB is provided by data management. The xx is required by data management and is blank-filled by the DATA routine.

The key length used is the key length of the DAM file. If the key length is zero, no move will be made and the whole input record will be treated as data.

The KEY in the input record is used to show only the portion (if any) of the input record that is used as the DAM output block key (if any) and has no relation to the actual record key in the input file (if one exists).

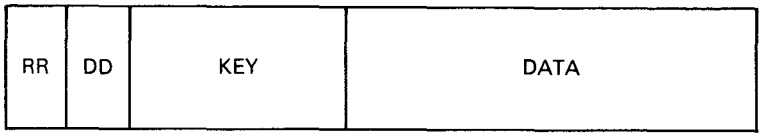
2.2.6. Variable Unblocked DAM File to Variable SAM/MIRAM/Tape File

In this case, the conversion is the reverse of that described in 2.2.5.

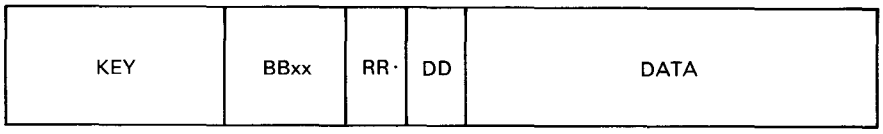
2.2.7. Variable Blocked ISAM File to Variable Unblocked DAM File

This conversion is physically identical to the conversion of other variable records to a DAM block except that there are some logical differences. Also, the DATA routine ignores the fact that the ISAM record descriptor word (RR) is only two bytes.

Input ISAM Record



Output DAM Block



BBxx denotes the block descriptor word that must be allowed for in the output block area. The BB is provided by data management. The xx is required by data management and is blank-filled by DATA.

DD denotes the first two bytes of input data record. These bytes are moved intact to the third and fourth bytes of the output record descriptor word.

The key length used is the key length of the DAM file. If the key length is zero, then no move will be made and the whole input record will be treated as data.

The KEY in the ISAM input record is used to show only the portion of the record that is moved to the key field in the DAM block and has no relation to the ISAM record key.

2.2.8. Variable Unblocked DAM File to Variable Blocked ISAM File

In this case, the conversion is the reverse of that described in 2.2.7.

2.3. OUTPUT LISTINGS

The DATA routine provides output listings in two forms:

- Listings of output data in different formats.
- Automatic display of the file statistics for your job's primary files (INPUT1 and OUTPUT1/INPUT2) at termination.

2.3.1. Printer Formats

The DATA routine can produce printed copy output in two formats: display or list. By your selection of keyword parameters in the Uio statement, you can designate which format you prefer and whether the character mode should be EBCDIC or hexadecimal. Hexadecimal displays require twice as many print positions as the same data displayed in EBCDIC. Figures 2-3 through 2-8 show examples of the display and list formats.

2.3.1.1. Display Format

The first 20 print positions of the first line contain column headings; the remainder of the line shows the position of each byte.

Subsequent lines show the physical location and size of each block and record, as well as the contents of each byte. The columns, record number (REC#), block size (BLKSZ), and record size (RCSZ), represent the following:

Column	Description
REC#	The number of the record relative to the first record of the file. All records are numbered starting with 1.

Column	Description
BLKSZ	The input block size; i.e., for fixed-length, nondisk files, this number is taken from the b portion of the *A* keyword parameter (A=(r,b)). For disk files, this number is taken from the VTOC. For DAM files, this number is equal to the sum of the input key length plus input data length.
RCSZ	The output record size; i.e., for fixed-length SAM, IRAM, or ISAM files, this number is taken from the r portion of the keyword *B* parameter (B=r,b)). For DAM files, this number is the output data length. For variable-length SAM, ISAM, or MIRAM output files, this number is the record length given in the first two bytes of each output record.

Figures 2–3 through 2–5 are examples of the display format in EBCDIC and hexadecimal modes.

```
 REC NO   BLKSZ  RECSZ  1.......10 .......20 .......30 .......40 .......50 .......60 .......70 .......80 .......
00000001  00080  00080  MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 DATA RECOR 01
00000002  00080  00080  MIRAMKEY02 MIRAMKEY02 MIRAMKEY02 MIRAMKEY02 MIRAMKEY02 DATA RECOR 02
00000003  00080  00080  MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 DATA RECOR 03
00000004  00080  00080  MIRAMKEY04 MIRAMKEY04 MIRAMKEY04 MIRAMKEY04 MIRAMKEY04 DATA RECOR 04
00000005  00080  00080  MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 DATA RECOR 05
00000006  00080  00080  MIRAMKEY06 MIRAMKEY06 MIRAMKEY06 MIRAMKEY06 MIRAMKEY06 DATA RECOR 06
00000007  00080  00080  MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 DATA RECOR 07
00000008  00080  00080  MIRAMKEY08 MIRAMKEY08 MIRAMKEY08 MIRAMKEY08 MIRAMKEY08 DATA RECOR 08
00000009  00080  00080  MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 DATA RECOR 09
00000010  00080  00080  MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 DATA RECOR 010
00000011  00080  00080  MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 DATA RECOR 010
00000012  00080  00080  MIRAMKEY12 MIRAMKEY12 MIRAMKEY12 MIRAMKEY12 MIRAMKEY12 DATA RECOR 012
00000013  00080  00080  MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 DATA RECOR 013
00000014  00080  00080  MIRAMKEY14 MIRAMKEY14 MIRAMKEY14 MIRAMKEY14 MIRAMKEY14 DATA RECOR 014
00000015  00080  00080  MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 DATA RECOR 015
00000016  00080  00080  MIRAMKEY25 MIRAMINDX16 MIRAMMIX16 MIRAMKEY16 MIRAMKEY16 DUPLICATE  RECORD 16
00000017  00080  00080  MIRAMKEY26 MIRAMKEY16 MIRAMMIX16 MIRAMKEY16 MIRAMKEY16 DUPLICATE  RECORD 16
00000018  00080  00080  MIRAMKEY17 MIRAMKEY17 MIRAMKEY17 MIRAMKEY17 MIRAMKEY17 MIRAM RECO RD 17
00000019  00080  00080  MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAM RECO RD 18
00000020  00080  00080  MIRAMKEY19 MIRAMKEY19 MIRAMKEY19 MIRAMKEY19 MIRAMKEY19 MIRAM RECO RD 19
00000021  00080  00080  MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAM RECO RD 20
```

Figure 2—3. EBCDIC Mode Display Format

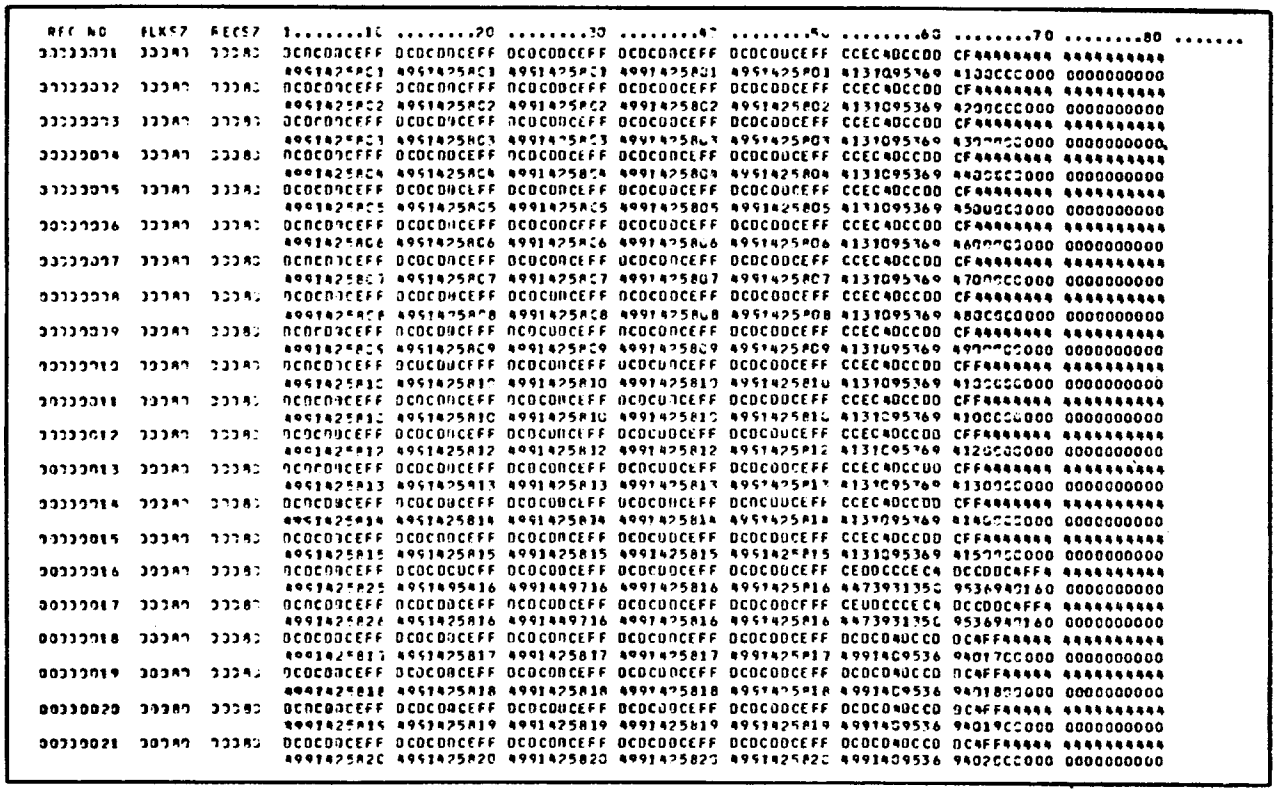


Figure 2—4. Hexadecimal Mode Display Format

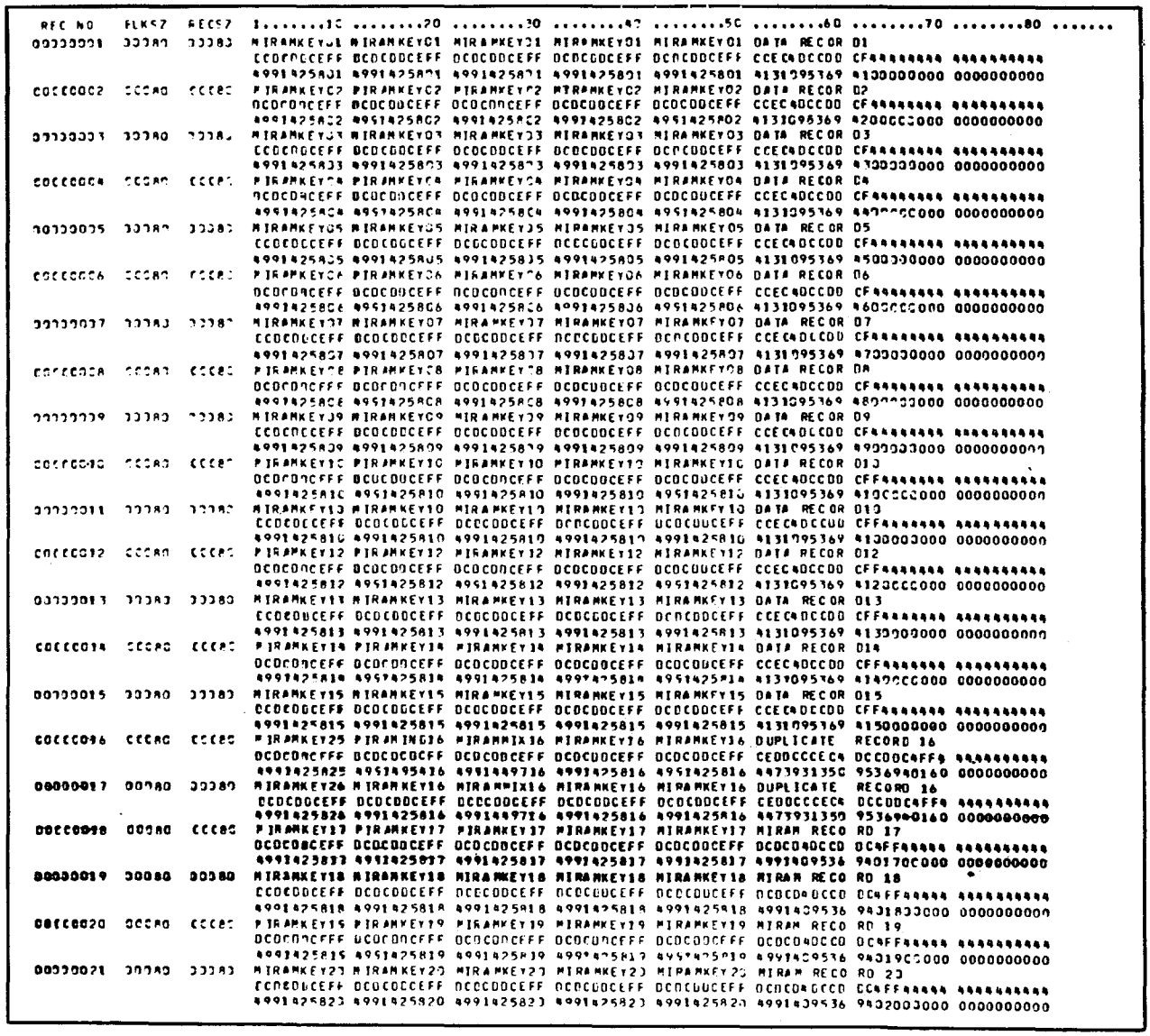


Figure 2—5. Combination of EBCDIC and Hexadecimal Mode Display Format

2.3.1.2. List Format

The list format displays your files, byte by byte. When you request the list format in combination EBCDIC and hexadecimal, the position of each byte is printed on the following line.

Figures 2-6 through 2-8 show examples of the list format.

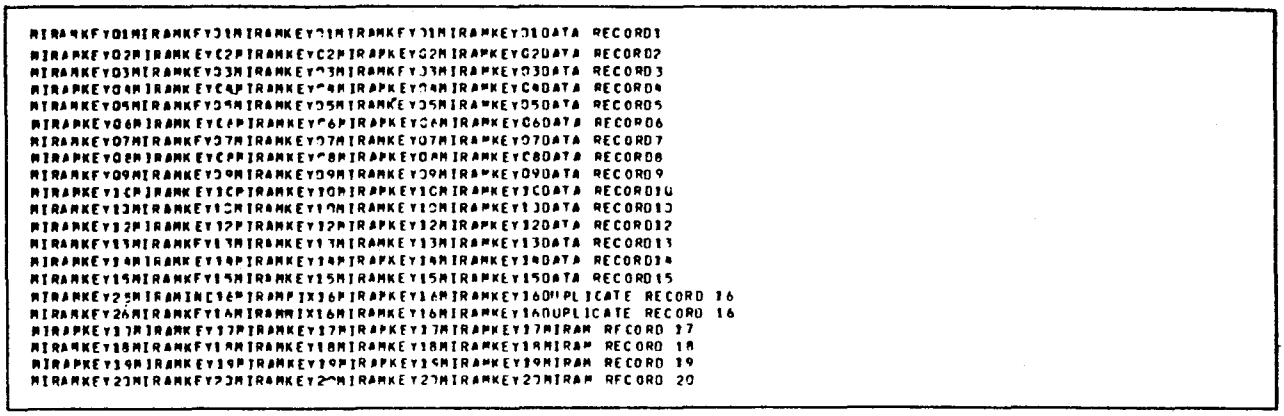


Figure 2—6. EBCDIC Mode List Format

Figure 2—7. Hexadecimal Mode List Format

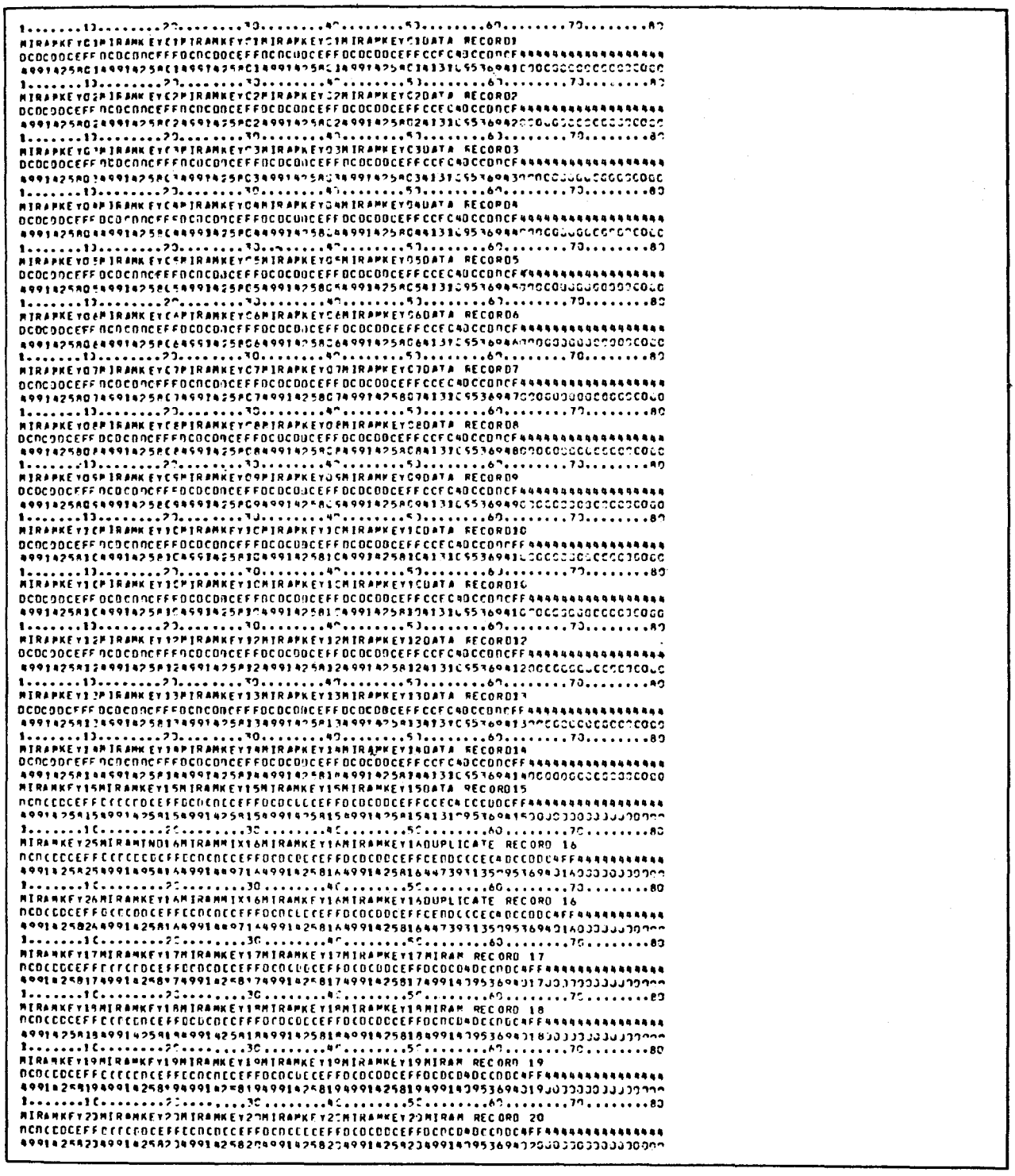


Figure 2—8. Combination of EDCDIC and Hexadecimal Mode List Format

2.3.2. Termination Information

Upon termination of the DATA routine, file statistics for your primary files are written to the system log file or the printer file. They are written to the system console if you include a // PARAM DISPLAY statement (2.1.10.3) in your job control stream. When you execute a program in basic or mixed data management environments, the format of the termination information for card, tape, printer, and non-MIRAM disk files is as follows (consolidated data management termination information is presented in 6.3):

- INPUT1

```
INPUT1...(FILENAME).....{DISK  }
                        {CARD  }
                        {TAPE  }
                        {DCON4 }
RECORD SIZE.....nnnnn
BLOCK SIZE.....nnnnn
KEY LENGTH.....nnnnn
KEY LOCATION.....nnnnn
RECORD FORMAT.....{FIXBLK}
                  {FIXUNB}
                  {VARBLK}
                  {VARUNB}
FILE ORG.....{SAM                }
             {NI                 }
             {DAM                }
             {ISAM               }
             {IRAM {CONSEC}      }
                   {INDEX }      }
```

- OUTPUT1/INPUT2

```
OUTPUT1/INPUT2.....{DISK    }
                   {TAPE    }
                   {CARD    }
                   {PRINTER }
RECORD SIZE.....nnnnn
BLOCK SIZE.....nnnnn
KEY LENGTH.....nnnnn
KEY LOCATION.....nnnnn
RECORD FORMAT.....{FIXBLK}
                  {FIXUNB}
                  {VARBLK}
                  {VARUNB}
FILE ORG.....{SAM                }
             {NI                 }
             {DAM                }
             {ISAM               }
             {IRAM {CONSEC}      }
                   {INDEX }      }
```

The format of the termination for MIRAM disk files is:

```
INPUT1/INPUT2/OUTPUT1...(FILENAME).....DISK
RECORD SIZE.............................nnnnn
BLOCK SIZE..............................nnnnn
KEY LENGTH..............................nnnnn
KEY LOCATION............................nnnnn
MIRAM KEY      LOC      LEN      CHG      DUP
  KEY1         nnnnn    nnnnn    Y/N      Y/N
  KEY2         nnnnn    nnnnn    Y/N      Y/N
  KEY3         nnnnn    nnnnn    Y/N      Y/N
  KEY4         nnnnn    nnnnn    Y/N      Y/N
  KEY5         nnnnn    nnnnn    Y/N      Y/N
INDEX BUFFER SIZE.......................nnnnn
RECORD CONTROL BYTE.....................{YES}
                                        {NO }
RECORD SLOT SIZE........................nnnnn
DISK SECTOR SIZE........................nnnnn
VOLUME MOUNT SETTING....................{SINGLE}
                                        {MULTI }
RECORD FORMAT...........................{FIXBLK}
                                        {VARBLK}
FILE ORG...MIRAM {CONSEC}...RCDS UNEQUAL..nnnnnnnn
                 {INDEX }
```

NOTE:

RCDS UNEQUAL field appears only on INPUT2 termination information.

2.4. ERROR MESSAGES

If an error occurs during the execution of the DATA routine, error messages are produced. The error messages are displayed on the printer, the system console, the system log file, and the workstation.

The errors are grouped into four categories as follows:

1. Informative
2. Warning
3. Serious
4. Fatal

Table 2-1 lists the UPSI byte settings for these categories. All the error messages are listed in the system messages programmer/operator reference, UP-8076 (current version).

3. DATA Routine Statements

3.1. BASIC UTILITY INPUT AND OUTPUT STATEMENT

The basic utility input and output statement (Uio) is used to specify the copy or compare functions that you wish the DATA routine to perform. The Uio statement identifies the input and output devices required and describes the data format of your files to the DATA routine through the keyword parameters you specify. Keyword parameters allow you to describe:

- the format and access method of your input and output files;
- whether this is a copy or compare function;
- the file locations where processing begins;
- the number of unequal records accepted, or the number of blocks or records processed before DATA routine termination;
- whether the output record is written to more than one device;
- whether input and output tapes are rewound; and
- whether printer mismatches are ignored or terminate the DATA routine.

To do this, you must include the appropriate Uio statement in your control stream between the /$ and /* job control statements following the // EXEC DATA job control statement.

3.1.1. Input and Output Statement Mnemonics

The utility input and output statement contains the mnemonic Uio, which you code to identify the types of devices you require for input and output.

In this mnemonic:

- U is a constant that identifies the utility input and output statement.
- i is a variable that identifies the input device as a card reader or 8413 diskette (C), magnetic tape unit (T), or disk (D).
- o is a variable that identifies the primary output device as a card punch or 8413 diskette (C), magnetic tape unit (T), disk (D), or printer (P).

Valid Uio mnemonics are:

Mnemonic	Input and Output Device
UCC	Card reader to card punch
UCD	Card reader to disk
UCT	Card reader to tape
UCP	Card reader to printer (Default option – When you copy a card file to the printer and require no modifications to the file, no Uio mnemonic is needed.)
UDC	Disk to card punch
UDD	Disk to disk
UDT	Disk to tape
UDP	Disk to printer
UTC	Tape to card punch
UTD	Tape to disk
UTT	Tape to tape
UTP	Tape to printer

Figure 3-1 shows the Uio mnemonics and their relationship to the input and output devices needed to copy or compare files.

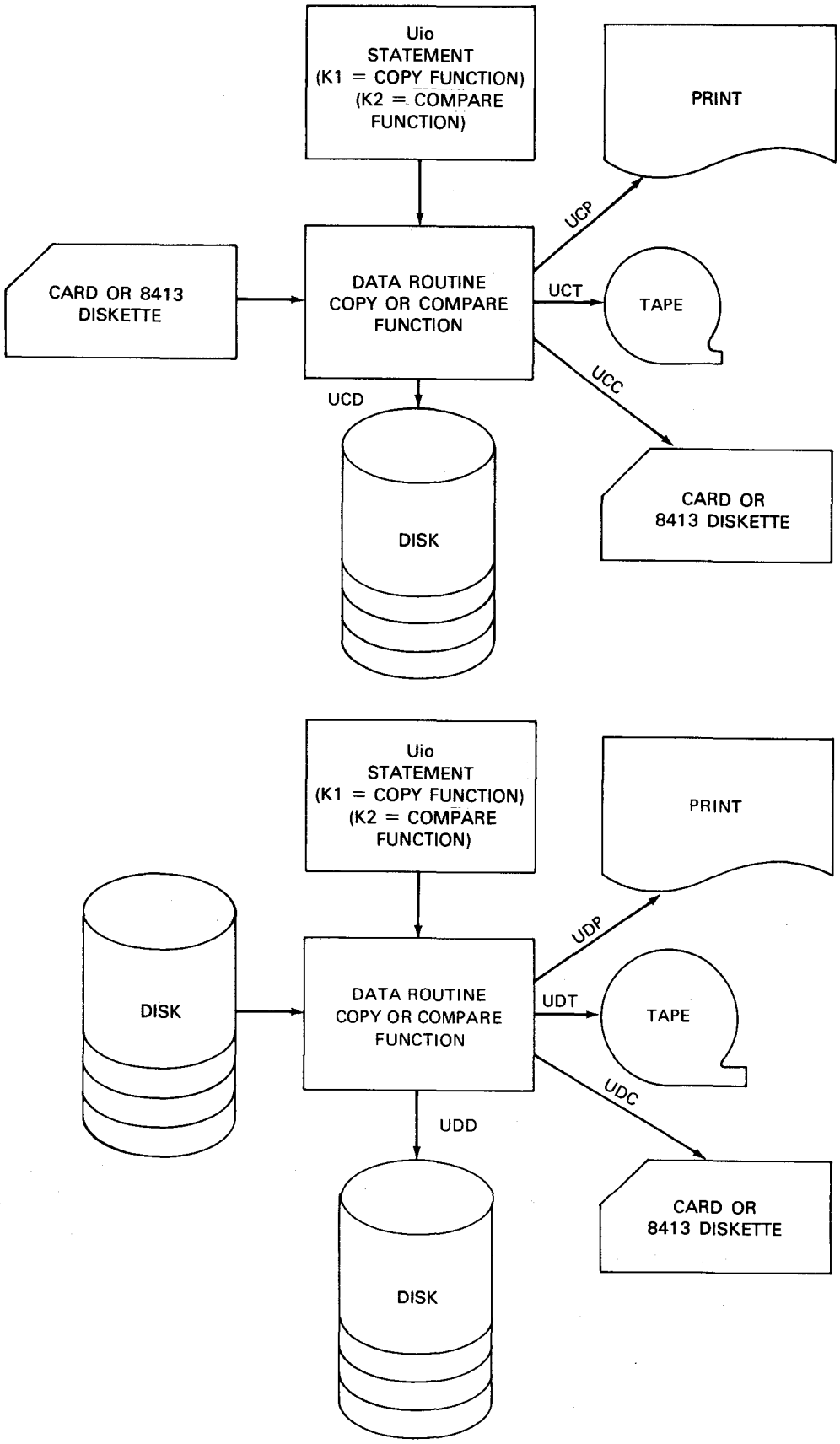


Figure 3—1. Relationship of Uio Mnemonics to Input and Output Devices (Part 1 of 2)

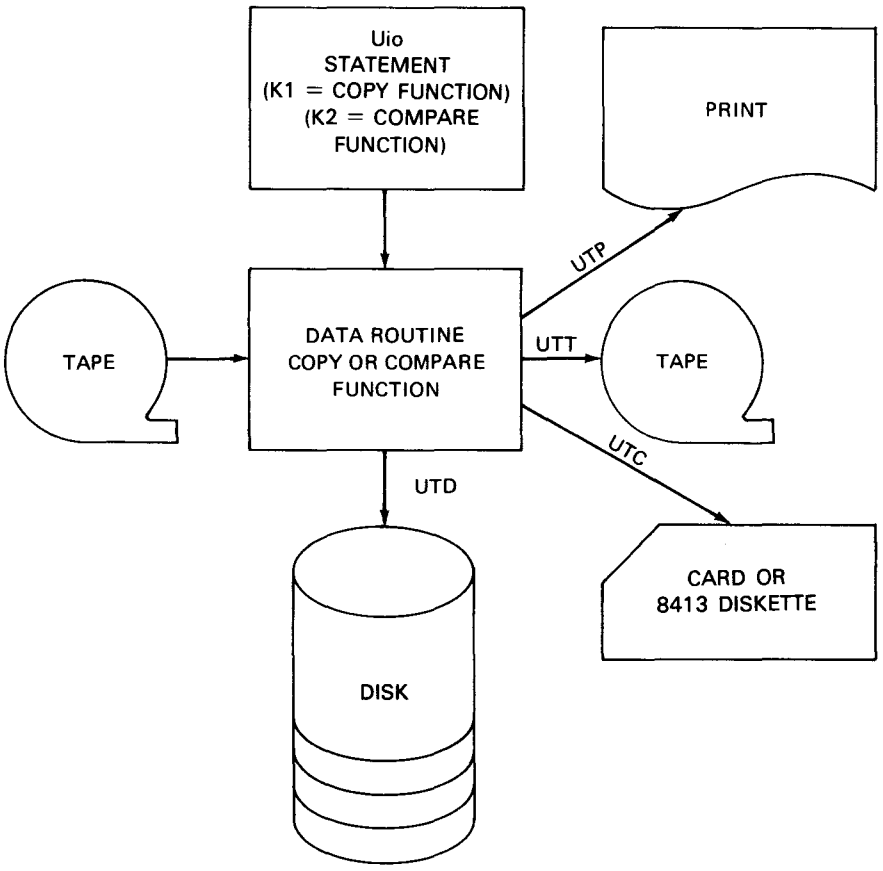


Figure 3—1. Relationship of Uio Mnemonics to Input and Output Devices (Part 2 of 2)

3.1.2. Uio Parameters

The available Uio parameters are required only when you want to perform an operation other than a file copying operation and when an exact copy is being performed to the same device type. If the input is a multiple keyed MIRAM disk, then the key that references the file must be specified [MKR=( )]. Table 3-1 (see 3.2.4) describes these parameters.

3.2. INPUT AND OUTPUT STATEMENT FORMATS

Different formats are available for the utility input and output (Uio) statements that describe the files and devices involved in DATA routines. These formats use variations of the same operand set, depending upon the device and access method used to copy or compare files. The general format for each statement is:

Mnemonic Portion → Keyword Parameter Portion

OPERATION△	OPERAND
Uio	ppp,ppp...ppp

Statements start in column 1 and end in or before column 71. The statements may be repeated as often as necessary; they may not be continued. If more than one card is required, do not insert a continuation character in column 72, but repeat the Uio operation in columns 1–3 of the next card.

The mnemonic portion, Uio, specifies the device type for the input (i) and output (o) files. It must be followed by a blank.

The keyword parameter portion, ppp, defines the files being used and the options that are needed. A comma follows each parameter except the last, which must be followed by a blank. Embedded blanks are not allowed except with alphanumeric literals. All keyword parameters are optional. Default values are supplied by the DATA routine.

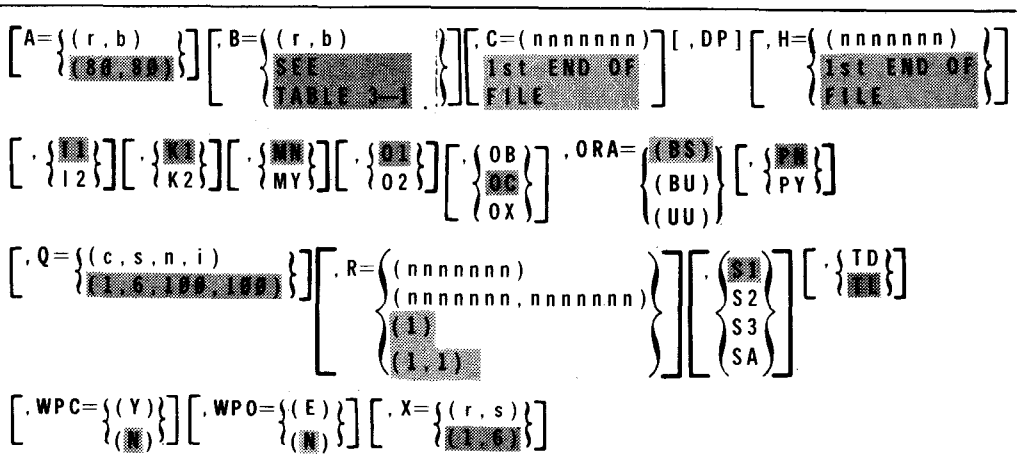
The keyword parameters required by your job must be supplied on the Uio statement and in the form specified in Table 3–1. The complete Uio statement for your job must be written in one of the formats shown on the following pages. There is a format for every combination of input and output device types. The default options are indicated by shading.

3.2.1. Card Input Formats

Diskettes are treated as card files.

3.2.1.1. Card-to-Card

- Basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UCC	

NOTE:

The ORA, WPC, and WPO parameters are not used in basic data management.

3.2.1.2. Card-to-Disk

- For SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UCD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C=(nnnnnnn) / 1st END OF FILE][,{DC / DP}][,H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / I2}][,{K1 / K2}][,{MN / MY}][,{OS / OSY}][,{OB / OC / OX}][,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,0)}]

- For ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UCD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C=(nnnnnnn) / 1st END OF FILE][,{DC / DP}][,G={(nn) / (10)}] [,H={(nnnnnnn) / 1st END OF FILE}][,{I1 / I2}][,{K1 / K2}][,{MN / MY}][,{OI / OIY}][,{OB / OC / OX}][,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}] [,V={(nnnnn) / (10)}][,W={(nnnnn) / (0)}][,X={(r,s) / (1,0)}]

- For IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UCD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / I2}][,{K1 / K2}][,{MN / MY}],{OR / ORY}={(C[,V]) / (I[,{P / 1}][,S][,V])}][,{OB / OC / OX}][,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}] [,V={(nnnnn) / (10)}][,{VS / VS=(nnnnn)}][,W={(nnnnn) / (0)}][,x={(r,s) / (1,6)}]

- For DAM files: basic and mixed data management environments

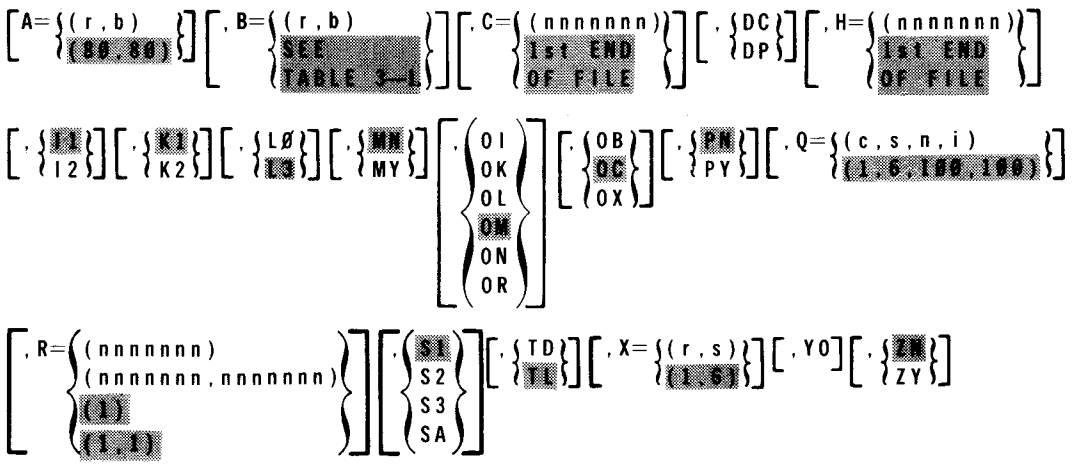
OPERATION△	OPERAND
UCD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / I2}][,{K1 / K2}][,{MN / MY}][,{OD / ODY}][,{OC / OB / OX}][,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,V={(nnnnn) / (10)}][,X={(r,s) / (1,6)}]

- For MIRAM files: basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UCD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / I2}][,{K1 / K2}][,{MN / MY}],{OM / OMY}={(C[,V][,R]) / (I[,n][,V][,R])}] [,{MK1 / MK2 / MK3 / MK4 / MK5}={{len / 10}[,{loc / 0}][,DUP][,CHG] / SEE TABLE 3-1}][,{OB / OC / OX}][,{PN / PY}] [,Q={(C,S,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}] [,{VS / VS=(nnnnn)}][,X={(r,s) / (1,6)}]

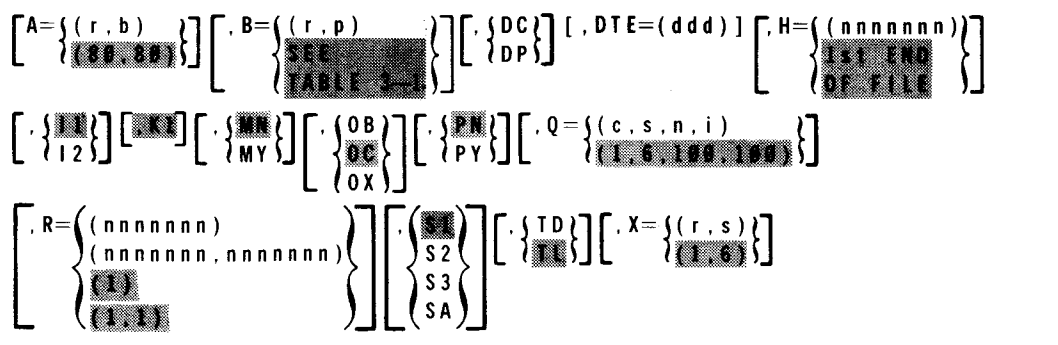
3.2.1.3. Card-to-Tape

- Basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UCT	

3.2.1.4. Card-to-Printer

- Basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UCP	

- Example 1:

```
1         10        20        30
UCT OR,Q=(2,4,1,1),R=(50),X=(2,4),ZN
```

A card reader and tape unit are the devices used in the operation designated by the UCT mnemonic. A file input in the card reader is copied into a magnetic tape output file.

- Default processing

  The default keyword parameters were omitted because their default values are automatically assigned to the job. Omitting the *A* and *B* keyword parameters specifies 80-byte record and block lengths. Fixed-length records are assumed. Omitting keyword parameters *I1* and *K1* specifies that card input is in EBCDIC and that this is a copy operation.

- Requested processing

  Options requested, via keyword parameters, to process the UCT statement include: *OR*, which specifies rewinding of output tape before and after processing; *Q*=(2,4,1,1), which specifies that sequence numbers be written on the output tape file starting in column 2, the sequence field is 4 bytes long, 1 is the first sequence number written, and that the sequence field is incremented by 1 for each record; *R*=(50), *X*=(2,4), and *ZN* specify that processing begins at logical record 50, a sequence check is made on the input file starting at column 2 for a length of 4 bytes, and leading tape marks are not written on the output file.

- Example 2:

```
1         10
UCC
```

A card reader and card punch are identified by the mnemonic UCC as the devices used. A file input in the card reader is copied into a card punch output file.

- Default processing

  Because the default values sufficiently describe the particular file and operation, all keyword parameters are omitted. The default values for the keyword parameters *A, B, I1, K1, O1,* and *R* specify: The input and output files contain 80-byte blocks and records, with input mode in EBCDIC; this is a copy operation, output mode is EBCDIC, and processing begins with the first record.

3.2.2. Tape Input Formats

3.2.2.1. Tape-to-Card

- Basic, mixed, and consolidated data management environments

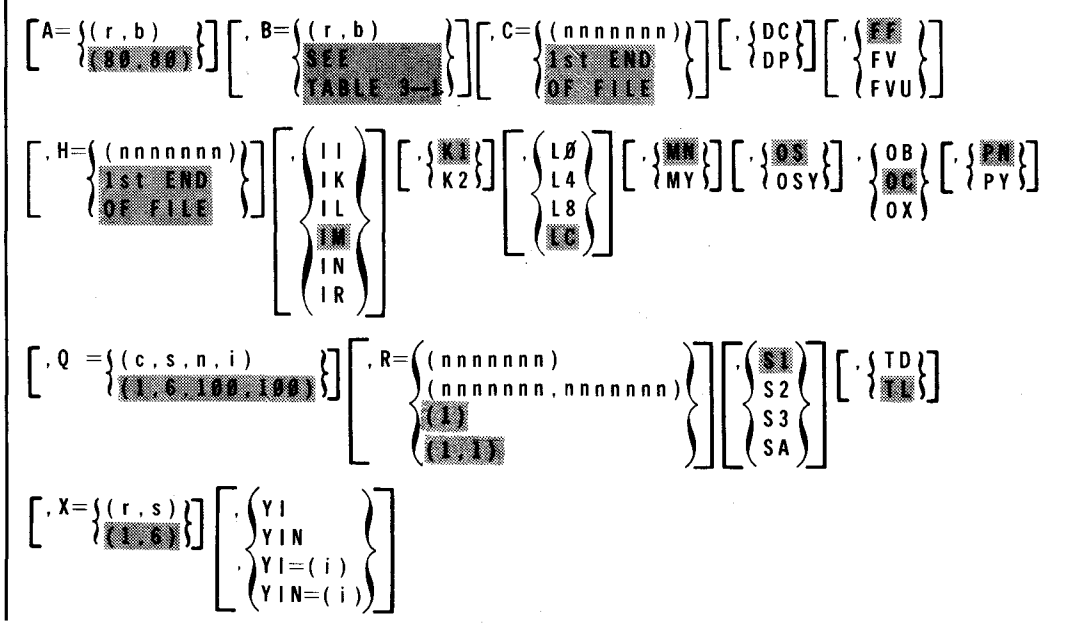
OPERATION△	OPERAND
UTC	[A={(r,b) / (80,80)}] [,B={(r,b) / FCT / TABLE-name}] [,C={(nnnnnnn) / 1st END OF FILE}] [,DP] [,H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / IK / IL / IM / IR}] [,{K1 / K2}] [,{MM / MY}] [,{L3 / L7 / LB / LF}] [,{01 / 02}] [,{OB / OC / OX}] [,ORA={(BS) / (BU) / (UU)}] [,{PR / PY}] [,Q={(c,s,n,i) / (1,0,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,WPC={(Y) / (N)}] [,WPO={(E) / (N)}] [,X={(r,s) / (1,6)}] [,{YI / YIN / YI=(i) / YIN=(i)}]

NOTE:

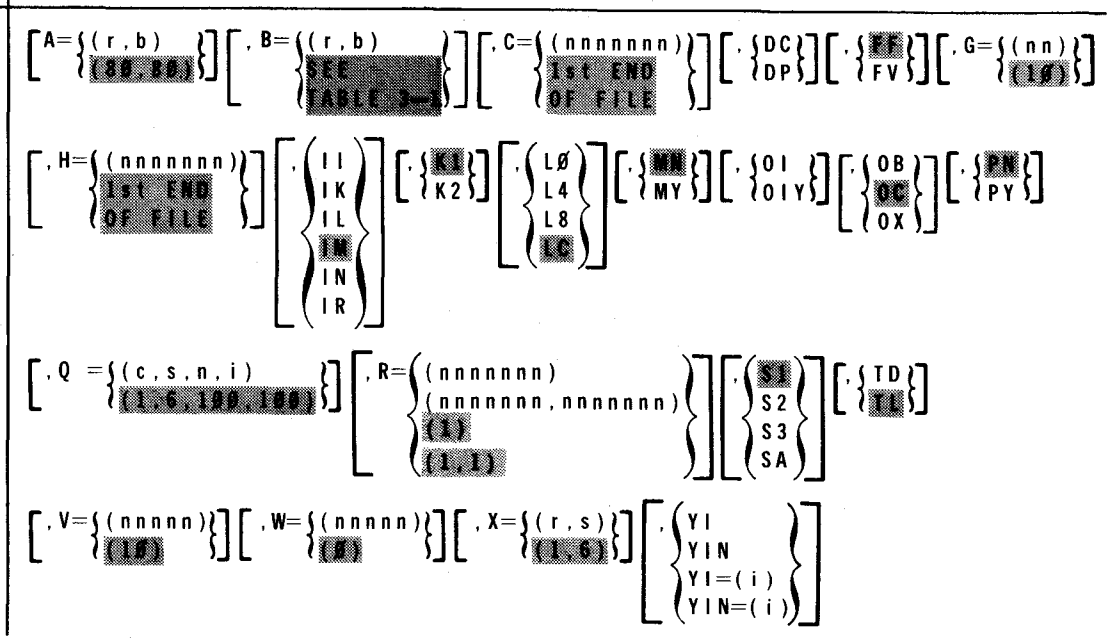
The ORA, WPC, and WPO parameters are not used in basic data management.

3.2.2.2. Tape-to-Disk

- For SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UTD	

- For ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UTD	

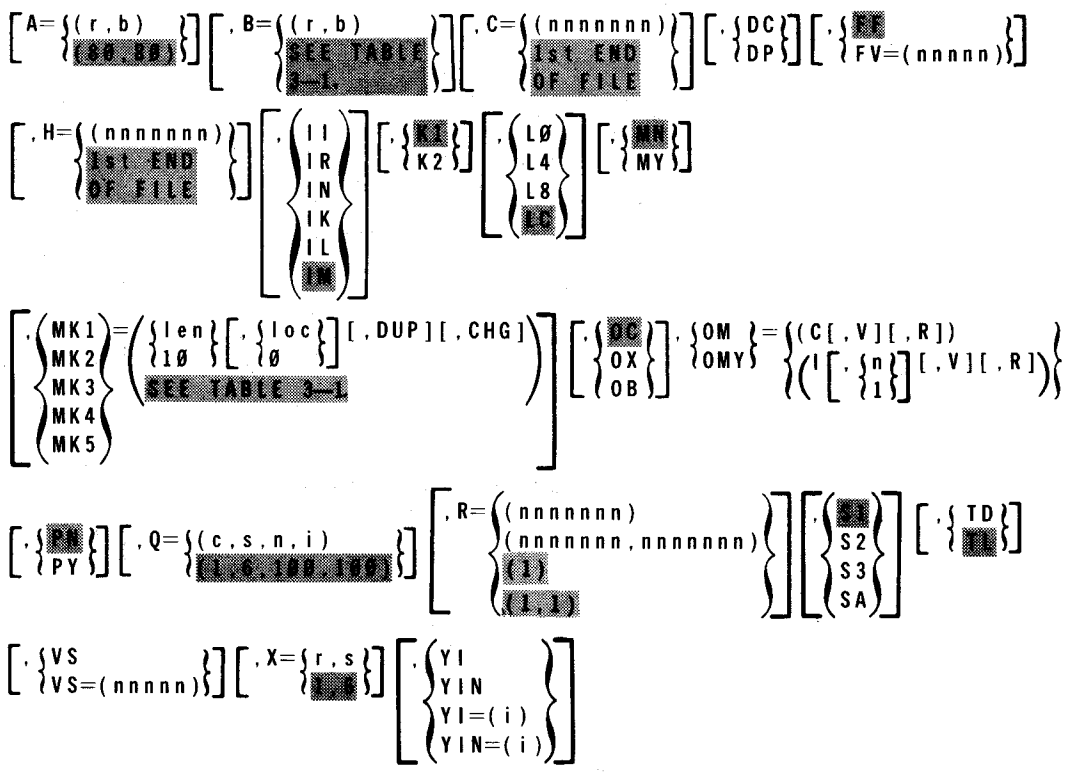
- For IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UTD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,H={(nnnnnnn) / 1st END OF FILE}] [,{I1 / IK / IL / IM / IN / IR}][,{K1 / K2}][,{LØ / L4 / L8 / LG}][,{MM / MY}][,{OR / ORY}={(C[,V]) / (I[,{P / I}][,S][,V])}][,{OB / OC / OX}] [,{PM / PY}][,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][{S1 / S2 / S3 / SA}][,{TD / TE}] [,V={(nnnnn) / (10)}][,W={(nnnnn) / (0)}][,X={(r,s) / (1,6)}][,{YI / YIN / YI=(i) / YIN=(i)}]

- For DAM files: basic and mixed data management environments

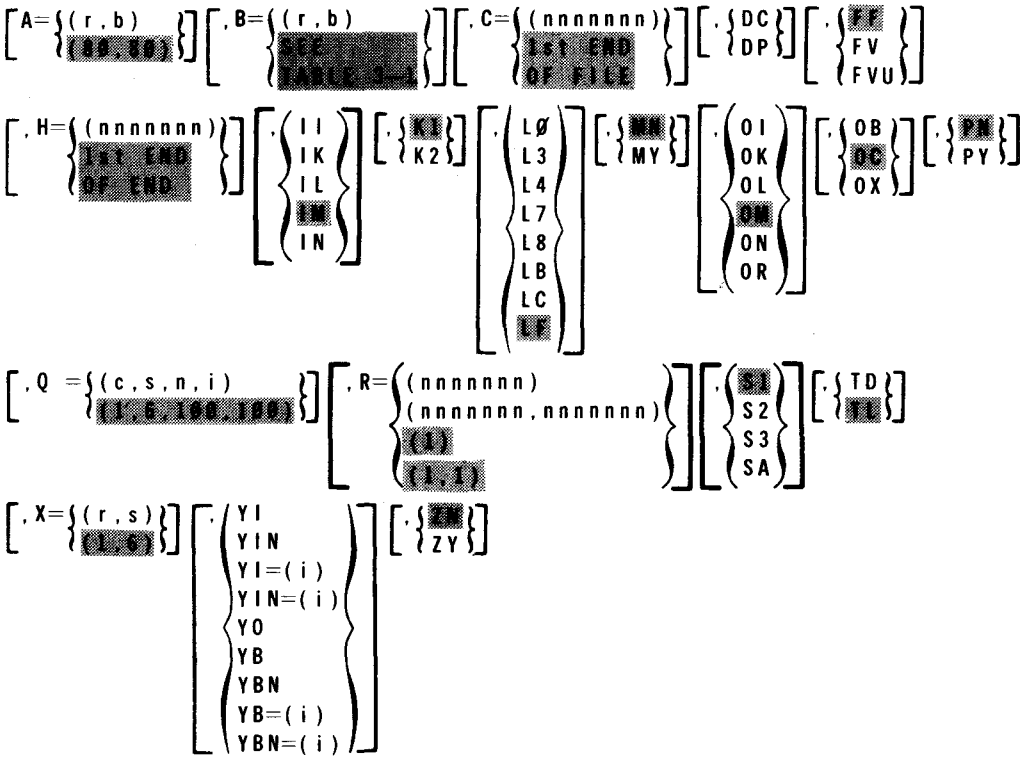
OPERATION△	OPERAND
UTD	[A={r,b / (80,80)}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV}] [,H={(nnnnnnn) / 1st END OF FILE}][,{I1 / IR / IN / IK / IL / IM}][,{K1 / K2}][,{LØ / L4 / L8 / LG}][,{MM / MY}][,{OD / ODY}][,{OC / OX / OB}] [,{PM / PY}][,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][{S1 / S2 / S3 / SA}][,{TD / TE}] [,V={(nnnnn) / (10)}][,X={r,s / (1,6)}][,{YI / YIN / YI=(i) / YIN=(i)}]

- For MIRAM files: basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UTD	

3.2.2.3. Tape-to-Tape

- Basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UTT	

3.2.2.4. Tape-to-Printer

- Basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UTP	[A={(r,b) / (80,80)}][,B={(r,p) / SEE TABLE 3-1}][,DC][,DTE=(ddd)][,{CE / FV}][,H={(nnnnnnn) / END OF FILE ON INPUT}] [,{I1 / IK / IL / IM / IN / IR}][,K1] [,{L3 / L7 / LB / LE}][,{NR / MY}][,{OB / OC / OX}][,{PN / PY}][,Q={(c,s,n,i) / (1,0,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TR}][,X={(r,s) / (1,0)}][,{YI / YIN / YI=(i) / YIN=(i)}]

- Example 1:

```
1         10
UTT
```

A tape-to-tape operation is designated by the mnemonic UTT. The data on one magnetic tape file is to be copied to another magnetic tape file.

- Default processing

  Because the default options sufficiently describe the magnetic tape files and the operation, all keyword parameters can be omitted. The default keyword parameters, *A, B, FF, IM, K1, LF, OM, R,* and *ZN* specify: the input and output record and block lengths are 80 bytes; the input tape is not rewound before or after processing; this is a copy operation; there are no input or output file labels; the output tape is not rewound after processing; processing begins with the first logical record; and leading tape marks are not written on the output file.

- Example 2:

```
1         10        20        30        40
UTT A=(80,90),C=(500),IL,K2,OR,R=(5,5)
```

A tape-to-tape operation is designated by the mnemonic UTT. Two magnetic tape files are compared and the unequal records are printed.

- Default processing

  The *B* keyword parameter is omitted. By default it is assumed that the second input file block and record lengths are the same as the first input file specified by *A*=(80,90). Omitting the *LF* keyword parameter indicates that the input and output files are unlabeled. By omitting the *ZN* keyword parameter, tape marks are not written on the output tape.

- Requested processing

  Input record lengths are 80 bytes long, and input block lengths are 90 bytes long as specified by the *A* keyword parameter. The *C* keyword parameter specifies that 500 unequal records are accepted and printed before job termination. The *IL* keyword parameter specifies that the first input tape is rewound before processing and rewound with interlock after processing. *K2* specifies a compare operation; *OR* specifies the second input tape is rewound before and after processing. The *R* keyword parameter specifies that processing begins with the fifth logical record on both input tapes.

3.2.3. Disk Input Formats

3.2.3.1. Disk-to-Card

- For SAM, DAM, ISAM and IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDC	[B={(r,b) / SEE TABLE 3-4}][,C={(nnnnnnn) / 1ST END OF FILE}][,DP][,H={(nnnnnnn) / 1ST END OF FILE}][,{K1 / K2}][,{MY}] [,{01 / 02}][,{OB / OC / OX}][,ORA={(BS) / (BU) / (UU)}][{PW / PY}][,Q={(c,s,n,i) / (I,C,IRC,IRF)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][{S1 / S2 / S3 / SA}][{TD / TE}][,WPC={(Y) / (N)}][,WPO={(E) / (N)}] [,X={(r,s) / (E,E)}]

NOTE:

The ORA, WPC, and WPO parameters are not used in basic data management.

- For MIRAM files: basic, mixed, and consolidated data management environments

OPERATION△	OPERATION
UDC	[B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,DP][,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,MKR={(n) / (0)}] [,{MN / MY}][,{01 / 02}][,{OC / OX / OB}][,ORA={(BS) / (BU) / (UU)}][,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{SI / S2 / S3 / SA}][,{TD / TL}][,WPC={(Y) / (N)}][,WPO={(E) / (N)}][,X={(r,s) / (1,6)}]

NOTE:

The ORA, WPC, and WPO parameters are not used in basic data management.

3.2.3.2. Disk-to-Disk

- For SAM-to-SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV / FVU / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MN / MY}],{DS / OSY}[,{OB / OC / OX}][,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{SI / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,6)}]

- For DAM-to-DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MN / MY}],{OD / ODY}[,{OB / OC / OX}][,{PN / PY}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{SI / S2 / S3 / SA}][,{TD / TL}][,V={(nnnnn) / SEE TABLE 3-1}][,X={(r,s) / (1,6)}]

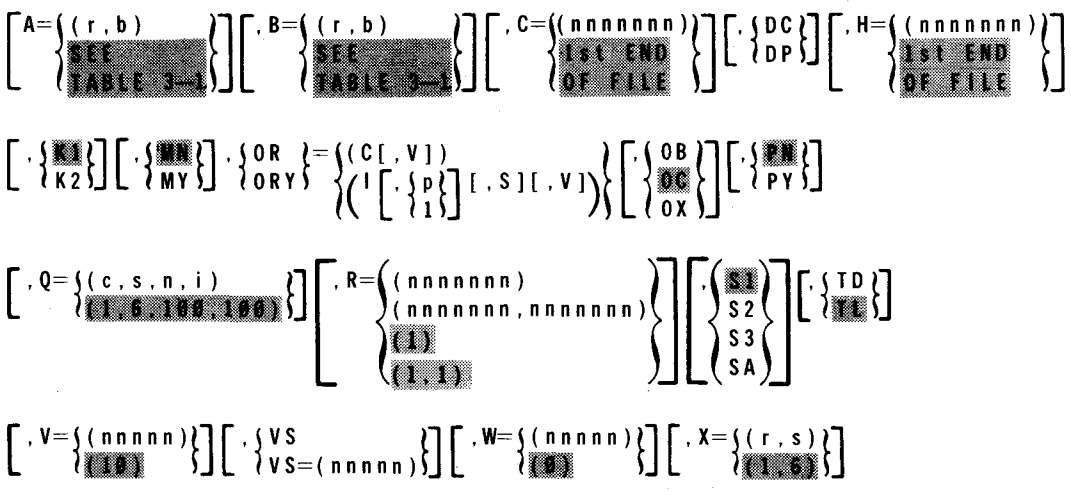
- For DAM-to-SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV / FVU / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MM / MY}],{OS / OSY}[,{OB / OC / OX}][,{PM / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,6)}]

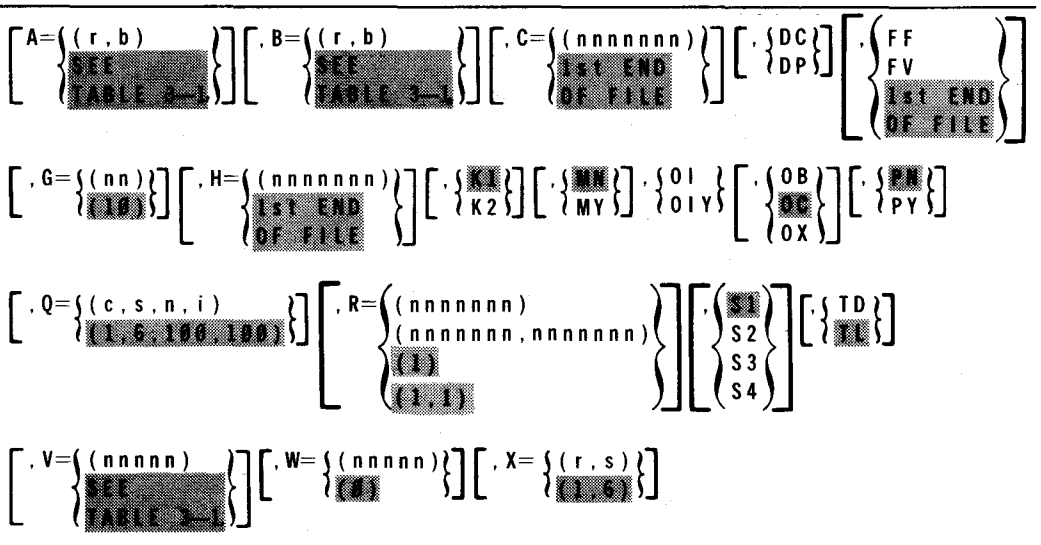
- For SAM-to-ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV / SEE TABLE 3-1}] [,G={(nn) / (10)}][,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MM / MY}],{OI / OIY}[,{OB / OC / OX}][,{PM / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}] [,V={(nnnnn) / (10)}][,W={(nnnnn) / (0)}][,X={(r,s) / (1,6)}]

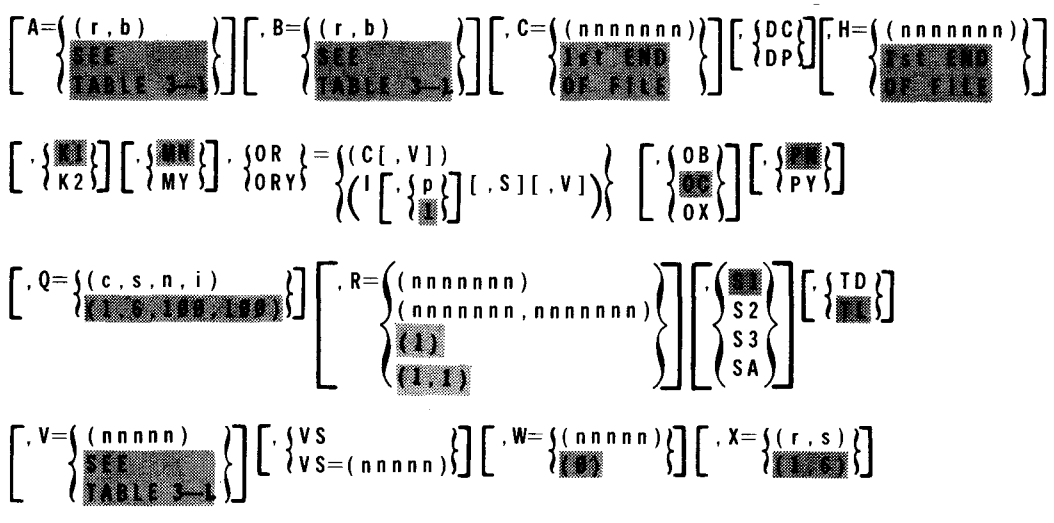
- For SAM-to-IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

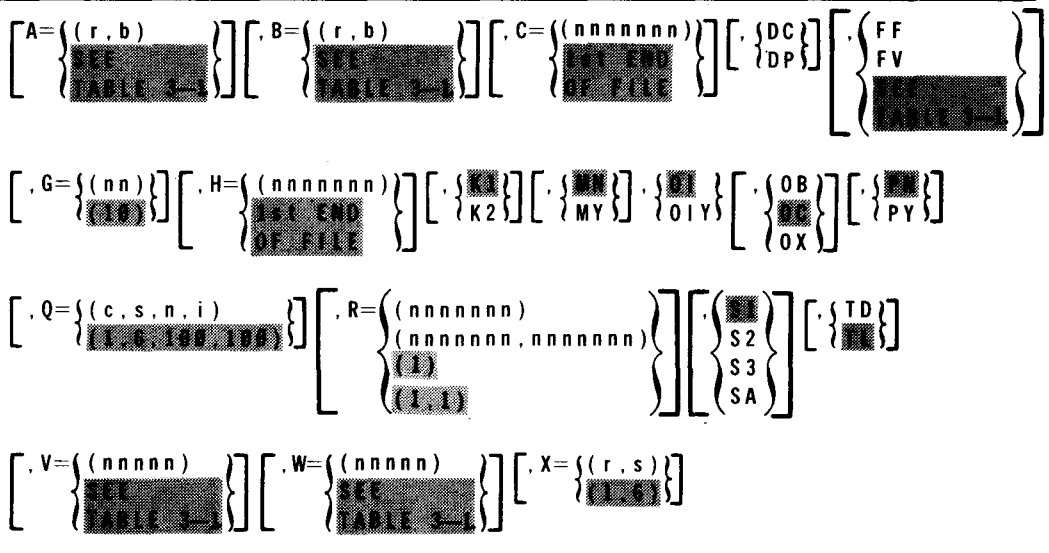
- For DAM-to-ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

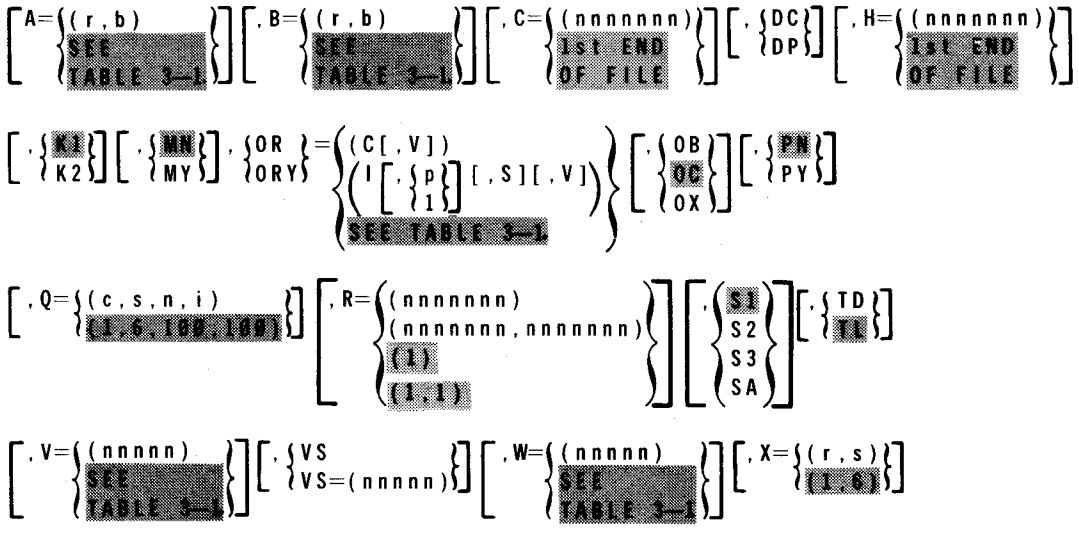
- For DAM-to-IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

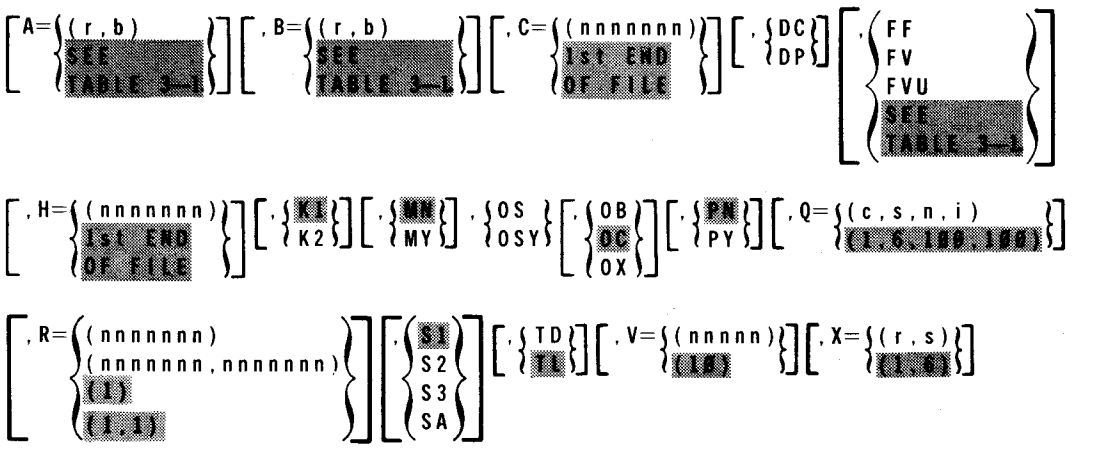
- For ISAM-to-ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

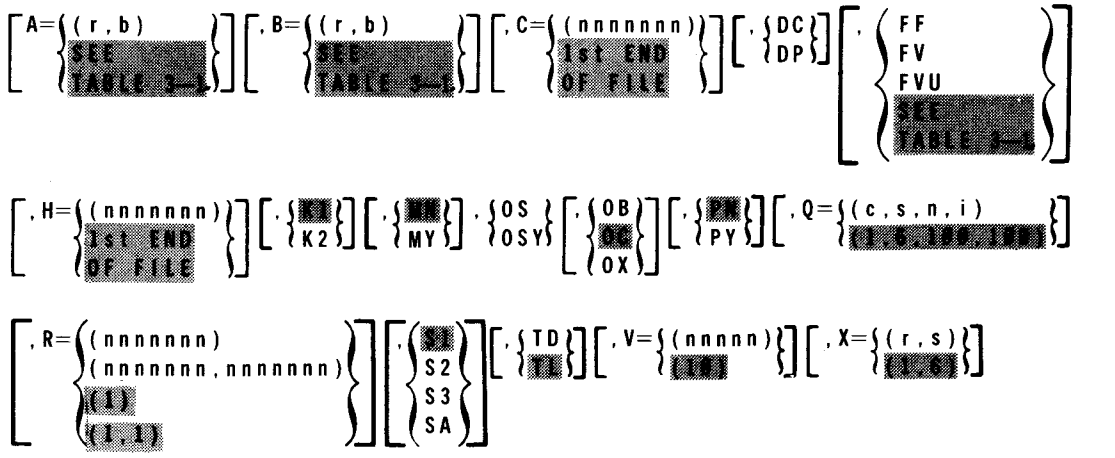
- For IRAM-to-IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

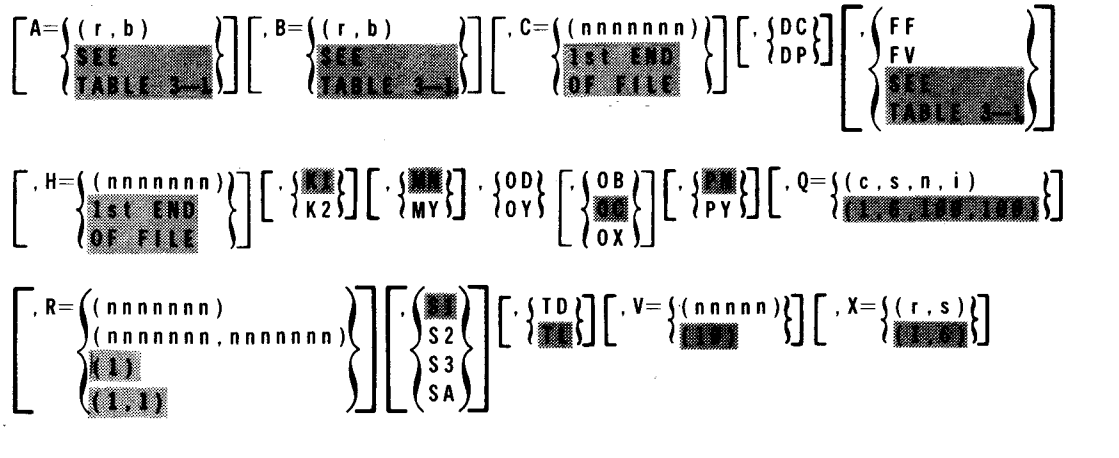
- For ISAM-to-SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

- For IRAM-to-SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

- For SAM-to-DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	

- For SAM-to-MIRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV=(nnnnn) / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MK1 / MK2 / MK3 / MK4 / MK5}=({len / 10}[,{loc / 0}][,DUP][,CHG])][,{MN / MY}] [,{OB / DC / OX}],{OM / OMY}={(C[,V][,R]) / (I[,{n / 1}][,V][,R])}[,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (I) / (I,I)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,{VS / VS=(nnnnn)}][,X={(r,s) / (1,6)}]

- For ISAM-to-DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}][,B={(r,b) / SEE TABLE 3-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}][,{FF / FV / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}][,{K1 / K2}][,{MN / MY}],{OD / ODY}[,{OB / DC / OX}][,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (I) / (I,I)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,V={(nnnnn) / SEE TABLE 3-1}][,X={(r,s) / SEE TABLE 3-1}]

- For DAM-to-MIRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) | SEE TABLE 3-1}] [,B={(r,b) | SEE TABLE 3-1}] [,C={(nnnnnnn) | 1st END OF FILE}] [,{DC | DP}] [,{FF | FV=(nnnnn) | SEE TABLE 3-1}] [,H={(nnnnnnn) | 1st END OF FILE}] [,{K1 | K2}] [,{MK1 | MK2 | MK3 | MK4 | MK5}=({[len | 10][,{loc | 0}][,DUP][,CHG]} | SEE TABLE 3-1)] [,{MN | MY}] [,{OB | OC | OX}],{OM | OMY}={(C[,V][,R]) | (I[,{n | 1}][,V][,R])} [,{PN | PY}] [,Q={(c,s,n,i) | (1,6,100,100)}] [,R={(nnnnnnn) | (nnnnnnn,nnnnnnn) | (I) | (1,1)}] [,{S1 | S2 | S3 | SA}] [,{TD | TL}] [,{VS | VS=(nnnnn)}] [,X={(r,s) | (1,6)}]

- For ISAM-to-IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) | SEE TABLE 3-1}] [,B={(r,b) | SEE TABLE 3-1}] [,C={(nnnnnnn) | 1st END OF FILE}] [,{DC | DP}] [,H={(nnnnnnn) | 1st END OF FILE}] [,{K1 | K2}] [,{MN | MY}] [,{OB | OC | OX}],{OR | ORY}={(C[,V][,R]) | (I[,p][,S][,V])} [,{PN | PY}] [,Q={(c,s,n,i) | (1,6,100,100)}] [,R={(nnnnnnn) | (nnnnnnn,nnnnnnn) | (I) | (1,1)}] [,{S1 | S2 | S3 | SA}] [,{TD | TL}] [,V={(nnnnn) | SEE TABLE 3-1}] [,{VS | VS=(nnnnn)}] [,W={(nnnnn) | SEE TABLE 3-1}] ,X={(r,s) | (1,6)}

- For ISAM-to-MIRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A=((r,b) / SEE TABLE 3-1.)] [,B=((r,b) / SEE TABLE 3-1.)] [,C=(nnnnnnn) / 1st END OF FILE] [,DC / DP] [,FF / FV=(nnnnn) / SEE TABLE 3-1.] [,H=(nnnnnnn) / 1st END OF FILE] [,K1 / K2] [,(MK1 / MK2 / MK3 / MK4 / MK5)=(len / 1Ø [,loc / Ø] [,DUP][,CHG] / SEE TABLE 3-1.)] [,MN / MY] [,OB / OC / OX],OM / OMY=((C[,V][,R]) / (I[,n / *][,V][,R])) [,PN / PY] [,Q=(c,s,n,i) / (1,6,100,100)] [,R=(nnnnnnn / nnnnnnn,nnnnnnn / (1) / (1,1))] [,(S1 / S2 / S3 / SA)] [,TD / TL] [,VS / VS=(nnnnn)] [,X=(r,s) / (1,6)]

- For IRAM-to-DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A=((r,b) / SEE TABLE 3-1.)] [,B=((r,b) / SEE TABLE 3-1.)] [,C=(nnnnnnn) / 1st END OF FILE] [,DC / DP] [,FF / FV=(nnnnn) / SEE TABLE 3-1.] [,H=(nnnnnnn) / 1st END OF FILE] [,K1 / K2] [,MN / MY] [,OB / OC / OX],OD / OY [,PN / PY] [,Q=(c,s,n,i) / (1,6,100,100)] [,R=(nnnnnnn / nnnnnnn,nnnnnnn / (1) / (1,1))] [,(S1 / S2 / S3 / SA)] [,TD / TL] [,V=(nnnnn) / SEE TABLE 3-1.] [,X=(r,s) / (1,6)]

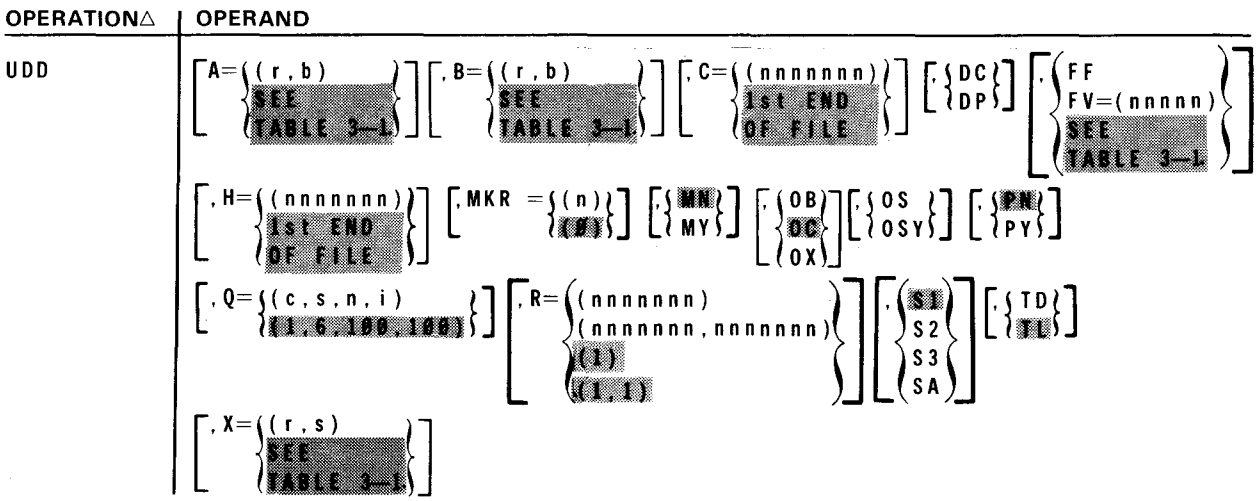
- For IRAM-to-ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) | SEE TABLE 3-1}][,B={(r,b) | SEE TABLE 3-1}][,C={(nnnnnnn) | 1st END OF FILE}][,{DC | DP}][,{FF | FV | SEE TABLE 3-1}] [,G={(nn) | (10)}][,H={(nnnnnnn) | 1st END OF FILE}][,{K1 | K2}][,{MN | MY}][,{OI | OIY}][,{OB | OC | OX}][,{PN | PY}] [,Q={(c,s,n,i) | (1,6,100,100)}][,R={(nnnnnnn) | (nnnnnnn,nnnnnnn) | (1) | (1,1)}][,{S1 | S2 | S3 | SA}][,{TD | TL}] [,V={(nnnnn) | SEE TABLE 3-1}][,W={(nnnnn) | SEE TABLE 3-1}][,X={(r,s) | (1,6)}]

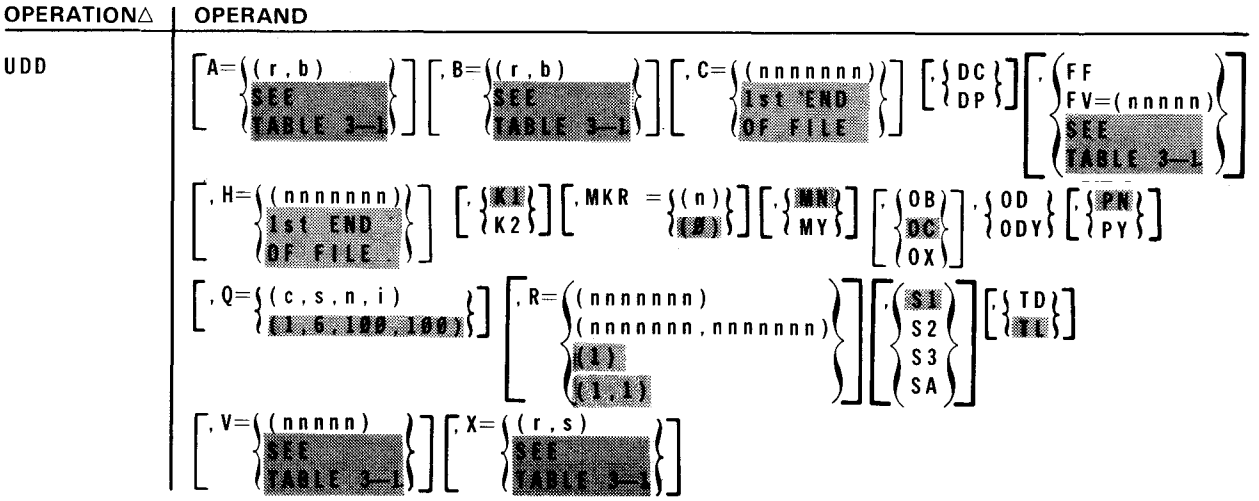
- For IRAM-to-MIRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) | SEE TABLE 3-1}][,B={(r,b) | SEE TABLE 3-1}][,C={(nnnnnnn) | 1st END OF FILE}][,{DC | DP}][,{FF | FV=(nnnnn) | SEE TABLE 3-1}] [,H={(nnnnnnn) | 1st END OF FILE}][,{K1 | K2}][,{MK1 | MK2 | MK3 | MK4 | MK5}=[{len | 10}][,{loc | 0}][,DUP][,CHG]][,{MN | MY}] [,{OB | OC | OX}],{OM | OMY}={(C[,V][,R]) | (I,{n | 1}[,V][,R])}[,{PN | PY}][,Q={(c,s,n,i) | (1,6,100,100)}] [,R={(nnnnnnn) | (nnnnnnn,nnnnnnn) | (1) | (1,1)}][,{S1 | S2 | S3 | SA}][,{TD | TL}][,{VS | VS=(nnnnn)}][,X={(r,s) | (1,6)}]

- For MIRAM-to-SAM files: basic and mixed data management environments



- For MIRAM-to-DAM files: basic and mixed data management environments



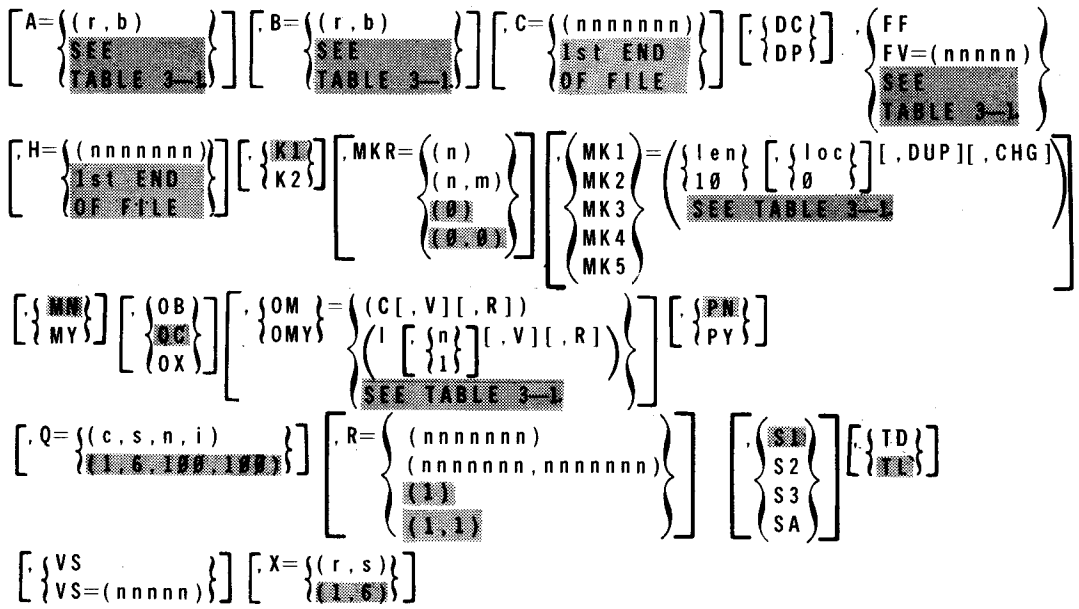
- For MIRAM-to-ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,{FF / FV / SEE TABLE 3-1}] [,G={(nn) / (10)}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,MKR={(n) / (0)}] [,{MN / MY}] [,{OB / OC / OX}] [,{OI / OIY}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,V={(nnnnn) / SEE TABLE 3-1}] [,W={(nnnnn) / SEE TABLE 3-1}] [,X={(r,s) / (1,6)}]

- For MIRAM-to-IRAM files: basic and mixed data management environments

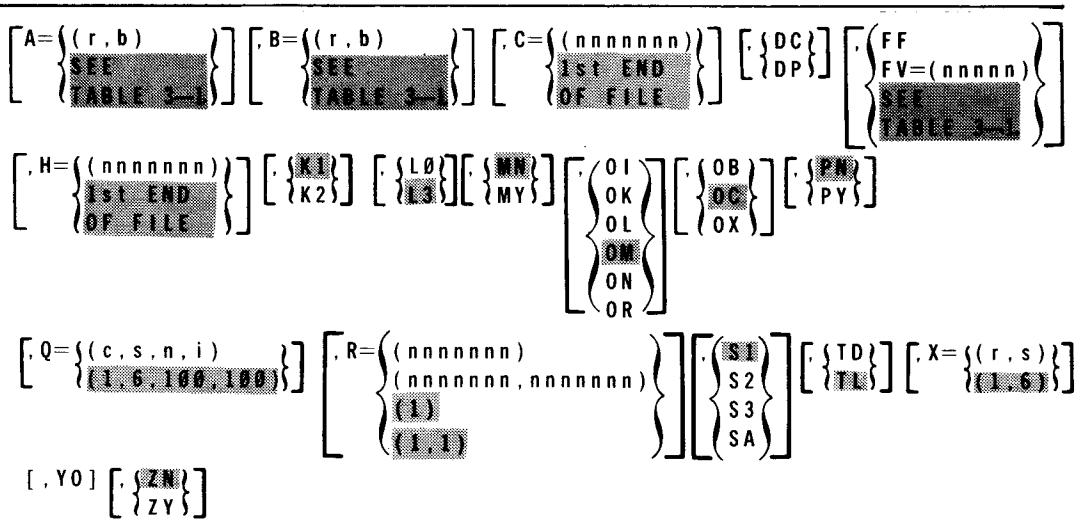
OPERATION△	OPERAND
UDD	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,MKR={(n) / (0)}] [,{MN / MY}] [,{OB / OC / OX}] [,{OR / ORY}={(C[,V]) / (I[,{P / 1}][,S][,V])}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,V={(nnnnn) / SEE TABLE 3-1}] [,{VS / VS=(nnnnn)}] [,W={(nnnnn) / SEE TABLE 3-1}] [,X={(r,s) / (1,6)}]

- For MIRAM-to-MIRAM files: basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UDD	

3.2.3.3. Disk-to-Tape

- For SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDT	

- For DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDT	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,{FF / FV=(nnnnn) / FVU / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,{LØ / L3}] [,{MN / MY}] [,{OI / OK / OL / OM / ON / OR}] [,{OB / OC / OX}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,X={(r,s) / (1,6)}] [,YO] [,{ZN / ZY}]

- For ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDT	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,{FF / FV=(nnnnn) / FVU / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,{LØ / L3}] [,{MN / MY}] [,{OI / OK / OL / OM / ON / OR}] [,{OB / OC / OX}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,X={(nnnnn) / (1,6)}] [,YO] [,{ZN / ZY}]

- For IRAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDT	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,{LØ / L3}] [,{MN / MY}] [,{OI / OK / OL / OM / ON / OR}] [,{OB / OC / OX}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,X={(r,s) / (1,6)}] [,YO] [,{ZN / ZY}]

- For MIRAM files: basic, mixed, and consolidated data management environments

OPERATION△	OPERAND
UDT	[A={(r,b) / SEE TABLE 3-1}] [,B={(r,b) / SEE TABLE 3-1}] [,C={(nnnnnnn) / 1st END OF FILE}] [,{DC / DP}] [,{FF / FV / FVU / SEE TABLE 3-1}] [,H={(nnnnnnn) / 1st END OF FILE}] [,{K1 / K2}] [,{LØ / L3}] [,MKR={(n) / (0)}] [,{MN / MY}] [,{OB / OC / OX}] [,{OI / OK / OL / OM / ON / OR}] [,{PN / PY}] [,Q={(c,s,n,i) / (1,6,100,100)}] [,R={(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [,{S1 / S2 / S3 / SA}] [,{TD / TL}] [,X={(r,s) / (1,6)}] [,YO] [,{ZN / ZY}]

3.2.3.4. Disk-to-Printer

- For SAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDP	[A={(r,b) / SEE TABLE 3-1}][,DC][,DTE=(ddd)][,H={(nnnnnnn) / END OF FILE ON INPUT1}][,KI][,{MN / MY}][,{OB / OC / OX}] [,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,6)}]

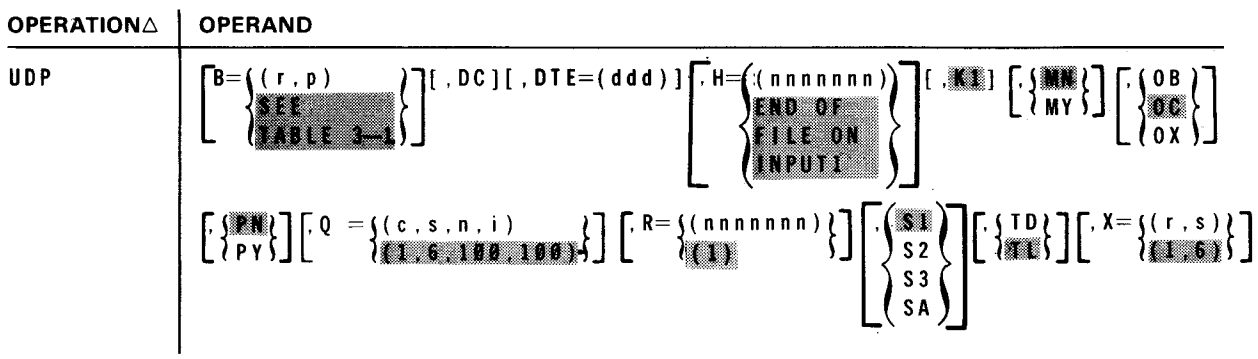
- For DAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDP	[B={(r,p) / SEE TABLE 3-1}][,DC][,DTE=(ddd)][,H={(nnnnnnn) / END OF FILE ON INPUT1}][,KI][,{MN / MY}][,{OB / OC / OX}] [,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,6)}]

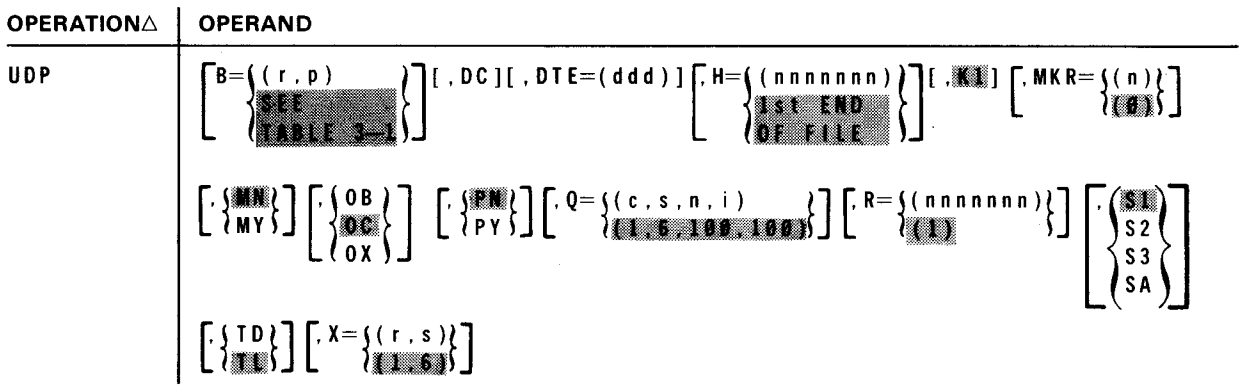
- For ISAM files: basic and mixed data management environments

OPERATION△	OPERAND
UDP	[B={(r,p) / SEE TABLE 3-1}][,DC][,DTE=(ddd)][,H={(nnnnnnn) / END OF FILE ON INPUT1}][,KI][,{MN / MY}][,{OB / OC / OX}] [,{PN / PY}][,Q={(c,s,n,i) / (1,6,100,100)}][,R={(nnnnnnn) / (1)}][,{S1 / S2 / S3 / SA}][,{TD / TL}][,X={(r,s) / (1,6)}]

- For IRAM files: basic and mixed data management environments



- For MIRAM files: basic, mixed, and consolidated data management environments



- Example 1:

```
1         10        20        30
UDC C=(10),H=(1000),K2,R=(10,10)
```

The mnemonic UDC specifies a disk-to-card operation. An input disk file in SAM format is compared with the card file entered from the card reader. The unequal records are printed.

- Default processing

  All disk input file characteristics are taken from information stored in the VTOC. These include the length of the input records and blocks, the fixed-length record format, and the SAM disk file. Because the *O1* keyword parameter is omitted, EBCDIC is assumed. The length of the output records and blocks is equal to the length of the input records and blocks.

- Requested processing

  The requested keyword parameters specify the following: a maximum of 10 unequal records are accepted and printed; DATA routine is terminated after comparing 1000 records (this is a compare operation); and begin processing at the tenth record in the disk and card files.

- Example 2:

```
1         10        20        30        40        50
UDC  B=(80,80),DP,H=(1000),Q=(73,8,1,5),R=(10),TD
```

A disk-to-card operation is specified by the UDC mnemonic. A disk file in DAM format is output at the card punch.

- Default processing

  The copy operation and EBCDIC are specified by omitting the keyword parameters *K1* and *O2*, respectively.

- Requested processing

  Taken in the order shown in the example, the keyword parameters specify: Output file record and block lengths are 80 bytes. Each punched record is printed. Up to 1000 input records are processed before the DATA routine terminates; sequence numbers are punched on each record of the output file. The sequence field starts in column 73, field length is 8 bytes, and sequence numbers start at 1 and are incremented by 5. Processing begins at the 10th record. Print is in display format.

- Example 3:

```
1         10        20        30        40
UDT  K2,C=(100),R=(5),Q=(1,,000000)
```

The mnemonic UDT specifies a disk-to-tape operation. A disk file in ISAM format is compared with a tape file.

- Default processing

  All disk input file characteristics are taken from information stored in the VTOC. These include the input and output file blocks and record length, the fixed-length record format, and the ISAM disk file. The default values for the *H, L3,* and *OM* keyword parameters specify that the entire file is to be processed, the tape has no output labels, and the output tape is not rewound before or after processing.

- Requested processing

  The keyword parameters specify that: This is a compare operation. A total of 100 unequal records are accepted and printed before the DATA routine terminates. Processing begins with logical record 5. Sequence numbering is to be performed where the sequence field begins in column 1, the size of the field is 6 (by default), the sequence number of the first record is 000000, and the increment is 100 (by default).

- Example 4:

```
1         10
UDD
```

The mnemonic UDD specifies a disk-to-disk operation. Because no parameters are specified, the default values are used.

- Default processing

  All disk input file characteristics are taken from information stored in the VTOC. These include input and output file block and record lengths, fixed-length records, the key length, and the key location. By default, both printing and punching are suppressed. This is a copy operation. The output file type is the same as the input file type. No sequence checking is performed and processing begins with the first logical record. Both the input and output are not in ASCII.

3.2.4. Input and Output Statement Keyword Parameters

To use the DATA routine, you should be thoroughly familiar with the keyword parameters listed in Table 3–1. They are listed in alphabetic order with a description of the function and default option for each.

NOTE:

MIRAM and IRAM files must have the volume mount indicator set to MULTI if the file is to be processed randomly later.

The following restrictions apply to IRAM format files only:

- Key length may not be greater than 80 bytes (*V* parameter).
- Records may not be of variable length.
- Duplicate keys are not allowed.
- Multikey files are not supported.

NOTE:

If any of the above requirements are not met, the job step will be terminated and an error message will be printed.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 1 of 11)

Keyword Parameters	Description
A=(r,b)	INPUT1 Block and Record Length For fixed-length (FF) records on 8413 card or tape files, this parameter specifies record length (r) and block length (b) expressed in decimal. If omitted, r and b both default to 80. The block length b must be a multiple of r. For variable-length (FV) records on tape input files, this parameter specifies a minimum r and a maximum b. If omitted, the minimum r defaults to 0 and the maximum b defaults to 80. For fixed-length (FF) records on SAM, DAM, or ISAM disk input files, this parameter is not used. The r and b lengths are taken from the VTOC entry for the INPUT1 file. If this parameter is specified for SAM, DAM, or ISAM files, no error message will be generated but the information specified in this parameter will be ignored. If DAM input is specified in the preceding case, the r represents the record data length; the key length is not included in r. If SAM input is specified, then b must be a multiple of r. For fixed-length (FF) records on IRAM or MIRAM files, this parameter specifies a dummy value less than six digits long (r) and the buffer length (b) expressed in decimal (b must be a multiple of sector size). If the specified buffer length is less than the minimum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum will be used. For variable-length (FV) records on SAM, DAM, or ISAM files, this parameter specifies a minimum r and a dummy b value less than six digits long. The block length will be taken from the VTOC entry for INPUT1. For variable-length (FV) records on IRAM or MIRAM files, this parameter specifies a minimum r and the buffer length (b must be a multiple of sector size). If the specified b is less than the minimum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum is used. All subparameters to this parameter have a maximum size limit of 5 decimal digits.
B=(r,b)	OUTPUT1/INPUT2 Block and Record Length For fixed-length (FF) records on 8414 card tape and SAM disk files, this parameter specifies record length (r) and block length (b). For these files, b should be an even multiple of r. For fixed-length (FF) records on DAM disk files, this parameter specifies r, which excludes key length, and b, which is data length plus key length. For fixed-length (FF) records on ISAM disk files, this parameter specifies r, which includes key length, and b, which includes ISAM block overhead of two bytes per block plus five bytes per record. For fixed-length (FF) records on IRAM/MIRAM disk files, this parameter specifies r, which includes key lengths but excludes MIRAM record control byte, and buffer length (b), which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for fixed-length (FF) records, r defaults to the INPUT1 file record length (from the A=( ) parameter for nondisk input or the file format labels for a disk input file). Block/buffer length (b) defaults to the INPUT1 file block/buffer length (from the A=( ) parameter for nondisk and IRAM/MIRAM input files or the file format labels for all other disk input files).

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 2 of 11)

Keyword Parameters	Description
B=(r,b)	For variable-length (FV) records on tape and SAM disk files, this parameter specifies minimum record length (r), which includes a 4-byte variable record descriptor word, and block length (b), which must be at least the maximum record plus four bytes for the variable block descriptor word. For variable-length (FV) records on DAM disk files, this parameter specifies minimum r, which includes the 4-byte variable record descriptor word but excludes key length and b, which is maximum record length plus key length plus four bytes for the variable block descriptor word. For variable-length (FV) records on ISAM disk files, this parameter specifies minimum r, which includes key length plus two bytes for the ISAM variable record discriptor word and b, which is at least maximum record size plus seven bytes for ISAM block overhead. For variable-length (FV) records on MIRAM disk files, this parameter specifies minimum r, which includes all keys plus the 4-byte variable record descriptor word, and b, which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for variable-length (FV) records, the minimum r defaults to zero and the block/buffer length b defaults to the INPUT1 file block/buffer length. If comparing (K2) files and the INPUT2 file is a disk file, the record length for fixed record (FF) files and the block length for fixed (FF) and variable (FV) record files will be obtained from the disk file format labels and override the information in this parameter. For IRAM and MIRAM files, buffer size may be specified here. If it isn't, the minimum allowed is used. This is a performance consideration.
B=(r,p)	For fixed-length (FF) records on the primary printer file (UiP), this parameter specifies the record length (r), and the print line size (p). If this parameter is omitted in this case, r will default to the INPUT1 record length, and a 120-character print line size (p) will be assumed. If r is greater than the record length of the INPUT1 file, the OUTPUT1 record will be padded on the right with blanks. If the specified r is less than the record length of the INPUT1 file, the INPUT1 record length will be used. If first character forms control (SA) is specified, print line size (p) must include one byte for the control character. For variable-length (FV) records on the primary printer file (UiP), this parameter specifies a minimum and a print line size (p). The minimum r cannot be less than 2. If this parameter is omitted in this case, the minimum r will default to the value used for the INPUT1 file, and a 120-character print line size will be assumed.
C=(nnnnnnn)	Compare Error Check This parameter can not be specified if copy (K1) is specified. It indicates the number (a decimal value represented by nnnnnnn) of unequal records accepted and printed prior to termination, which occurs when nnnnnnn+1 unequal records are found. If omitted, all unequal records are printed, and the job terminates on the first end-of-file.
DC	Dual Output File This parameter requests the creation of a dual card OUTPUT2 file. The first 80 bytes of each OUTPUT1 record are punched. If the record length of the OUTPUT1 file is less than 80, the OUTPUT2 record will be blank filled on the right. The device assigned to the OUTPUT2 file can be an 80-column card punch, a 96-column card punch, or an 8413 diskette. The OUTPUT2 records will be 80 bytes long regardless of the I/O device assigned to this file. This parameter can not be specified if file compare (K2) is specified, or if variable-length record (FV) is specified, or if the OUTPUT1 file is a card file (UiC).

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 3 of 11)

Keyword Parameters	Description
DP	This parameter requests the creation of a dual printer output file. Each record sent to the OUTPUT1 file is printed in its entirety. The print line size is fixed at 120 for this file. This parameter cannot be specified if file compare (K2), correction (COR), or primary printer file (UiP) is specified.
DTE=(ddd)	Print Date on Output Listing This parameter prints the date on all output listings. The ddd specifies YMD in any order, where Y is the year, M is the month, and D is the day.
	Record Format
FF	Specifies fixed-length records
FV	Specifies variable-length records. This parameter cannot be specified if any of the files assigned to the job are 8413 card files or card files.
FVU	Specifies variable-length records, with unblocked records in the OUTPUT1 file, for SAM disk and tape files only. If this is not specified and variable-length records (FV) are specified, then all files except DAM disk files will be variable length blocked.
FV=(nnnnn)	Specifies variable-length records for MIRAM files where: n Specifies the output file slot size for the MIRAM OUTPUT1 file.
	The FF AND FV parameters are mutually exclusive. If both of these parameters are omitted and the INPUT1 file is a disk file (UDo), then the record format of the INPUT1 file is used. If both of these parameters are omitted, and the INPUT1 file is not a disk file, then fixed length (FF) is assumed. NOTE: Files having fixed-length records cannot be copied to files having variable-length records and vice versa.
G=(nn)	ISAM Percent Cylinder Overflow This parameter specifies the percentage of a cylinder (nn) reserved for cylinder overflow for ISAM output files. The maximum percentage of a cylinder that can be reserved for overflow is 80 percent. If this parameter is omitted, 10 percent is assumed.
	If zero percent cylinder overflow is used to create the file, then the file can never be updated (i.e., new records can never be inserted in the file).
H=(nnnnnnn)	Halt on Record Count Specifies the number of records to be processed before terminating. nnnnnnn represents a global count of all records in the file. If R=( ) is specified, the number of records to be skipped before processing is included in nnnnnnn for this parameter. The job terminates at first end-of-file.
	EBCDIC/Column Binary
I1	Specifies that the card INPUT1 file is in EBCDIC

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 4 of 11)

Keyword Parameters	Description
I2	Indicates the card INPUT1 file is in column binary code. If this parameter is specified, then the INPUT1 record length, as specified in the first entry of the A=( ) parameter, should be twice as long as the number of characters in the INPUT1 record (column binary code requires two bytes to represent each character). If both of these parameters are omitted, EBCDIC (I1) is assumed. If these parameters are specified, the INPUT1 file must be assigned to a card reader (the INPUT1 file cannot be an 8413 diskette file).
	Tape Input Rewind
I1	Rewind before and rewind interlock after processing
IK	No rewind before and rewind after processing
IL	No rewind before and rewind with interlock after processing
IM	No rewind before and no rewind after processing
IN	Rewind before and no rewind after processing
IR	Rewind before and after processing
	If none of these parameters are specified, the INPUT1 file is not rewound before or after processing and IM is assumed
	Copy and Compare
K1	This parameter requests a file copy operation. The INPUT1 file is copied to the OUTPUT1 file.
K2	This parameter requests a file compare operation. The INPUT1 file is compared to the INPUT2 file, and unequal records are printed. If this parameter is specified, a printer must be assigned to the job, but none of the printer file parameters can be specified. This parameter can not be specified if correction (COR) or if dual output (DP or DC) are specified.
	If both of these parameters are omitted, file copy (K1) is assumed.
	Tape Label
L0	Indicates standard labeled INPUT1 and standard labeled OUTPUT1/INPUT2
L3	Indicates standard labeled INPUT1 and unlabeled OUTPUT1/INPUT2
L4	Indicates user labeled INPUT1 and standard labeled OUTPUT1/INPUT2
L7	Indicates user labeled INPUT1 and unlabeled OUTPUT1/INPUT2
L8	Indicates standard and user labeled INPUT1 and standard labeled OUTPUT1/INPUT2
LB	Indicates standard and user labeled INPUT1 and unlabeled OUTPUT1/INPUT2
LC	Indicates unlabeled INPUT1 and standard labeled OUTPUT1/INPUT2
LF	Indicates unlabeled INPUT1 and unlabeled OUTPUT1/INPUT2
	If all of these parameters are omitted, unlabeled INPUT1 and unlabeled OUTPUT1/INPUT2 (LF) are assumed.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 5 of 11)

Keyword Parameters	Description
	MIRAM Input Key
MKR=(n)	Specifies the key of reference to be used for this access of the MIRAM INPUT1 file, where n is a 1-digit decimal value between 0 and 5.
MKR=(n,m)	Specifies the key of reference to be used for this access of the MIRAM INPUT1 file (n) and the key of reference to be used for this access of the MIRAM INPUT2 file (m). Both n and m are 1-digit decimal values between 0 and 5. If 0 is specified in either of these parameters, the MIRAM file can be either indexed or consecutive, but consecutive reads will be performed. If a nonzero value is specified in either of the above parameters, the MIRAM file must be indexed, and a read by key will be performed by using key (n) for INPUT1 and key (m) for INPUT2 as specified during the creation of INPUT1 and INPUT2, respectively.
	If these parameters are omitted, a read by relative record number will be performed.
MKn=(len[,loc][,DUP][,CHG])	MIRAM Output Key Specifies one of up to five MIRAM output keys for an indexed or mixed MIRAM OUTPUT1 file, where: n Is a 1-digit number between 1 and 5 that specifies the MIRAM key number. len Is a 2-digit number less than or equal to 80 that specifies the length of this MIRAM key. loc Is a 5-digit number less than 32767 that specifies the key location, relative to zero, of this MIRAM key. This is the number of bytes that precedes the key. If this is omitted, zero will be assumed. DUP Specifies that this key can have duplicates. If this is omitted, then no duplicates are allowed. CHG Specifies this key can be changed. If this is omitted, no changes to this key are allowed. The keys specified by this parameter are the only keys associated with this file. Any subsequent access by key must reference one of the keys defined by this parameter. If MIRAM disk input and output are specified (UDD, the OM parameter with valid specifications, and INPUT1 is a MIRAM file), and the parameter is omitted, the MIRAM OUTPUT1 file will be created with the same keys as specified in the VTOC entry for the MIRAM INPUT1 file. If ISAM or IRAM disk input is specified (UDD, ISAM, or IRAM INPUT1), MIRAM disk output is specified (OM parameter with valid specifications), and the parameter is omitted. The MIRAM OUTPUT1 file is created with a single key equal to the ISAM or IRAM key as specified in the INPUT1 VTOC entry. If MIRAM disk output is specified (UiD and the OM parameter with valid specifications), and the INPUT1 file is not ISAM, IRAM, or MIRAM, the MIRAM OUTPUT1 file is created with a single key 10 bytes long starting in the first byte of the record.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 6 of 11)

Keyword Parameters	Description
	Printer mismatch
MN	Do not ignore printer mismatches.
MY	Ignore printer mismatches.
	If both of these parameters are omitted, MN is assumed.
	These parameters cannot be specified if compare (K2) or correction (COR) is specified.
	Output EBCDIC/Column Binary
01	Indicates the OUTPUT1/INPUT2 card file is in EBCDIC
02	Indicates the OUTPUT1/INPUT2 card file is in column binary code. If this parameter is specified, the OUTPUT1/INPUT2 record length, as specified in the first entry of the B=( ) parameter, should be twice as long as the number of characters in the OUTPUT1/INPUT2 record (i.e., column binary code requires two bytes to represent each character).
	If both of these parameters are omitted, then EBCDIC (01) is assumed. If these parameters are specified, the OUTPUT1/INPUT2 file must be a card file (the OUTPUT1/INPUT2 file can not be an 8413 diskette file).
	Tape Output Rewind
OI	Rewind before and rewind interlock after processing
OK	No rewind before and rewind after processing
OL	No rewind before and rewind with interlock after processing
OM	No rewind before and no rewind after processing
ON	Rewind before and no rewind after processing
OR	Rewind before and rewind after processing
	If none of the parameters are specified, the OUTPUT1/INPUT2 tape is not rewound before or after processing and OM is assumed.
	Output Disk File Type
OI	Indicates an ISAM OUTPUT1/INPUT2 disk file. If copy (K1) is specified, then the file is created without write verify active.
OIY	Indicates an ISAM OUTPUT1/INPUT2 disk file. If copy (K1) is specified, then the file is created with write verify active.
OS	Indicates a SAM OUTPUT1/INPUT2 disk file. If file copy (K1) is specified, the OUTPUT1 is created without write verify.
OSY	Indicates a SAM OUTPUT1/INPUT2 disk file. If file copy (K1) is specified, the OUTPUT1 is created with write verify.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 7 of 11)

Keyword Parameters	Description
OR=(C[,V])	Indicates an IRAM OUTPUT1/INPUT2 disk file is to be created without write verify active. When this parameter is specified, the first subparameter must also be specified; all other subparameters are optional. These may or may not have commas present to indicate omitted parameters. The subparameters for OR=( ) are as follows: C Indicates a consecutive IRAM file and the file has no INDEX partition. If C is specified, the "p" and "S" entries must not be specified in the OR=( ) parameter. I Indicates the output file is to be created with an index partition. p Specifies the index block size for this file. It is a 1-digit decimal number from 1 through 7 representing the number of 256-byte multiples in the index block. The index block size used here stands for the life of the file. The default value is 1. S Indicates a sequence check on key values is performed during the creation of this file. If not specified, no sequence check will be performed. V Provides IRAM/MIRAM file generation as either single or multivolume mount. If specified, the volume mount indicator is set so that data management requires all volumes of the file to be online at the same time. Although a multivolume mount file is the only type that can be processed randomly, the initial file allocation can never be extended. On the other hand, a single-volume mount file cannot be processed randomly, but can be extended. This also applies to a file occupying only one volume. If this parameter is omitted and input is disk, the default is the input file single/multivolume mount indicator. If input is card, tape, or diskette, single-volume mount output is assumed.
OD	Indicates a DAM OUTPUT1/INPUT2 disk file is to be created. If copy (K1) is specified, the OUTPUT1 file will be created without write verify.
ODY	Indicates a DAM OUTPUT1/INPUT2 disk file is to be created. If copy (K1) is specified, OUTPUT1 file will be created with write verify.
OM=(C[,V][,R]) or OM=(I[,n][,V][,R])	Specifies MIRAM disk OUTPUT1 with no write verify, where: C Indicates a consecutive file. I Indicates an indexed file. n Is a 1-digit decimal number from 1 through 7 specifying the number of 256-byte sectors the index buffer must accommodate. The default value is 1.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 8 of 11)

Keyword Parameters	Description
	V Indicates multivolume mount indicator is to be set (if this file is ever presented to job control as a multivolume file, then all volumes of the file must be online at the same time). If omitted, the file will be created with the mount indicator set to single-mount (if this file is ever presented to job control as a multivolume file, then each volume of the file must be mounted one-at-a-time in sequence on the same disk drive). R Indicates each record in the file is not to contain a record control byte.
OMY=(C[,V][,R]) or OMY=(I[,n][,V][,R]) or OMY=([,n][,V][,R])	Specifies MIRAM disk OUTPUT1 with write verify, where C, I, n, V, and R have the same meaning as they do for the OM=( ) parameter. If all of these parameters are omitted, a default will be assigned as follows: For a file compare (K2), where disk output was specified (UiD), the disk file type as specified in the VTOC entry for the INPUT2 file is used. For a file copy (K1), where disk input and disk output were specified (UDD), and none of the preceding parameters were specified, the OUTPUT1 disk file type will default to the INPUT1 disk file type. For a file copy (K1), where either card or tape input and disk output were specified (UCD or UTD), and none of the preceding parameters were specified, the OUTPUT1 disk file type will default to SAM.
ORY=(C[,V]) or ORY=(I[,p][,S][,V])	Indicates an IRAM OUTPUT1/INPUT2 disk file is to be created. If copy (K1) is specified, the OUTPUT1 file will be created with write verify active. C, I, p, S, and V have the same function as they have in the OR=( ) parameter.
	Printer Mode
OB	Specifies output is printed in both EBCDIC and hexadecimal code
OC	Specifies output is printed in EBCDIC
OX	Specifies output is printed in hexadecimal code
	If all of the these parameters are omitted, EBCDIC (OC) is assumed.
	Output Record Attribute for Diskette
ORA=(BS)	Specifies that the output diskette records are to be blocked spanned
ORA=(BU)	Specifies that the output diskette records are to be blocked unspanned
ORA=(UU)	Specifies that the output diskette records are to be unblocked unspanned
	This parameter is used only in mixed and consolidated data management. If the ORA parameter is omitted, BS is assumed; however, for fixed-length records, if sector size is 128 and record size is equal to or less than 128, the default is UU.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 9 of 11)

Keyword Parameters	Description
	Printer Numbering
PN	Specifies page number and date are not to be printed
PY	Specifies page number and date are to be printed on the first line of each page. This parameter cannot be specified if first character forms control (SA) is specified.
	If both PN and PY are omitted, PN is assumed.
	Sequence Numbering
Q=(c,s,n,i)	Indicates the following sequence numbering is to be done: c Is the column relative to column 1, where the sequence field begins. If omitted, 1 is assumed. s Is the size of the sequence field. If omitted, 6 is assumed. n Is the sequence number of the first record to be written (up to 10 decimal digits with or without leading zeros). If omitted, 100 is assumed. i Is the sequencing increment. If omitted, 100 is assumed.
	If the entire parameter is omitted, sequence numbering is not performed.
	File Repositioning
R=(nnnnnnn)	Indicates processing is to begin with logical record number nnnnnnn
R=(nnnnnnnn,nnnnnnn)	For compare (K2) only. Specifies where processing begins in INPUT1 and INPUT2, respectively
	The number of records skipped, as specified in this parameter, is included in the record count for the H=( ) parameter, if it is specified.
	If R=( ) is omitted, processing begins with the first logical input record.
	Printer Spacing
S1	Single space after printing
S2	Double space after printing
S3	Triple space after printing
SA	The first character of each input record is used for forms control. This option is not allowed if the page numbering (PY) parameter is specified. If this parameter is specified, the device independent control characters must be used. See the basic data management user guide, UP-8068 (current version) for details. If the SA parameter is specified, then TL and OC are assumed for print format and mode. The SA parameter is not allowed if a file compare (K2) is specified.
	These parameters cannot be specified if correction (COR) or compare (K2) is specified.
	If these parameters are omitted, single spacing (S1) is assumed.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 10 of 11)

Keyword Parameters	Description
	Printer Format
TD	Print in display format
TL	Print in list format
	If TD and TL are omitted, list format (TL) is assumed. These parameters cannot be specified if correction (COR) or compare (K2) is specified.
	Key Length
V=(nnnnn)	For a file copy (K1), this specifies the key length (less than 6 decimal digits) for the OUTPUT1 (UiD) disk file. If this parameter is omitted and the INPUT1 file is a disk file (UDD and K1 are specified), the INPUT1 key length specified in the INPUT1 VTOC entry is used. Otherwise, a 10-byte key is assumed. If file compare (K2) is specified, the key length is taken from the VTOC entry for the INPUT2 file. Since the key lengths for all input disk files are taken from the VTOC entries for the corresponding files, no input disk key length parameter is provided. Data management supports keys only for disk files.
V=(mmmmm,nnnnn)	This form of the parameter also is accepted. It has the same function as the V=(nnnnn) parameter except that mmmmm is a dummy entry (less than 6 decimal digits) that is always ignored. The second entry, nnnnn, has the same function and defaults as in the V=(nnnnn) parameter.
	Variable Sector Size
VS	Specifies that the data partition for an IRAM or MIRAM output file is to be written by using a sector size other than 256 bytes (illegal for 8416/8418 disk drives). Also, it specifies that the DATA routine is to compute the sector size as being equal to the highest even multiple of the slot size that fits into the buffer size as specified by the second entry in the B=( ) parameter. See the basic data management user guide, UP-8068 (current version) for an explanation of slot size.
VS=(nnnnn)	Specifies that DATA routine is to use value entered for nnnnn as sector size
	If both of these parameters are omitted, the sector size is 256 bytes.
	Key Location (DAM/ISAM/IRAM Files)
W=(nnnnn)	For a copy operation (K1), this specifies the key location (the number of bytes that precede the key) for the OUTPUT1 file (UiD). If this parameter is omitted and the INPUT1 file is a disk file (UDD and K1 are specified), the INPUT1 key location as specified in the INPUT1 VTOC entry is used. Otherwise, a key location of 0 is used for fixed-length (FF) files, a key location of 2 is used for ISAM variable (OI and FV specified) files, and a key location of 4 is used for all other variable-length disk files. For file compare (K2), the key location is taken from the VTOC entry for the INPUT2 file.
W=(mmmmm,nnnnn)	This form of the parameter also is accepted to indicate the key location. It has the same function as the W=(nnnnn) parameter, where mmmmm is a dummy entry that is always ignored. The second entry, nnnnn, has the same function and defaults as in the W=(nnnnn) parameter.
	Write Protect of Output Diskette on Close
WPC=(N)	Specifies that the write protect option is to remain as specified
WPC=(Y)	Specifies that the diskette be marked as write protected
	If omitted, N is assumed. This parameter is used only in mixed and consolidated data management environments.

Table 3—1. Keyword Parameters for Utility Input and Output Statements (Part 11 of 11)

Keyword Parameters	Description
	Write Protect of Output Diskette on Open
WPO=(E)	Erase the write protect.
WPO=(N)	Leave the write protect as specified.
	If you don't erase the write protect mark at open, the DATA routine cannot write to the file.
	If omitted, N is assumed. This parameter is used only in mixed and consolidated data management environments.
	Sequence Checking
X=(r,s)	Specifies sequence checking, where r specifies the first column of the sequence field (relative to column 1), and s is the size of the sequence field (maximum value is 40).
	If no column number is specified, 1 is assumed; if no size is specified, 6 is assumed.
	The record is checked for a sequence w field that is higher than the corresponding field in the preceding record. When a sequence error occurs, an error message is logged, processing continues, and the new lower number is used as the new base for subsequent checking. If this parameter is omitted, sequence checking is not performed.
	ASCII Tape
YI	Indicates the INPUT1 tape file (UTo), will be in ASCII format. For fixed length records (FF), a buffer offset length of 0 is used. For variable length records (FV), a buffer offset length of 4 is used. A data management length check is performed using the contents of the buffer offset, which must contain the block length in decimal.
	Only variable-length ASCII output tapes are supported by data management, so only variable format is available for data utilities to create.
YIN	Identical to YI except that no length check is made for variable records
YI=(i)	Identical to YI except that (i) is treated as a decimal value representing buffer offset length. For fixed format records, the buffer offset is ignored. For variable format records, the first four bytes are assumed to be the record length in decimal, and the rest of the buffer offset is ignored.
YIN=(i)	Identical to YI=(i), except that no length check is made for variable records
	If any of the preceding parameters are omitted, EBCDIC input is assumed.
YB	Combines functions of parameters YI (ASCII input) and YO
YBN	Combines functions of parameters YIN and YO
YB=(i)	Combines functions of parameters YI=(i) and YO
YBN=(i)	Combines functions of parameters YIN=(i) and YO
YO	Specifies ASCII output for variable-length record format; 4-byte buffer offset record length is written in decimal.
	If all of the previous parameters are omitted, EBCDIC output is assumed.
	Tape mark
ZN	Do not write tape mark on unlabeled output tape.
ZY	Write leading tape mark on unlabeled output tape.
	If both of these parameters are omitted, ZN is assumed.

3.3. MODIFIER STATEMENTS

You can modify DATA routine utility input and output (Uio) statements by using utility modifier statements:

- FS statement (field select)

  Move selected fields within a record.

- SEL/DEL statement (select/delete)

  Select or delete records.

- Hn statement (title)

  Print page headings.

- COR statement (correction)

  Insert, delete, or replace records.

- PAR statement (partition)

  Specify information required for processing nonindexed multiple-partitioned files.

If you use any of these optional statements, they must appear in your job control stream following the Uio statement. If no utility modifier statements are used, the records are copied or compared without change.

The modifier statements FS, SEL/DEL, COR, and PAR have the same general format as the Uio statement (3.2). The statements are written starting in column 1 and ending in or before column 71. The keyword parameter portion (ppp) defines the operation and the files. Coding follows the same rules as for the Uio statement. The title (Hn) statement uses the format described in 3.3.3. The five modifier statements are described in 3.3.1 through 3.3.5.

3.3.1. Field Select Statement - Moving or Deleting Input Record Fields

The field select (FS) statement specifies the fields in the input records that are to be rearranged, omitted, or converted when copied from the input data record to an output record. The input fields must be within fixed-length records or within the fixed portion of a variable-length record.

Fields may be moved to the same or different portions of the output record or deleted entirely from the output record. Fields in the output record not selected to receive data are blank filled (X'40'). A maximum of 50 formatting options may be specified to move a record within the limits of available space required to store the information in main storage. If the output is to the printer, the list format must be used. (See Figure 2-6.)

The selected fields may be:

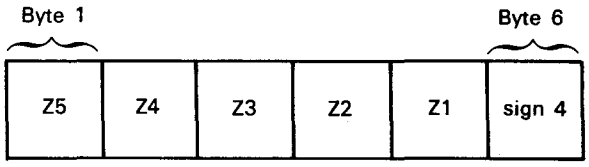
- moved without change;
- converted from zoned to packed decimal and then moved;
- converted from packed to zoned decimal and then moved;
- converted to hexadecimal for printer output and then moved;
- converted but remaining in the same location on the output record; or
- deleted.

Disk input key fields may be selected and transferred to output key fields or to data fields. However, the selected field must be contained entirely within either the disk key field or the data field.

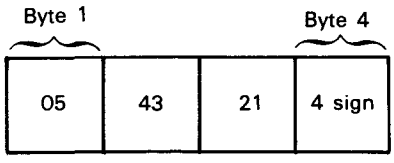
When converting a field from zoned to packed decimal, or packed to zoned decimal, the output field will be smaller or larger than the input field, depending on the conversion you specify. The resulting field-selected record may not exceed the output record size, the output block size, or the printer line size as defined in your Uio statement. The rules that apply to the output field size are:

- Zoned to packed decimal conversion

  When the input field contains an even number of bytes, as in:



  The resulting output field is:

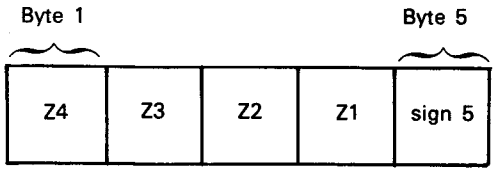


  To compute the size of the output field, apply the formula:

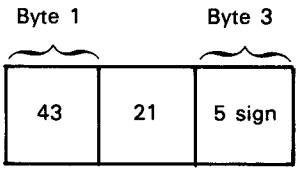
  $x=(n\div 2)+1$

  where *n* is the size of the input field and *x* is the size of the output field.

When the input field contains an odd number of bytes, as in:



The resulting output field is:



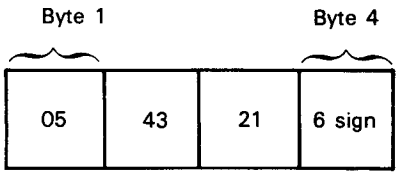
To compute the size of the output field, apply the formula:

$$x=(n+1)\div 2$$

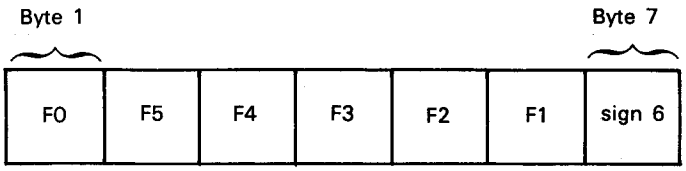
where *n* is the size of the input field and *x* is the size of the output field.

- Packed to zoned decimal conversion

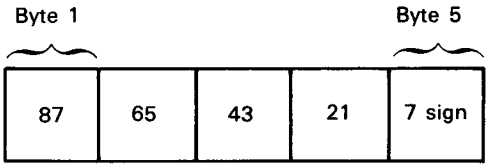
When the input field contains an even number of bytes, as in:



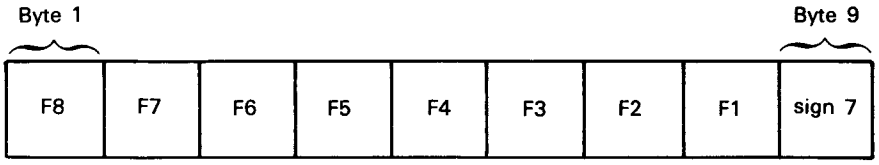
The resulting output field is:



When the input field contains an odd number of bytes, as in:



The resulting output field is:



Whether the input field contains an even or odd number of bytes, the size of the output field is computed by the formula:

$x=2n-1$

where *n* is the size of the input field and *x* is the size of the output field.

- Convert from EBCDIC character mode to hexadecimal

  The size of the output field is twice the size of the input field.

  Example:

  EBCDIC input field:

  0 1 2 3 4

  Hexadecimal output field:

  F0F1F2F3F4

In the field select statement, all numbers are expressed in decimal and the specified starting position of the selected field is relative to the first character of the record (position 1). Commas separate definitions within each selected field; slashes separate the selected fields. Selected fields must be contained entirely on one card. Also, parentheses must be coded as shown.

There are two basic formats for the field select statement: one for fixed-length records and one for variable-length records.

3.3.1.1. FS Statement Format for Fixed-Length Records

OPERATION△	OPERAND
FS	a,b,c[,d]/a,b,c[,d]/.../a,b,c[,d]

Each group of subparameters in the operand defines the field of a fixed-length input record and is specified in its output order. The subparameters are positional within each group.

a

Specifies the starting position of the fixed-length input record field. The two options for this subparameter are:

$$\left\{\begin{matrix} r \\ (K,r) \end{matrix}\right\}$$

where:

r

Is a decimal number specifying the starting position of the input record field, or the starting position of the key field.

K

Indicates that the input record field referenced is within the key field of a record in a DAM file.

b

Specifies the length of the field being copied or compared. The four options for this subparameter are:

$$\left\{\begin{matrix} s \\ (P,n,m) \\ (U,n,m) \\ (X,n) \end{matrix}\right\}$$

where:

s

Is a decimal number specifying the size of both the input and output field in bytes; this implies that no conversion is to take place.

P

Indicates that the contents of the input field are to be converted from zoned decimal to packed decimal before being transferred to the output file. The input field must be in EBCDIC.

U

Indicates that the contents of the input field are to be converted from packed decimal to zoned decimal (unpacked) before being transferred to the output file. The input field must be packed decimal.

X

Indicates that the contents of the input field are to be converted to hexadecimal before being transferred to the printer output file. The size of the output field must be twice the size of the input field. The input field must be packed decimal.

n

Is a decimal number specifying the byte size of the input field.

m

Is a decimal number specifying the byte size of the output field.

c

Specifies the starting position of the fixed-length output record field. The two options for this subparameter are:

$$\left\{\begin{matrix} t \\ (K,t) \end{matrix}\right\}$$

where:

t

Is a decimal number specifying the starting field position of the output record or the starting position of the key field.

K

Indicates that the output record referenced is within the key field of a record in a DAM file.

d

Is an optional parameter specifying that unpacked output fields are to reflect their sign. The four options are:

P

Print a plus (+) if the output field is positive.

M

Print a minus (-) if the output field is negative.

B

Print a plus (+) or minus (-) depending on whether the field is positive or negative.

N

Print no sign.

If omitted, N is assumed.

3.3.1.2. FS Statement Format for Variable-Length Records

For variable-length records, the following rules for field selection apply:

- Only that portion of the input record that is fixed in size and is always present may be selected.
- The record length field is generated in the first four bytes of each output record. Selected fields may not be transferred into these four bytes.
- When the variable portion of a variable-length record is to be copied to output, the letters CV (copy variable) must be specified within the FS statement. The variable portion of the record is placed in the output record following the fixed portion of the record.

A maximum of 50 formatting options may be included. The definitions for the a, b, c, and d subparameters are given in 3.3.1.1. The various positions for the CV specification field are as follows:

OPERATION△	OPERAND
FS	CV/a,b,c[,d]/a,b,c[,d]/a,b,c[,d]

This format is used when the variable portion of the input record is to be moved to the output record before the selected fields are moved.

OPERATION△	OPERAND
FS	a,b,c[,d]/a,b,c[,d]/a,b,c[,d]/CV

This format is used when the variable portion of the input record is to be moved to the output record after selected fields have been moved.

OPERATION△	OPERAND
FS	a,b,c[,d]/CV/a,b,c[,d]/a,b,c[,d]

This format is used when the variable portion of the input record is to be moved to the output record between the moves of the selected fields.

3.3.1.3. FS Statement Examples

The following examples illustrate the use of the FS statement for both fixed-length and variable-length records.

Example 1 – Pack fixed-length records:

Assume that an input field record contains eight fields. The fields numbered 1, 2, 7, and 4 are to be moved, in that order, to the output area; fields numbered 2 and 7 are to be packed while being moved. The fields are:

Number	Field Name	Positions
1	Name	1-15
2	Hourly rate	16-20
3	Number of dependents	21-22
4	Earnings to date	23-30
5	Address	31-66
6	Date of service	67-71
7	Hours worked	72-74
8	Weekly earnings	75-80

The FS statement is:

```
1         10        20        30        40        50
FS 1,15,1/16,(P,5,3),16/72,(P,3,2),19/23,8,21
```

The resulting output record is:

Number	Field Name	Positions
1	Name	1-15
2	Hourly rate	16-18
3	Hours worked	19-20
4	Earnings to date	21-28

Example 2 - Unpack and convert fixed-length records to hexadecimal:

Assume that an input file record contains five fields. Fields numbered 1, 3, 4, and 5 are to be moved to output. Field 1 is to be unpacked. Field 3 is to be converted to hexadecimal. Fields 4 and 5 are to be moved without change. The input fields are:

Number	Field Name	Positions
1	Life number, packed format	1-5
2	Number of dependents	15-20
3	Name	21-40
4	Hours worked	62-65
5	Weekly earnings	66-70

The FS statement is:

```
1         10        20        30        40        50
FS 1,(U,5,9),1/21,(X,20),21/62,4,62/66,5,66
```

The resulting output record is:

Number	Field Name	Positions
1	Life number, unpacked format	1-9
2	Name, in hexadecimal	21-60
3	Hours worked	62-65
4	Weekly earnings	66-70

Example 3 – Reorder and move key fields of fixed-length records on disk:

Assume that an input file contains nine data fields and a disk key field. The first field is the key field; its contents are to be transferred to the output key field. Fields 2, 5, 6, 7, and 8 are to be dropped. Fields 3, 4, 9, and 10 in that order are to be transferred to the output record. Fields 4 and 9 are to be packed while being moved. The input fields are:

Number	Field Name	Positions
1	Social security number	1–10 (Key field)
2	Department number	1–5
3	Name	6–20
4	Hourly rate	21–25
5	Number of dependents	26–27
6	Earnings to date	28–35
7	Address	36–71
8	Date of service	72–76
9	Hours worked	77–79
10	Weekly earnings	80–85

The FS statement is:

```
1         10        20        30        40        50        60
FS (K,1),1Ø,(K,1)/6,15,1/21,(P,5,3),16/77,(P,3,2),19/8Ø,6,21
```

The resulting output record is:

Number	Field Name	Positions
1	Social security number	1–10 (Key field)
2	Name	1–15
3	Hourly rate	16–18
4	Hours worked	19–20
5	Weekly earnings	21–26

Example 4 – Interchange fixed portion and copy variable portion of variable-length records:

Assume that an input file contains variable-length records. The minimum length of a logical record is 24 bytes, and the maximum block length is 300 bytes. The fixed portion of the logical record is defined as 24 bytes consisting of two 10-byte fields and the 4-byte record-length field. The two 10-byte fields are to be interchanged, and the variable portion of each logical record is to be copied. The input fields are:

Number	Field Name	Positions
1	Field A	5–14
2	Field B	15–24
	Variable portion	25–275 (maximum)

The FS statement is:

```
1         10        20        30
FS 5,10,15/15,10,5/CV
```

The resulting output record is:

Number	Field Name	Positions
1	Field B	5–14
2	Field A	15–24
	Variable portion	25–275 (maximum)

Example 5 – Move fixed portion of variable-length records:

Assume that an input file contains variable-length records. The minimum length of a logical record is 34 bytes, and the maximum block length is 300 bytes. The fixed portion of the logical record is defined as four bytes and three 10-byte fields. The first and third 10-byte fields are to be moved to positions 5 through 24 in the fixed portion of the output record, and the second 10-byte field is to be moved to positions 35 through 44.

Number	Field Name	Positions
1	Record length	1–4
2	Name	5–14
3	Address	15–24
4	Life number	25–34
5	Variable portion	35–300 (maximum)

The FS statement is:

```
1         10        20        30
FS 5,10,5/25,10,15/CV/15,10,35
```

The resulting output record is:

Number	Field Name	Positions
1	Record length	1–4
2	Name	5–14
3	Life number	15–24
4	Blanks	25–34
5	Address	35–44
6	Variable portion	45–310 (maximum)

3.3.2. Select or Delete Statement - Selecting or Deleting Records

The select statements (SEL, SELAND, and SELOR) and the delete statements (DEL, DELAND, and DELOR) are used in conjunction with the Uio statement to select or delete records, respectively, during a copy operation. The selection or deletion of records is based on matching a portion of each record with a search argument. The matching portion is defined as the search key field and is specified in either the data or key portion of the record. The search argument consists of a relational operator and a character string that represents the constant against which all search fields are matched. The relational operators used to qualify the search argument are equal to (=), greater than (>), or less than (<).

The INPUT1 file is searched, comparing the specified search key field of each record with the search argument. If the search key field of an input record matches the search argument (depending on the relational operator), then the record is either selected or deleted, depending on the requested function. Either SEL or DEL may be specified, but not both. Only one SEL or one DEL statement can be specified. If more than one is entered during a job, only the last SEL or DEL statement is executed. You can specify a maximum of five SELAND, SELOR, DELAND, or DELOR statements, each containing a single argument; however, the statements must all be the same (i.e., all SELAND statements or all DELOR statements).

If the FS statement is also specified, the select or delete function is performed before the field select function.

The format for the select and delete statements is:

OPERATION△	OPERAND
{SEL / SELAND / SELOR / DEL / DELAND / DELOR}	{D=(nnnnn) / K=(nnnnn)},A{> / = / <}(sssss,aa...a)

SEL is the operation code used to select records when a match exists between the search key field and the search argument. All records not selected are ignored.

DEL is the operation code used to delete records when a match exists between the search key field and the search argument. All records not deleted are used as input.

SELAND and DELAND are the operation codes for the "select and" and "delete and" functions, which allow from one to five search key field/search argument pairs. If *all* of the search key fields of an input record match their corresponding search arguments, the record is selected (SELAND) or deleted (DELAND) for output.

SELOR and DELOR are the operation codes for the "select or" and "delete or" functions, which allow from one to five search key field/search argument pairs. If *any* of the search key fields of an input record matches its corresponding search argument, the record is selected (SELOR) or deleted (DELOR) for output.

Keyword parameter *D=(nnnnn)* specifies, in decimal, the starting position of the first byte of the search key field relative to the first character in the record. If the record is a disk record with a key field, this parameter indicates the position of the search key field in the data portion of the record.

Keyword parameter *K=(nnnnn)* specifies, in decimal, the starting position of the field within the hardware key of a keyed DAM or NI file.

Keyword parameter *A* is the search argument definition, where *sssss* is the length in bytes of the search argument and *aa...a* is the search argument. The value of the comparison for a match between the search key field and the constant in the search argument is indicated as follows:

`A>(sssss,aa...a)`
: All records whose search key field is greater than the search argument are either selected or deleted, depending on the function specified.

`A<(sssss,aa...a)`
: All records whose search key field is less than the search argument are either selected or deleted, depending on the function specified.

`A<(sssss,aa...a)`
: All records whose search key field is less than the search argument are either selected or deleted, depending on the function specified.

Subparameter *sssss* specifies the length of the search argument.

Subparameter *aa...a* is the character string representing the search argument.

Example 1:

```
1         10    16
SEL D=(6),A=(3,500)
```

The search field for each record starts in position 6 of the data portion of the record and is matched for an equal condition with the search argument containing a 3-byte character string of 500. The records that satisfy the match are written to the output device.

Example 2:

```
1         10        20
DEL K=(1),A>(3,222)
```

The search field for each record starts at position 1 of the disk key field and is matched for a greater-than condition with the search argument containing a 3-byte character string of 222. The records that satisfy the match are deleted during the copy function.

Example 3:

Suppose you want to select all records for women older than 35. If the sex of the person is in column 3 and the age is in column 50 of the record, the following SELAND statements are used:

```
1         10        20        30
SELAND D=(3),A=(1,F)
SELAND D=(50),A>(2,35)
```

3.3.3. Title Statement - Printing Page Headings

The title (Hn) statement is used, in conjunction with those Uio statements that specify the printer as the primary or secondary output device, to print page headings. You can code a maximum of three Hn statements (each with a maximum of 160 characters) to have page headings printed on the first three lines of each page. If you have included the *PY* keyword parameter in your Uio statement, page headings begin on the first line of each page. The title statement is not supported for compare (K2) operations.

The Hn statement has the following format:

```
     1         10        20        30        40        50        60        72
1.   △Hn ◄———— heading ————————————————————————————————————————————► X
2.   △ ◄———————— heading ————————————————————————————————————————————► X
3.   △ ◄———— heading ————►
```

where:

n

Is 1, 2, or 3 specifying that the character string beginning in column 5 is to be printed as line 1, 2, or 3, respectively, of the title.

heading

Is the character string representing the title. This may be a maximum of 160 characters, with 67 characters coded on line 1, 70 characters on line 2, and 23 characters on line 3.

X

Represents any nonblank character used to indicate continuation. The continuation character is coded in column 72. The heading is continued on the next line (2 or 3) beginning in column 2. The DATA routine assumes that column 2 of line 2 is location 68 of the heading and that column 2 of line 3 is location 137 of the heading.

When the DATA routine scans your Hn statement, it assumes column 5 of the Hn statement is location 1 of the heading; therefore, a character to be printed in position n of a heading must be punched in column n+4 of the Hn statement. For example, if a particular column of information in your job is to be printed beginning in location 10, you must code the heading in the Hn statement beginning in column 14.

If you define a printer line length via the B=(r,p) keyword parameter in the UCP, UDP, or UTP statement, the character string must not exceed this length.

Example:

```
     1         10        20        30        40
1.   △H1           GOLF TOURNAMENT
2.   △H2     PLAYER                SCORES
```

Printer output:

```
          GOLF TOURNAMENT

    PLAYER                SCORES
```

The printed output of line 1 shows the character string GOLF TOURNAMENT beginning in print position 10, PLAYER beginning in print position 5, and SCORES beginning in print position 24. These character strings were coded four positions to the right because the DATA routine considers column 5 in the Hn statement equal to print position 1 in the printed page.

The headings are double spaced because the keyword parameter *S2* was included in the Uio statement.

3.3.4. Correction Statement - Correcting Records

The correction (COR) statement is used to correct records in an existing file during a copy operation. The corrections are made by means of a series of control cards. These control cards are read in as data cards from the control stream. Through their use, a file can have records inserted, deleted, or replaced. The control cards must be in numerical order according to record number. The corrections appear on the OUTPUT1 file; the records in the INPUT1 file are unchanged. In the event of deletions or insertions, the records may be reblocked to suit record positions.

The COR statement specifies the number of new correction records to be processed, where processing is to begin in the file, and where processing is to end, if applicable.

When replacing or inserting records, data cards containing the new or corrected data must immediately follow the COR statement that describes the location in the file for these records. The COR statement, together with any associated data cards, must be the last utility modifier statement in your control stream.

The format of the COR statement is:

OPERATION△	OPERAND
COR	N=(x),A=(s),B=(e)

Parameters:

N=(x)
: Specifies the number of correction records (99 maximum) to be replaced or inserted. If the records are to be deleted, *x* must be zero.

A=(s)
: Specifies the 1- to 7-digit number of the starting record within the file. When replacing or deleting records, *s* specifies the record number of the starting location for the correction records. When inserting corrections, s specifies the record number that the correction records are to follow.

B=(e)
: Specifies the 1- to 7-digit number of the ending record within the file. The ending record, specified by *e*, is the last one to be replaced by the new records.

The data card formats that follow the COR statements are:

First data card:

Subsequent cards:

Columns 1 through 5 of the first data card specify the number of bytes supplied on the data cards for this record. For variable record format INPUT1 files, the DATA routine will construct a 2- or 4-byte record descriptor word as appropriate for the INPUT1 file type if field selection is specified or the OUTPUT1/OUTPUT2 file type if field selection is not specified. The maximum length is 32767. Columns 6 through 80 of the first data card contain positions 1 through 75 of the record to be added. If the record length is less than 75, the rightmost columns of the card are ignored.

If the data exceeds column 80 of the first data card, subsequent cards may be used with the data description beginning in column 1. As many cards as necessary can be used to supply the required number of characters in the record. The subsequent cards successively contain positions 76 through 155, 156 through 235, 235 through 315, etc., of the record. Portions of the record that exceed the length specified by nnnnn are ignored.

Each new record must begin on a new card.

Example:

```
     1         10         20         30
1.    COR N=(2),A=(1),B=(2)
2.   ØØØ17TOURNAMENT△RECORD
3.   ØØØ2ØNO△THREE△PUTT△GREENS
4.    COR N=(2),A=(6)
5.   ØØØ13AVERAGE△DRIVE
6.   ØØØØ9260△YARDS
7.    COR N=(Ø),A=(1Ø),B=(12)
```

The first COR statement specifies that the contents of lines 2 and 3 are to be placed between record 1 and record 2. The COR statement in line 4 specifies that two records are to be inserted after record 6. Lines 5 and 6 contain the data to be inserted. The last COR statement specifies that all records between record 10 and record 12 are to be deleted.

3.3.5. Partition Statement - Handling Nonindexed Files

The DATA routine is capable of handling nonindexed partitioned data files on direct access storage devices supported by OS/3 data management. The partition (PAR) statement is used in conjunction with the Uio statement to process files composed of multiple partitions as either input files or output files. A maximum of seven partitions is allowed.

The format of the PAR statement is:

OPERATION△	OPERAND
PAR	$Pi_1Po_1,[Pi_2Po_2,Pi_3Po_3,\ldots,Pi_nPo_n,]$
	$P1=\ recsize_1,blksize_1,\{^{record\text{-}format_1}_{KEYED}\},size_1,uos_1\ ,$
	$P2=\ recsize_2,blksize_2,\{^{record\text{-}format_2}_{KEYED}\},size_2,uos_2\ ,\ldots$
	$Pn=\ recsize_n,blksize_n,\{^{record\text{-}format_n}_{KEYED}\},size_n,uos_n$

If more than one card is used to enter the partition information, each card must begin with the statement identifier *PAR*, and the last operand on each card must be followed by at least one blank space (no comma).

The PAR statement may be placed anywhere within the appropriate data utility control cards except after the first COR card.

The *PiPo* operand specifies the partition number of the input and/or output partitions. The *i* portion specifies the partition number of the partition in INPUT1 to be processed, and the *o* portion specifies the partition number of the partition in OUTPUT1 (or INPUT2) to be processed.

If either the input or output file is SAM (created by DTFSD), the partition number for that file must be zero.

If only the input data or only the output data is from a partitioned file, or if both the input and output partition numbers are the same, the *PiPo* operand takes the form of *Pn*, where *n* is the number of the partitions to be processed. The value of *n* may be from 1 through 7.

When copying partitions, all Uio, FS, SEL, DEL, and COR parameters apply only to the partition specified in the first *PiPo* operand encountered in the control stream. All other partitions ($Pi_2Po_2,...,Pi_nPo_n$) are copied without change.

The operands

$$P_n=(recsize_n,blksize_n,\left\{\begin{matrix}record\text{-}format_n\\KEYED\end{matrix}\right\},size_n,uos_n)$$

specify the necessary volume table of contents (VTOC) information for the OUTPUT1 partition specified as n. If operands of this type are not entered, the VTOC information for OUTPUT1 partitions will be taken from the INPUT1 VTOC information. The *size* subparameter defaults to 100/x, where x is the number of partitions being written and the *uos* subparameter defaults to zero if the input file is not a nonindexed (NI) file.

One operand of this type must be entered for each partition of the OUTPUT1 file unless the input file contains the same number of partitions, and a *PiPo* operand is specified for each partition. If the *INIT* option is specified on the LFD job control statement for OUTPUT1, one operand of this type must be entered for each partition of the OUTPUT1 file whether or not a *PiPo* operand is entered for that partition.

Subparameters

`recsize`
Specifies the record size for fixed-length records. For variable-length records, this subparameter specifies the minimum record size of the variable-length record. If the file has keys, this subparameter specifies the key length.

`blksize`
Specifies the block size for fixed-length blocks. For variable-length blocks, this subparameter specifies the maximum block size of the variable-length block. If the file has keys, this subparameter specifies the length of the data.

`KEYED`
Specifies the partition is keyed. All keyed partitions must be fixed length and unblocked.

`record-format`
Specifies the record format for nonkeyed partitions. The specification may be:

FIXUNB	fixed-length, unblocked records
FIXBLK	fixed-length, blocked records
VARBLK	variable-length, blocked records
VARUNB	variable-length, unblocked records

size

Specifies the percentage of total file allocation to be initially assigned by data management. Used only with partitioned, nonindexed files. If omitted, the input specification in the VTOC is used.

uos

Specifies, as the unit of store, the percentage of secondary disk storage allocation for the file that data management is to suballocate to the partition being defined each time it requires more space. The value may not exceed 100. Secondary storage allocation is specified in the EXT job control statement in the device assignment set for the file. Used for the first (or only) partition of a nonindexed file. If omitted, the value stored in the VTOC is used.

Refer to the basic data management user guide, UP-8068 (current version) for a further explanation of the DTFNI SIZE=n and UOS=n keyword parameters.

Example 1:

Assume that INPUT1 is to be copied to OUTPUT1 with the following specifications:

INPUT1 Partition	OUTPUT1 Partition	Partition recsize	Partition blksize	Partition Format	OUTPUT1 Partition size (%)	OUTPUT1 Partition uos (%)
1	7	0	244	VARBLK	10	33
2	4	5	109	KEYED	18	50
3	1	72	720	FIXBLK	14	50
4*	5	115	345	FIXBLK	15	20
5	2	100	100	FIXUNB	17	100
6	6	0	120	VARBLK	15	75
7	3	9	80	KEYED	11	50

*When copying partition 4 to partition 5, apply the parameters of the Uio, FS, SEL, DEL, and COR statements.

The PAR statements are:

```
      1         10        20        30        40
1.  | PAR P4P5,P1P7,P2P4,P3P1,P5P2,P6P6,P7P3
2.  | PAR P1=(72,720,FIXBLK,14,50)
3.  | PAR P2=(100,100,FIXUNB,17,100)
4.  | PAR P3=(9,80,KEYED,11,50)
5.  | PAR P4=(5,109,KEYED,18,50)
6.  | PAR P5=(115,345,FIXBLK,15,20)
7.  | PAR P6=(0,120,VARBLK,15,75)
8.  | PAR P7=(0,244,VARBLK,10,33)
```

If statements in lines 2 through 8 had not been entered, the required information for the first three subparameters would have been taken from INPUT1 VTOC. However, all size subparameters would have defaulted to 14, and all *uos* parameters would have defaulted to zero. Thus, the recsize, blksize, and record format of partition 3 of INPUT1 would have been used for partition 1 of OUTPUT1; the recsize, blksize, and record format of partition 5 of INPUT1 would have been used for partition 2 of OUTPUT1, etc. (Note the *PiPo* operands.)

Example 2:

Assume that partition 5 of INPUT1 is to be copied to partition 1 of OUTPUT1 and OUTPUT1 is to be a single partition file (recsize=90, blksize=180, fixed block).

The PAR statement is:

```
1         10        20        30        40
PAR P5P1,P1=(90,180,FIXBLK,100,0)
```

Note that the second operand (*P1=*) could have been omitted because the given values are default values.

Example 3:

Assume that partition 7 of INPUT1 is to be compared with partition 7 of INPUT2.

The PAR statement is:

```
1         10
PAR P7P7
```

Note that operand *P7P7* could have been entered as *P7*.

Example 4:

Assume that partition 3 of INPUT1 is to be printed.

The PAR statement is:

```
1         10
PAR P3
```

3.4. MINIMUM MAIN STORAGE REQUIREMENTS

Most executions of the DATA routine can be accomplished in 32K bytes of main storage. However, if you want to compute the exact amount of minimum main storage required to process the DATA routine, use the following formula:

M = Maximum of (31,145, B) ←

where:

M

Is the minimum amount of main storage required to run this job.

B

Is the calculated total number of bytes of main storage required for your job. It represents the sum of the requirements for the functional routines, input/output routines, and data management buffers. B can be determined by the following formula:

$$B = (24{,}300 + FT + IOT + C + D + E + F + G + H + I + J)$$

where:

FT

Is the total size of the functional routines required by this job. The sizes of the functional routines are specified in the functional routine size table (Table 3-2) along with the parameters that specify the function.

IOT

Is the size of all DTFs and I/O routine modules for all files used in the job. The printer DTF and I/O routine module is included in the printer routine size given in Table 3-2, so printer files are not included in this value. See Table 3-3 for the I/O routine sizes.

C

Is the INPUT1 record size as specified in the first entry of the A=( ) parameter. If a // DD statement with a RCSZ= parameter is specified for this file, then this value will be used and it will override the value specified in the A=( ) parameter. If the INPUT1 file is a disk file, then the record size specified in the VTOC entry is used and it will have the final override.

D

Is the INPUT1 block/buffer size as specified in the second entry of the A=( ) parameter. If a // DD statement with a BDSZ= parameter is specified for this file, then this value will be used and it will override the value specified in the A=( ) parameter. If the INPUT1 file is a disk file and a buffer size was specified in the second entry of the A=( ) parameter, then that value will be used. If the INPUT1 file is a disk file other than IRAM/MIRAM and the A=( ) parameter is not specified, then the minimum allowable buffer size will be calculated by using the record size specified in the VTOC entry for the INPUT1 file.

E

Is the OUTPUT1/INPUT2 record size as specified in the first entry of the B=( ) parameter. If a // DD statement with a RCSZ= parameter is specified for this file, then this value will be used and it will override the value specified in the B=( ) parameter. If this is a compare operation, (K2 parameter) and the INPUT2 file is a disk file, then the record size specified in the VTOC entry is used and it will have the final override.

F

Is the OUTPUT1/INPUT2 block/buffer size as specified in the second entry of the B=( ) parameter. If a // DD statement with a BKSZ= parameter is specified for this file, then this value will be used and it will override the value specified in the B=( ) parameter. If compare is specified (K2), the INPUT2 file is a disk file, and the buffer size is specified in the B=( ) parameter, then that value will be used. If compare is specified, the INPUT2 file is a disk file, and the B=( ) parameter is not specified, then the minimum allowable buffer size is calculated by using the record size specified in the VTOC entry for the INPUT2 file.

G

This variable is for ISAM and IRAM INPUT1 files only. It is the record key length specified in the VTOC entry for the INPUT1 file, and for ISAM and IRAM INPUT1 files only.

H

Is for ISAM and IRAM OUTPUT1/INPUT2 files only. It is the record key length, as specified in the V=( ) parameter. If this is a compare (K2), then the key length is taken from the disk file format labels.

I

Is for IRAM and MIRAM INPUT1 disk files only. It is the index buffer size and is taken from the disk file format labels.

J

Is for IRAM and MIRAM OUTPUT1/INPUT2 files only. It represents the disk file index buffer size and it is calculated by multiplying the second entry in the OR=(I,n) or the OM=(I,n) parameter by 256 decimal bytes. If this is a compare (K2) procedure, then this value will be taken from the disk file format labels.

NOTE:

FT = 0 whenever none of the functions in Table 3—2 are specified. Table 3—2 specifies the functional routine size with the associated parameter or Uio statement.

Table 3—2. Functional Routine Sizes

Functional Routine ①	Associated Parameter or Statement	Size
Correction	COR statement	4410 ③
Select/Delete	SEL/DEL statement	2030
Field Selection ②	FS statement	2630 ④
Sequence Checking	X=( ) parameter	680
Compare	K2 parameter	3970
Print Routine	UCP,UTP,UDP, or DP	3630

NOTES:

① A size is not shown for the copy functional routine (K1 parameter) because the copy function is included in the base size of the formula (17,300 or 24,700).

② The field selection routine is automatically included when you specify variable record processing with DAM and nonindexed files. It is also included if you specify OS/4 to OS/3 processing (// PARAM MODE=OS4). Consequently, the size of this routine must be included in your minimum main storage calculations when these situations arise. If the DU23-I-INPUT REC LENG NOT EQUAL TO OUTPUT REC LENG is displayed, the field selection routine will be used to truncate or blank fill the INPUT1 records.

③ The maximum INPUT1 file record size must be added to this value.

④ The maximum OUTPUT1/OUTPUT2 file record size must be added to this value.

Table 3—3. I/O Routine Sizes

Type	INPUT1	INPUT2	OUTPUT1	OUTPUT2
Basic Data Management				
Card/Diskette	510	500	475	600
Tape	1470	1355	1380	
Disk				
DAM	780	775	720	
SAM	880	770	660	
ISAM	950	830	1000	
Nonindexed	1430	1350	1350	
IRAM	875	890	1075	
MIRAM	1040	880	1260	
DCON4 Tape	920			
Mixed or Consolidated Data Management				
8413 Card	425	430	435	520
Tape	1280	1175	1170	
Disk				
DAM	780	775	720	
SAM	880	770	660	
ISAM	950	830	1000	
Nonindexed	1430	1350	1350	
IRAM	875	890	1075	
MIRAM	900	740	1130	
DCON4 Tape	735			
Diskette	900	740	1130	

Example 1:

Copy SAM disk file to tape in a basic data management (DTF) environment.

UDT A=(128,256),B=(128,1024),K1

$$B = (17{,}300 + FT + IOT + C + D + E + F + G + H + I + J)$$

FT = 0 because the copy function is included in the 17,300 base size of the formula.

IOT = 880 for SAM disk INPUT1 I/O routine
+ 1380 for tape OUTPUT1 I/O routine
2260

C = 128 for INPUT1 record size

D = 256 for INPUT1 block size

E = 128 for OUTPUT1/OUTPUT2 record size

F = 1024 for OUTPUT1 block size

G through I are not used for SAM disk files or tape files

B = (17,300 + 0 + 2260 + 128 + 256 + 128 + 1024)

B = 21,096

M = 24,700 or B, whichever is greater

Since B = 21,096

M = 24,700 or X'607C' bytes required for the DATA routine

Example 2:

Copy an ISAM disk file to an ISAM disk file and print the output file using mixed or consolidated data management.

UDD A=(122,256),B=(122,256),DP,OB,OI,PY,TD,V=(20,20),W=(O,O)

NOTES:

1. *The A=( ) parameter is used only to show the INPUT1 record and block sizes. These values are from INPUT format labels.*

2. *The V=( ) and W=( ) parameter should have only an output entry. The input entries are used to show only the INPUT1 key length and key location. These values are from the INPUT1 format labels.*

B = (17,300 + FT + IOT + C + D + E + F + G + H + I + J)

FT = 4700 for PRINT functional routine

IOT = 950 for ISAM disk INPUT1 I/O routine
+ 1000 for ISAM disk OUTPUT1 I/O routine
1950

C = 122 for INPUT1 record size

D = 256 for INPUT1 block size

E = 122 for OUTPUT1 record size

F = 256 for OUTPUT1 block size

G = 20 for INPUT1 key size

H = 20 for OUTPUT1 key size

I and J are not used for ISAM files

B = (17,300 + 4700 + 1950 + 122 + 256 + 122 + 256 + 20 + 20)

B = 24,746

M = 24,700 or B, whichever is greater

Since B = 24,746

M = 24,746 or X'60AA' bytes required for the data utility routine

Example 3:

Compare a card file to a tape file with field select by using basic data management.

```
UCT A=(80,80),B=(80,800),OR,K2
    FS1,5,1/6,10,70/20,10,20
```

B = (17,300 + FT + IOT + C + D + E + F + G + H + I + J)

FT = 2500 for compare functional routine
3000 for field selection functional routine
\+ 80 for maximum INPUT2 record size
5580

IOT = 510 for card INPUT1 I/O routine
\+ 1380 for tape OUTPUT1 I/O routine
1890

C = 80 for INPUT1 record size

D = 80 for INPUT1 block size

E = 80 for OUTPUT1/OUTPUT2 record size

F = 800 for INPUT2 block size

G through J are not used for card or tape files.

B = (17,300 + 5580 + 1890 + 80 + 80 + 80 + 800)

B = 25,810

M = 24,700 or B, whichever is greater

Since B = 25,810

M = 25,810 or X'64D2' bytes required for the data utility routine

4. Sample Control Streams

4.1. PURPOSE AND APPLICATION

The sample programs in this section illustrate some of the functions of the DATA routine. These programs all use the same input data and represent a typical progression for handling the data. Selected fields from the input data card deck are written to a work area on disk and then the two are compared. The data in the work area is copied to another work area on the disk and again these two are compared. The data in the first work area is printed out. Finally, one of the records in the data file is corrected.

The input data deck shown is for illustrative purposes; actual data decks could be considerably larger. Figure 2-1 shows a sample control stream using embedded card data. All card and disk files used in these examples are in SAM format. Job control statements, Uio statements, utility modifier statements, and data decks needed for the job steps are included.

The amount of minimum main storage required for each job step is computed according to the specifications shown in 3.4.

4.2. COPY CARD-TO-DISK OPERATION

In this example, the employee records of the data deck are copied to a disk file and written to the printer file. Some fields of the input file are rearranged by the FS statement, and records are deleted by the DEL statement. Also requested is the printing of headings on the printer output.

Job control stream:

```
                1         10        20        30        40        50        60        70        80
JOB CONTROL  1. // JOB DATAUT,,6000
             2. // DVC 20      // LFD PRNTR
             3. // DVC 30      // LFD INPUT1
             4. // DVC 51      // VOL DSP028
             5. // EXT SQ,C,,CYL,10
             6. // LBL WORK2   // LFD OUTPUT1
             7. // EXEC DATA
             8. /$
Uio CARD     9. UCD B=(90,90),DP,MY,PY,S2
UTILITY     10. FS 1,10,25/15,20,1/45,3,45/55,2,55/65,3,65/78,3,78/1,10,81
MODIFIER    11. DEL D=(1),A=(4,1111)
CARDS       12.  H1                              EMP              HOURLY    EMP         EMPX
            13.             SEQ
            14.  H2         NAME                 NO               RATE      CLASS       RATIX
            15.  NG          NO
            16. /*
            17. /&
            18. // FIN
            19. // DATA FILEID=DATAUTINPUT1
DATA        20. 0000789123     DONAHUE, MARY M              255       10        030         120
CARDS       21. 0000776312     GIAGLONE, JOHN K             255       10        035         125
            22. 0000617892     LANGINES, SAMANTHA R         270       11        040         130
            23. 1111777321     LONG, GEORGE A               255       10        063         135
            24. 0000678978     LOTIERZO, HELENE S           310       13        050         140
            25. 0000776321     MATHEWS, SAMUEL T            425       15        071         145
            26. 0000667843     MATTHEWS,JANE L              270       11        045         150
            27. 0000667712     NORDICK, JOHN M              425       14        055         155
            28. 1111573212     OGALSBY, PETER K             500       15        075         160
            29. 0000673124     PARYIN, THOMAS G             425       14        060         165
            30. /*
            31. // FIN
```

Printer output:

```
PAGE  0001   DATE 75/02/11

                          EMP              HOURLY    EMP         EMP          SEQ
      NAME                NO               RATE      CLASS       RATING       NO

DONAHUE, MARY M           0000789123       255       10          030          120000789123
GIAGLONE, JOHN K          0000776312       255       10          035          125000776312
LANGINES, SAMANTHA R      0000617892       270       11          040          130000617892
LOTIERZO, HELENE S        0000678978       310       13          050          140000678978
MATHEWS, SAMUEL T         0000776321       425       15          071          145000776321
MATTHEWS, JANE L          0000667843       270       11          045          150000667843
NORDICK, JOHN M           0000667712       425       14          055          155000667712
PARYIN, THOMAS G          0000673124       425       14          060          165000673124
```

The job control cards to assign devices, allocate main storage and disk space, and initiate the DATA routine are included in the job control stream (cards 1 through 8). The printer file, card reader file and disk file are assigned, plus the space SAM with no data management disk check function. The input file type is taken from the VTOC.

The UCD mnemonic of the Uio statement specifies that this is a card-to-disk operation. The *A* keyword parameter is not coded because the default value of 80-byte record and block lengths applies. The *B* keyword parameter specifies the output record and block length of 90 bytes. The *DP* keyword parameter specifies printing of output records.

The keyword parameters *K1* and *OS* are not coded because their default values apply to this job step. *K1* specifies a copy operation, and *OS* specifies that the format of the output file is in SAM with no data management disk check function. The input file type is taken from the VTOC.

Keyword parameters *MY, PY,* and *S2* specify to ignore printer mismatches, print page numbers and date on the first line of each page, and double-space the printer file output. The printer output is written in the *list* format by the default value of the *TL* keyword parameter.

The fields beginning in columns 1 and 15 of the card file are moved to different locations in the disk file by the specifications in the first two parameters of the FS statement. The first field of the card file is repeated in columns 81 through 90 of each record in the output disk file by the specifications in the last parameter (card 10).

The DEL statement (card 11) deletes all records containing the characters 1111 in the first four columns of the card file. Therefore, the input data records in cards 23 and 28 are not copied to the disk file and do not appear in the printer output.

Note that the column headings in the Hn statements (cards 12 through 15) are coded five positions to the right of their positions shown on the printer output. Within the DATA routine, column 5 of Hn statements is considered equal to column 1 (3.3.3).

The continuation of the title lines is shown in cards 13 and 15.

The // FIN card is required by spooling at end-of-data cards.

The // DATA FILEID job control statement on card 19 directs the card reader input to a spool file for the job. DATAUTINPUT1 is a concatenation of the job name and the LFD file name for the card reader. To spool in the data, key in the command IN at the console.

The printer output shows that all DATA routine specifications have been executed successfully.

4.3. COMPARE CARD-TO-DISK OPERATION

This example illustrates the programmer messages that result when unequal records are compared.

Job control stream:

```
                1         10        20        30        40        50        60        70        80
JOB CONTROL  1. // JOB DATAUTIL,,6000
             2. // DVC 20      // LFD PRNTR
             3. // DVC 30      // LFD INPUT1
             4. // DVC 51      // VOL DSP028
             5. // LBL WORK2   // LFD INPUT2
             6. // EXEC DATA
             7. /$
Uio CARD     8. UCD B=(90,90),K2
             9. /*
            10. /&
            11. // FIN
DATA CARDS  12. // DATA FILEID=DATAUTIL INPUT1
            13. 0000789123     DONAHUE, MARY M              255       10       030       120
            14. 0000776312     GIAGLONE, JOHN K             255       10       035       125
            15. 0000617892     LANGINES, SAMANTHA R         270       11       040       130
            16. 1111777321     LONG, GEORGE A               255       10       063       135
            17. 0000678978     LOTIERZO, HELENE S           310       13       050       140
            18. 0000776321     MATHEWS, SAMUEL T            425       15       071       145
            19. 0000667843     MATTHEWS, JANE L             270       11       045       150
            20. 0000667712     NORDICK, JOHN M              425       14       055       155
            21. 1111573212     OGALSBY, PETER K             500       15       075       160
            22. 0000673124     PARYIN, THOMAS G             425       14       060       165
            23. /*
```

Printer output:

```
INPUT1
0000789123     DONAHUE, MARY M              255       10       030       120
INPUT2
DONAHUE, MARY M          0000789123         255       10       030       1200000789123
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000776312     GIAGLONE, JOHN K             255       10       035       125
INPUT2
GIAGLONE, JOHN K         0000776312         255       10       035       1250000776312
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000617892     LANGINES, SAMANTHA R         270       11       040       130
INPUT2
LANGINES, SAMANTHA R     0000617892         270       11       040       1300000617892
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
1111777321     LONG, GEORGE A               255       10       063       135
INPUT2
LOTIERZO, HELENE S       0000678978         310       13       050       1400000678978
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000678978     LOTIERZO, HELENE S           310       13       050       140
INPUT2
MATHEWS, SAMUEL T        0000776321         425       15       071       1450000776321
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000776321     MATHEWS, SAMUEL T            425       15       071       145
INPUT2
MATTHEWS, JANE L         0000667843         270       11       045       1500000667843
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000667843     MATTHEWS, JANE L             270       11       045       150
INPUT2
NORDICK, JOHN M          0000667712         425       14       055       1550000667712
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000667712     NORDICK, JOHN M              425       14       055       155
INPUT2
PARYIN, THOMAS G         0000673124         425       14       060       1650000673124
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
UDD14 INPUT1 AND INPUT2 ARE OF UNEQUAL LENGTH
```

We used our original data deck (cards 12 through 21) to compare the card file to the WORK2 area of the disk file. The job control cards assign the printer file, the card reader file, disk file, and the WORK2 disk area (cards 2 through 5).

The UCD mnemonics specify a card-to-disk operation (card 8). The default values of the *A* and *OS* keyword parameters are assigned. The *B* keyword parameter specifies 90-byte output record and block lengths. A compare operation is specified by the *K2* keyword parameter.

The resulting printout shows all records unequal because the first two fields were rearranged and a 10-byte field was added to all disk file records during the copy operation. If all records were equal, the job step would have terminated normally.

4.4. COPY DISK-TO-DISK OPERATION

In this example, the data stored in the WORK2 area of the disk file is copied to another area in the same disk file.

Job control stream:

JOB CONTROL CARDS (cards 1–9)

```
             1        10        20        30
        1.  // JOB DAMAIX,,6000
        2.  // DVC 20       // LFD PRNTR
        3.  // DVC 51       // VOL DSP028
        4.  // LBL WORK2    // LFD INPUT1
        5.  // DVC 51       // VOL DSP028
        6.  // EXT SQ,C,,CYL,10
        7.  // LBL WORK3    // LFD OUTPUT1
        8.  // EXEC DATA
        9.  /$
       10.  UDD DP,MY,S2
       11.  /*
       12.  /&
       13.  // FIN
```

Printer output:

DONAHUE, MARY M	0000789123	255	10	030	120000789123
GIAGLONE, JOHN K	0000776312	255	10	035	125000776312
LANGINES, SAMANTHA R	0000617892	270	11	040	130000617892
LOTIERZO, HELENE S	0000678978	310	13	050	140000678978
MATHEWS, SAMUEL T	0000776321	425	15	071	145000776321
MATTHEWS, JANE L	0000667843	270	11	045	150000667843
NORDICK, JOHN M	0000667712	425	14	055	155000667712
PARYIN, THOMAS G	0000673124	425	14	060	165000673124

The printer file, the disk file, input WORK2 area, and output WORK3 area are assigned to the job step (cards 2 through 7). The output area, WORK3, is allocated by the // EXT statement (card 6).

A disk-to-disk operation is specified by the mnemonic UDD. The input record and block lengths are taken from the VTOC, and the output record and block lengths are assumed to be equal to the input lengths. We have used the *DP* keyword parameter to request that records copied to the output area, WORK3, be written to the printer file. By using the keyword parameters *MY* and *S2*, printer mismatches are ignored and the printer output is double spaced. The printer output is printed in the *list* format by the default value of the *TL* keyword parameter.

The printer output shows that the complete WORK2 area has been copied to the WORK3 disk area.

4.5. COMPARE DISK-TO-DISK OPERATION

This example shows the job control stream used to compare the data stored in disk WORK2 area to the data stored in disk WORK3 area. Any unequal records are listed on the printer output.

Job control stream:

```
                1         10        20        30
JOB CONTROL CARDS
 1.  // JOB CATUDT,,6000
 2.  // DVC 20     // LFD PRNTR
 3.  // DVC 51     // VOL DSP028
 4.  // LBL WORK2  // LFD INPUT1
 5.  // DVC 51     // VOL DSP028
 6.  // LBL WORK3  // LFD INPUT2
 7.  // EXEC DATA
 8.  /$
Uio CARD
 9.  UDD K2
10.  /*
11.  /&
12.  // FIN
```

The card reader file, printer file, and disk file are assigned to this job step in the job control stream (cards 2, 3, and 5). The WORK2 disk area is defined as INPUT1, and WORK3 disk area as INPUT2 (cards 4 and 6).

A disk-to-disk operation is specified by the UDD mnemonic. The same disk files are used for this compare operation as were used for the previous copy operation, so our *A* and *B* keyword parameter values remain the same, although the value for the *A* keyword parameter is taken from the VTOC for INPUT1 file and the value for the *B* keyword parameter is taken from the VTOC for INPUT2 file. *K2* keyword parameter specifies a compare operation.

The printer output shows that all records in disk area WORK2 compare equally to the records in disk area WORK3. Had there been unequal records, they would have been displayed as they were in 4.2.

4.6. COPY DISK-TO-PRINTER OPERATION

Shown in this example is the job control stream required to write the contents of the disk WORK2 area into the printer file.

Job control stream:

```
                    1         10        20
                    |---------|---------|
JOB CONTROL CARDS
 1.  // JOB PRINTED,,6000
 2.  // DVC 51
 3.  // VOL DSP028
 4.  // LBL WORK2
 5.  // LFD INPUT1
 6.  // DVC 20
 7.  // LFD PRNTR
 8.  // EXEC DATA
 9.  /$
Uio CARD
10.  UDP MY,OB,PY,TD,S2
11.  /*
12.  /&
13.  // FIN
```

Printer output:

```
PAGE  0001   DATE 02/01/78

     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000001    00240  00080    0000789123    DONAHUE, MARY M              255       10        030          120
                               FFFFFFFFFF4444CDDCCEC64DCDE4D444444444444444FFF444444FF444444444FF444444444FFF
                               0000789123000046518458041980400000000000000025500000010000000000300000000000120
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000002    00240  00080    0000776312    GIAGLONE, JOHN K             255       10        035          125
                               FFFFFFFFFF4444CCCCDDDC64DDCD4D444444444444444FFF444444FF444444444FFF444444444FFF
                               0000776312000079173655801685020000000000000025500000010000000000350000000000125
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000003    00240  00080    0000617892    LANGINES, SAMANTHA R         270       11        040          130
                               FFFFFFFFFF4444DCDCCDCE64ECDCDECC4D444444444FFF444444FF444444444FFF444444444FFF
                               0000617892000031579552802141538109000000000270000000110000000040000000000130
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000004    00240  00080    1111777321    LONG, GEORGE A               255       10        043          135
                               FFFFFFFFFF4444DDDC64CCDDCC4C444444444444444FFF444444FF444444444FFF444444444FFF
                               1111777321000036578075697501000000000000000255000000100000000004300000000135
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000005    00240  00080    0140678978    LOTIERZO, HELENE S           310       13        050           40
                               FFFFFFFFFF4444DDECCDED64CCDCDC4E444444444444FFF444444FF444444444FFF444444444FFF
                               0140678978000036365996808535550200000000000031000000001300000000500000000F40
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000006    00240  00080    0000776321    MATHEWS, SAMUEL T            425       15        071          145
                               FFFFFFFFFF4444DCECCEE64ECDECD4E444444444444444FFF444444FF444444444FFF444444444FFF
                               0000776321000041384562802144530300000000000042500000001500000000071000000000145
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000007    00240  00080    0000667843    MATTHEWS, JANE L             270       11        045          150
                               FFFFFFFFFF4444DCEFCCEE64DCDC4D444444444444444FFF444444FF444444444FFF444444444FFF
                               0000667843000041338562801155030000000000027000000011000000004500000000150
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000008    00240  00080    0000667712    NORDICK, JOHN M              425       14        055          155
                               FFFFFFFFFF4444DDDCCCD64DDCD4D444444444444444FFF444444FF444444444FFF444444444FFF
                               0000667712000056949328016850400000000000000042500000001400000000055000000000155
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000009    00240  00080    1111573212    OGALSBY, PETER K             500       15        075          160
                               FFFFFFFFFF4444DCCDECE64DCECD4D444444444444444FFF444444FF444444444FFF444444444FFF
                               1111573212000067132288075359020000000000000050000000015000000000750000000000160
     REC#      BLKSZ  RECSZ    ........10........20........30........40........50........60........70........80

 0000000010    00240  00080    0000673124    PARYIN, THOMAS G             425       14        060          165
                               FFFFFFFFFF4444DCDECD64ECDDCE4C444444444444444FFF444444FF444444444FFF444444444FFF
                               0000673124000071989580386412070000000000000042500000001400000000600000000000165
```

The WORK2 area of the disk file is assigned as INPUT1 (cards 2 through 5). The mnemonic UDP signifies a disk-to-printer operation. A 90-byte record and block length input file is specified by values stored in the VTOC. The default values of the *B* keyword parameter, 90-byte record length, and 120-byte print line, are acceptable for this job step. Printer mismatches are ignored by use of the *MY* keyword parameter. The output is printed in hexadecimal and EBCDIC as specified by the keyword parameter *OB*. Page numbers and dates are printed by specifying the *PY* keyword parameter. The *TD* keyword parameter specifies printing the output in the display format.

4.7. CORRECTION OPERATION

In this example, we will correct one of the records on the WORK3 area of the disk file by using the COR statement.

The amount of the hourly rate and employee class rating for John K. Giaglone will be changed from 255 to 270 for the hourly rate, and from 10 to 11 for the employee class. All other information in this record remains the same.

Job control stream:

```
                1         10        20        30        40        50        60        70        80
                |         |         |         |         |         |         |         |         |
JOB CONTROL CARDS
 1.  // JOB CORRECT,,6000
 2.  // DVC 20        // LFD PRNTR
 3.  // DVC 51        // VOL DSP028
 4.  // LBL WORK3     // LFD INPUT1
 5.  // DVC 51        // VOL DSP028
 6.  // EXT SQ,C,,CYL,10
 7.  // LBL WORK4     //LFD OUTPUT1
 8.  // EXEC DATA
 9.  /$
Uio CARD
10.  UDD B=(90,90),MY,OSY
UTILITY MODIFIER CARDS
11.  COR N=(1),A=(2),B=(2)
12.  00090GIAGLONE,  JOHN K          000077 6312          270          11          035
13.            125 0000776312
14.  /*
15.  /&
16.  // FIN
```

Printer output:

DONAHUE, MARY M	0000789123	255	10	030	1200000789123
GIAGLONE, JOHN K	0000776312	270	11	035	1250000776312
LANGINES, SAMANTHA R	0000617892	270	11	040	1300000617892
LOTIERZO, HELENE S	0000678978	310	13	050	1400000678978
MATHEWS, SAMUEL T	0000776321	425	15	071	1450000776321
MATTHEWS, JANE L	0000667843	270	11	045	1500000667843
NORDICK, JOHN M	0000667712	425	14	055	1550000667712
PARYIN, THOMAS G	0000673124	425	14	060	1650000673124

Device assignment, allocation of main storage and disk space, and initiation of the DATA routine are included in the job control cards (cards 1 through 9). Note that card 6 is included to allocate disk space for the WORK4 area. This new area on the disk file is required because our original area (WORK3) cannot be corrected, but must be written to another area while corrections are being made.

The UDD mnemonic specifies a disk-to-disk operation. The input block and record lengths are taken from the VTOC. The *B* keyword parameter specifies that the output record length is 90 bytes and the block length is also 90. The records are fixed length. The *MY* and *OSY* keyword parameters specify that printer mismatches are to be ignored and that the output file is in SAM format with a write disk check configured.

In the COR statement, the keyword parameter *N* specifies that one record follows. The keyword parameter *A* specifies that the starting location of the record to be corrected is record 2 and the keyword parameter *B* specifies the ending of the correction (card 11). Lines 12 and 13 specify the correction that is to be made to record 2.

The printer output shows that the corrections to this record have been executed successfully.

5. DATA Routine Jprocs

5.1. PURPOSE AND APPLICATION

A job control procedure (jproc) is a series of job control statements that establishes a sequence of steps that lead to the solution of a problem, i.e., the proper control stream needed to assign the files and devices used by your job. This jproc eliminates the need to repeatedly code identical sequences of job control statements. OS/3 allows you to code a single statement (a jproc call statement) that generates the proper job control statement sequence for you.

Sperry Univac supplies a set of jprocs that contains the job control statements that are needed to execute the DATA routine. These jprocs are stored in the job control stream library file (YJCS). In order to use them, you place a jproc call statement in your control stream at the point where you would have otherwise placed the job control statements (if you had used them). The jproc call statement then generates the proper sequence of job control statements, depending on values you supply through positional and keyword parameters. (Refer to the job control user guide, UP-8065 (current version) for detailed information on writing and calling procedure definitions.)

The DATA routine jprocs can be used to simplify the job control stream when you run a disk-to-disk (UDD), disk-to-tape (UDT), or tape-to-disk (UTD) operation. The jproc call statement can also be used when you compare two files. This method can be used to shorten the job control stream described in 2.1.12. However, both methods are functionally identical, and use of either method is at your discretion.

5.2. COMBINATION OF FILE TYPES

Tape or disk files may be copied or relocated to the same volume or to a different volume by use of a jproc call statement. Table 5–1 lists the types of input and output files that may be copied or compared by a jproc call statement.

Table 5—1. File Types Used with Jprocs

Input File	Output File
SAM tape, fixed or variable	SAM disk, fixed, or variable DAM disk, fixed or variable ISAM disk, fixed or variable IRAM disk, fixed MIRAM disk, fixed or variable SAM tape, fixed or variable
DAM disk, fixed or variable	SAM disk, fixed or variable DAM disk, fixed or variable ISAM disk, fixed or variable IRAM disk, fixed MIRAM disk, fixed or variable SAM tape, fixed or variable
ISAM disk, fixed or variable	SAM disk, fixed or variable DAM disk, fixed or variable ISAM disk, fixed or variable IRAM tape, fixed MIRAM disk, fixed or variable IRAM disk, fixed MIRAM tape, fixed or variable SAM tape, fixed or variable
IRAM disk, fixed or variable	SAM disk, fixed or variable DAM disk fixed or variable ISAM disk, fixed or variable IRAM disk, fixed MIRAM disk, fixed or variable SAM tape, fixed or variable
MIRAM disk, fixed or variable	SAM disk, fixed or variable DAM disk, fixed or variable ISAM disk, fixed or variable IRAM disk, fixed MIRAM disk, fixed or variable SAM tape, fixed or variable
SAM tape, fixed or variable	SAM disk, fixed or variable DAM disk, fixed or variable ISAM disk, fixed or variable IRAM disk, fixed or variable MIRAM disk fixed or variable SAM tape, fixed or variable

NOTE:

SAM = sequential access method
DAM = direct access method
ISAM = indexed sequential access method
IRAM = indexed random access method
MIRAM= multiple indexed random access method

5.3. JOB CONTROL REQUIREMENTS

Your operations with job control are simplified with a jproc call statement. The UDD, UDT, and UTD statements generate the EXEC job control statement and all other device assignment cards you need to run a disk-to-disk, disk-to-tape, or tape-to-disk operation. To ensure compatibility between disk and tape unit numbers, the DVCVOL jproc call statement assigns logical unit numbers for disk units, and the DVCVTP jproc call statement assigns logical unit numbers for tape units. (Refer to the job control user guide, UP-8065 (current version).)

The basic structure of your job control stream when using a jproc call statement is as follows:

1. JOB control statement (// JOB parameters)
2. Data utilities jproc call statement (// UDD or // UDT or // UTD)
3. Start-of-data job control statement (/$)
4. Data utilities control statement (Refer to 2.1.5.)
5. End-of-data job control statement (/*)
6. End-of-job control statement (/&)
7. FIN job control statement (// FIN)

5.4. JOB CONTROL PROCEDURES

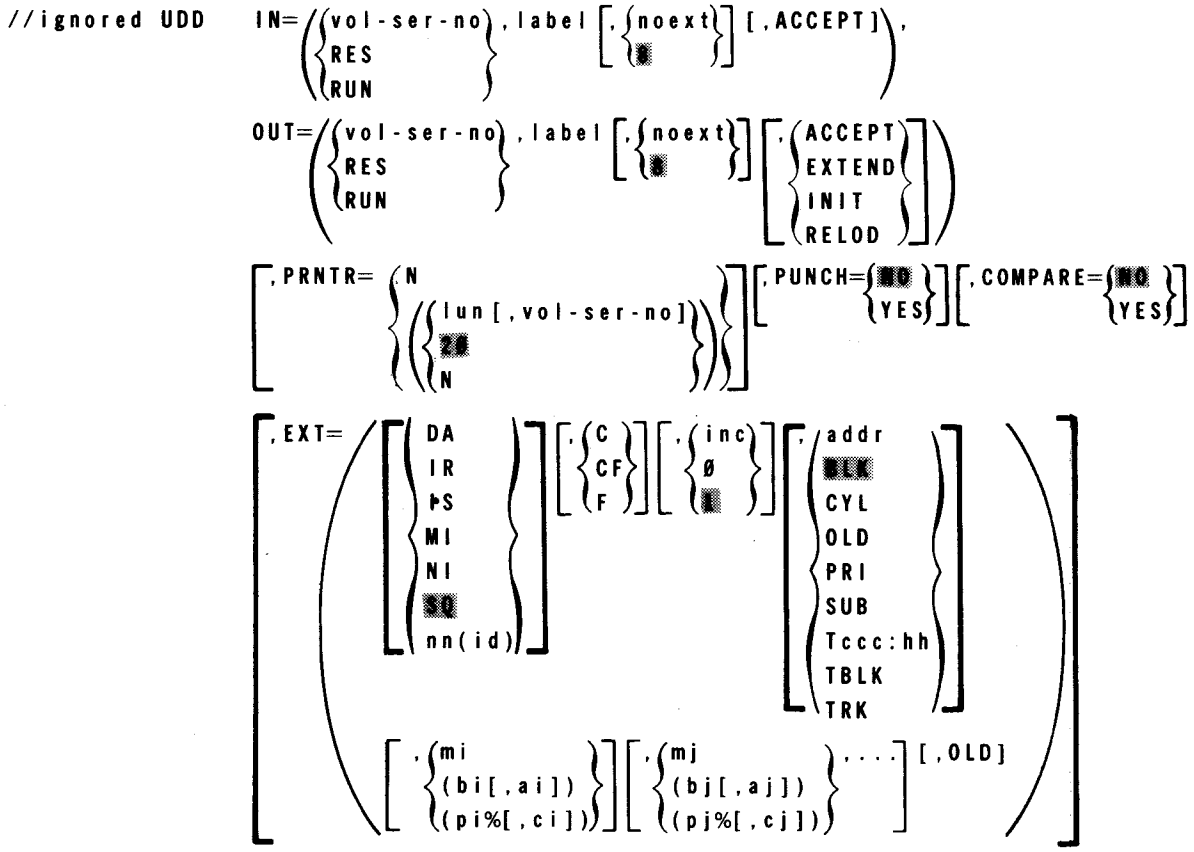
The three types of job control procedures (UDD, UDT, and UTD) are described in 5.4.1 through 5.4.3.

5.4.1. UDD Job Control Procedure

Function:

UDD generates the job control statements for your device assignment sets that are required by the DATA routine to copy or compare one disk file to another disk file (disk-to-disk operation).

Format:



Label:

The label field is ignored.

Parameters:

IN=

Supplies the information required to define the input file. The information is coded within the parentheses.

where:

vol-ser-no

Specifies the volume serial number of the volume containing the input file.

RES

Specifies that the input file is located on the SYSRES.

RUN

Specifies that the input file is located on the volume containing the job's YRUN file.

label

Specifies the file identifier for the input file.

noext

Specifies the number of extents in the file requiring space to be reserved in the prologue. The default value is 8.

ACCEPT

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

OUT=

Supplies information required to define the output file for a copy operation or to define a secondary input file for a compare operation.

where:

vol-ser-no

Specifies the volume serial number of the disk containing the output (or secondary input) file.

RES

Specifies that the output (or secondary input) file is located on the SYSRES disk.

RUN

Specifies that the output (or secondary input) file is located on the volume containing the job's YRUN file.

label

Specifies the file identifier for the output (or secondary input) file.

noext

Specifies the number of extents in the file requiring space to be reserved in the job prologue. The default value is 8.

ACCEPT

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

EXTEND

Indicates that a sequential (SQ) file is to be extended. The information is appended to the present end of the file.

INIT

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

RELOD

Indicates that the specified IRAM or ISAM file is not to be reformatted at file open time when the file is being reloaded.

PRNTR=

Indicates whether a device assignment set is to be generated for the printer and defines the print file.

where:

N

Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of SPL, LCB, and VFB job control statements.

{lun / [redacted]}

Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no

Provides a 1- to 6-alphanumeric-character remote destination identifier for the print file when dealing with remote job entry.

PUNCH=YES

Generates the device assignment set for the card punch. This parameter is required when the punch dual output feature (DC) is selected on the data control card.

PUNCH=NO

No device assignment set is generated for the card punch. This is the default value and need not be specified.

COMPARE=YES

Specifies that the compare operation (K2) is selected on the data control card. This keyword causes the file name INPUT2 (secondary input) to be generated for the file specified by the *OUT* keyword parameter.

COMPARE=NO

Specifies that this is a copy operation. This is the default value and need not be specified.

EXT=

Specifies the extent specifications to be used when reserving system resources for an output file. The information is supplied as a subparameter list enclosed within parentheses.

where:

DA

Indicates this is a direct access file.

IR

Indicates this is an IRAM or a MIRAM file.

IS

Indicates this is an ISAM file.

MI

Indicates this is a MIRAM file.

NI

Indicates this is a nonindexed (DA or SQ) file.

SQ

Indicates this is a sequential file. *SQ* is the default option.

nn(id)

Indicates this file is a member of a split cylinder set, where *nn* is an *SQ, NI,* or *DA,* and *(id)* is the 2-character set identification. If the first character of the set identification is *$,* the set is a job step temporary work file, and all member names must begin with $SCR. Job control automatically deallocates the file.

C

Indicates that the file must be allocated contiguously.

F

Indicates that the file is to be formatted at allocation time. You must select *BLK* for subparameter 4.

CF

Indicates that both of the preceding parameters apply for subparameter 2.

NOTE:

If subparameter 2 is omitted (C, F, or CF), none of the previous options apply.

inc

Specifies the secondary increment (in cylinders) by which the file is to be extended if automatic extension is required.

Ø

Specifies there can be no dynamic extension of the file.

1

Indicates one cylinder block increment is specified.

NOTE:

If there is no EXT keyword parameter for this file, the value of the most recently specified secondary increment is implemented.

addr

Specifies the absolute cylinder address where the file is to begin. Allocation is specified in terms of cylinders.

BLK

Indicates that allocation is in terms of blocks. This is the default value if no allocation type is specified.

CYL

Indicates that allocation is in terms of cylinders.

OLD

Indicates that the secondary allocation increment (subparameter 3) is changed. No additional space is allocated by job control. If specified, it must be the last subparameter in the list.

PRI

Indicates that this is the primary (extent definition) member of a split cylinder set. Subparameter 1 must be *nn(id).*

SUB

Indicates that this is a subsequent member of a split cylinder set. Subparameter 1 must be *nn(id).*

Tccc:hh

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

TBLK

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

TRK

Allocates space in tracks.

`mi`

Specifies the number of cylinders to allocate for this file. Subparameter 4 must be *CYL* or *addr.*

If a split cylinder allocation is to be made, remember that:

- If subparameter 4 is *PRI* and this is not the first *EXT* keyword parameter for this file, this option specifies the number of cylinders for this extent. The number of tracks is assumed to have been specified on the first *EXT* keyword parameter for the primary member.
- If subparameter 4 is *SUB,* this option specifies the number of tracks to be allocated to this secondary member of a split cylinder.

`(bi[,ai])`

This option is used when allocation is in terms of blocks, where *bi* is the average block length, and *ai* is the number of blocks to allocate. If split cylinder allocation is made, remember that:

- If subparameter 4 is *PRI* and this is the first *EXT* keyword parameter for this file, this option specifies the number of tracks per cylinder to allocate for this member and the number of cylinders for all members in the extent.
- If subparameter 4 is *SUB,* this option sacrifices the number of tracks per cylinder allocated to this member. The number of cylinders is omitted.

`(pi%,ci)`

This is split cylinder allocation only.

- This parameter specifies the number of tracks allocated as a percentage of the total number of tracks per cylinder and, if subparameter 4 is *PRI,* the total number of cylinders to allocate to this extent. If subparameter 4 is *SUB,* the total number of cylinders may be omitted.
- The percent sign (%) must be coded as shown. This option is not used with regular cylinder allocation.

Subparameters 6 through n:

These subparameters are essentially the same as subparameter 5 and are used to describe additional extents, where $n \leqslant 20$. This allows you to specify up to 16 extents.

Subparameter n + 1:

`OLD`

Indicates that this extent applies to a previously allocated file. Primary members specified by this keyword parameter will be allocated by job control and added to the existing file. If this parameter is omitted, the request is for a new extent.

Example 1a:

This example illustrates a SAM disk to SAM disk copy operation using the UDD jproc call statement for a simple disk-to-disk copy. It is assumed that space has been previously allocated for the output file. In this example, the input and output files are on the same disk.

```
      1         10        20        30        40        50
1.  // JOB DCDCCPY1
2.  // UDD IN=(DSPØØ1,LABEL1),OUT=(DSPØØ1,LABEL2,,INIT)
3.  /$
4.  UDD
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCCPY1
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the file and serial number (DSP001) of the disk where the file is located. The *OUT* keyword parameter indicates the file identifier (LABEL2) of the output file and the volume serial number (DSP001) of the disk where the file is located. Since the file is assumed to be previously allocated, *INIT* is used to indicate that any information that may be on the file is to be overwritten by the new information.
3	Indicates the start of data
4	Indicates the data utility control statement. The input and output file type are both defaulted to the input file type specified in the VTOC, and the default block size and record size for input and output are 80.
5	Indicates the end of data
6	Indicates the end of the job
7	Indicates termination of card reader operation

Example 1b:

This example illustrates the expanded job stream generated by example 1a.

```
     1         10        20
1.   // JOB DCDCCPY1
2A.  // DVC 20
2B.  // LFD PRNTR
2C.  // DVC 50
2D.  // VOL DSP001
2E.  // LBL LABEL1
2F.  // LFD INPUT1
2G.  // DVC 50
2H.  // VOL DSP001
2I.  // LBL LABEL2
2J.  // LFD OUTPUT1,,INIT
2K.  // EXEC DATA
3.   /$
4.   UDD
5.   /*
6.   /&
7.   // FIN
```

Line	Explanation
1	Same as for example 1a
2A–2B	These lines assign the printer to the job. The default logical unit number is 20 for this example.
2C–2F	Indicates that the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk with volume serial number DSP001. This information is obtained from the *IN* keyword parameter in line 2 of example 1a. Volume serial number DSP001 is assigned to logical unit number 50.
2G–2J	Indicates that the output file has a file identifier of LABEL2 and a file name of OUTPUT1 and is on the same volume as the input file (DSP001). The same logical unit number is assigned (50). This information is obtained from the *OUT* parameter in line 2 of example 1a.
2K	This line loads the DATA routine from YLOD.
3–7	Same as for example 1a

Example 2a:

This example illustrates a DAM-to-DAM disk copy using the UDD procedure call statement. Space for the output file is allocated to the file in this job (but before execution). The input file is printed in single-space display format as it is being copied.

```
      1         10        20        30        40        50        60                72
1.  // JOB DCDCCPY2
2.  // UDD IN=(DSP001,LABEL1),                                                      X
3.  //1 OUT=(DSP002,LABEL2),                                                       X
4.  //2 EXT=(DA,C,,CYL,3)
5.  /$
6.  UDD  OD,DP
7.  /*
8.  /&
9.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCCPY2
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the file is located.
3	Indicates a continuation of the UDD jproc call. The *OUT* keyword parameter indicates the file label (LABEL2) of the output file and the volume serial number (DSP002) of the disk where the file is located.
4	Indicates further continuation of the UDD procedure call; space for the file is to be allocated on the output disk. This line supplies information required to allocate the file.
5	Indicates the start of data
6	Data utility statement; the disk input file characteristics are taken from the VTOC. The *OD* keyword indicates the output file is a DAM file. The *DP* keyword indicates the input file is to be printed in addition to the output file. The default block and record sizes for input and output are 80.
7	Indicates the end of data
8	Indicates the end of job
9	Indicates the termination of card reader operation

Example 2b:

This example illustrates the expanded job stream generated by example 2a.

```
      1         10        20        30        40        50        60        72
1.    // JOB DCDCCPY2
2A.   // DVC 20
2B.   // LFD PRNTR
2C.   // DVC 50
2D.   // VOL DSP001
2E.   // LBL LABEL1
2F.   // LFD INPUT1
2G.   // DVC 51
2H.   // VOL DSP002
2I.   // EXT DA,C,,CYL,3
2J.   // LBL LABEL2
2K.   // LFD OUTPUT1
2L.   // EXEC DATA
5.    /$
6.    UDD    OD,DP
7.    /*
8.    /&
9.    // FIN
```

Line	Explanation
1	Same as for example 2a
2A–2B	These lines assign the printer to the job. The default logical unit number is 20 for this example.
2C–2F	Indicates that the input file has a file identifier of LABEL1 and a file name of INPUT1 and is on the disk with volume serial number DSP001. This information was obtained from the *IN* keyword in line 3 of example 2a. The volume is assigned to logical unit number 50.
2G, 2H, 2J	Indicates that the output file has a file identifier of LABEL2 and a disk volume serial number of DSP002. This information was obtained from the *OUT* keyword parameter in line 3 of example 2a. The volume is assigned to logical unit number 51.
2I	Indicates that space is to be allocated for a direct access file, and three contiguous cylinders are required. This is obtained from line 4 of example 2a. By default, one cylinder is provided for dynamic extension.
2K	Indicates the file name for the output file is OUTPUT1
2L	Indicates the DATA routine is loaded from YLOD
5–9	Same as for example 2a

Example 3a:

This example illustrates an ISAM-to-ISAM disk copy using the UDD jproc call statement for disk-to-disk copy of an ISAM file. The input and output files have identical file identifiers, but are on different disk volumes. It is assumed that space has been previously allocated for the output file.

```
      1         10        20        30        40        50
1.  // JOB DCDCCPY3
2.  // UDD IN=(DSP001,LABEL1),OUT=(DSP002,LABEL1,,INIT)
3.  /$
4.  UDD B=(80,852),V=(5),W=(30),OI
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates that the name of the job is DCDCCPY3
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the file is located. The *OUT* keyword parameter indicates the file identifier (LABEL1) of the output file and the volume serial number (DSP002) of the disk where the file is located. Since the file is assumed to be previously allocated, *INIT* is used to indicate that any information that may be on the file is to be overwritten by the new information.
3	Indicates the start of data
4	Indicates the data utility control statement. The input file characteristics are taken from the VTOC. The *OI* keyword indicates the output file is also ISAM. The *B* parameter indicates the output record size is 80 and the block size is 852. The *V* parameter indicates the key length is 5, and the *W* parameter indicates a key location of 30 for the output file.
5	Indicates the end of data
6	Indicates the end of job
7	Indicates the termination of card reader operation

Example 3b:

This example illustrates the expanded job stream generated by example 3a.

```
      1         10        20        30
1.  // JOB DCCPY3
2A. // DVC 20
2B. // LFD PRNTR
2C. // DVC 50
2D. // VOL DSP001
2E. // LBL LABEL1
2F. // LFD INPUT1
2G. // DVC 51
2H. // VOL DSP002
2I. // LBL LABEL1
2J. // LFD OUTPUT1,,INIT
2K. // EXEC DATA
3.  /$
4.  UDD B=(80,852),V=(5),W=(30),OI
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Same as for example 3a
2A–2B	These lines assign the printer to the job. The default logical unit number is 20 for this example.
2C–2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk with volume serial number DSP001. This information was obtained from the *IN* keyword in line 2 of example 3a. Volume serial number DSP001 is assigned to logical unit number 50.
2G–2J	Indicates the output file also has a file identifier of LABEL1 and is located on the disk with volume serial number DSP002, which is assigned to logical unit number 51. This information is obtained from the *OUT* keyword parameter in line 2 of example 3a. The file name is OUTPUT1.
2K	Indicates the DATA routine is to be loaded from YLOD
3–7	Same as for example 3a

Example 4a:

This example illustrates a SAM disk-to-disk compare operation using the UDD jproc call statement. In this example, the user chooses to define a printer other than logical unit 20 for the printed output.

```
      1         10        20        30        40        50        60              72
1.  // JOB DCDCOMP
2.  //        UDD    IN=(DSPØØ1,LABEL1),OUT=(DSPØØ2,LABEL2),                      X
3.  //1             COMPARE=YES,PRNTR=21
4.  /$
5.  UDD K2
6.  /*
7.  /&
8.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCOMP
2–3	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the input file is located. The *OUT* keyword parameter gives information on the secondary input file (the file to be compared against the input file) and indicates the file identifier is LABEL2. The volume serial number (DSP002) for the disk containing this file is also given through the *OUT* parameter. The *COMPARE* parameter indicates this is a comparison operation instead of a copy operation. The K2 option must also be selected on the data utility control statement. The *PRNTR* keyword indicates the printer is to be assigned to a logical unit number of 21.
4	Indicates the start of data
5	Indicates the data utility control statement. The K2 option specifies this is a compare operation. All input file characteristics are taken from the VTOC.
6	Indicates the end of data
7	Indicates the end of the job
8	Indicates that card reader operation is terminated

Example 4b:

This example illustrates the expanded job stream generated by example 4a.

```
     1         10        20
1.  // JOB DCDCOMP
2A. // DVC 21
2B. // LFD PRNTR
2C. // DVC 5Ø
2D. // VOL DSPØØ1
2E. // LBL LABEL1
2F. // LFD INPUT1
2G. // DVC 51
2H. // VOL DSPØØ1
2I. // LBL LABEL2
2J. // LFD INPUT2
2K. // EXEC DATA
4.  /$
5.  UDD K2
6.  /*
7.  /&
8.  // FIN
```

Line	Explanation
1	Same as for example 4a
2A–2B	Assigns a printer with logical unit number of 21 to the job
2C–2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk with volume serial number DSP001. This information was obtained from the *IN* parameter. The logical unit number 50 is assigned to DSP001.
2G–2J	Indicates the secondary file has a file identifier of LABEL2 and a file name of INPUT2 and is located on the disk having a volume serial number of DSP002. This information was obtained from the *OUT* and *COMPARE* parameters.
2K	Loads the DATA routine from YLOD
4–8	Same as for example 4a

Example 5a:

This example illustrates an IRAM-to-IRAM disk copy using the UDD jproc call statement for a disk-to-disk copy of an IRAM file. Identical file identifiers are used for the input and output files, but the files are located on different volumes. Space for the output file has been allocated.

```
      1         10        20        30        40        50
1.  // JOB DCDCCPY4
2.  // UDD IN=(DSP003,LABEL2),OUT=(DSP004,LABEL2,,INIT)
3.  /$
4.  UDD B=(80,800),V=(6),W=(30),ORY=(1)
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCCPY4
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL2) of the input file and the volume serial number (DSP003) of the disk where the file is located. The *OUT* keyword parameter indicates the file identifier (LABEL2) of the output file and the volume serial number (DSP004) of the disk where the file is located. The file is assumed to be previously allocated, therefore *INIT* is used to indicate that any information currently on the file is overwritten by the new information.
3	Indicates start of data
4	Indicates the data utility control statement. The input file characteristics are taken from the VTOC. The *ORY=(I)* keyword parameter indicates the output file is indexed and has data management disk write checking. The index block size for the output file defaults to one 256-byte sector. The *B* parameter indicates the output record size is 80 bytes and block size is 800 bytes. The *V* parameter indicates the key length of the output IRAM file is 6. The *W* parameter indicates that the key location of the output IRAM file is 30.
5	Indicates end of data
6	Indicates end of job
7	Indicates termination of card reader operations

Example 5b:

This example illustrates the expanded job stream generated by example 5a.

```
     1         10        20        30        40
1.   // JOB DCPCCPY4
2A.  // DVC 20
2B.  // LFD PRNTR
2C.  // DVC 60
2D.  // VOL DSP003
2E.  // LBL LABEL2
2F.  // LFD INPUT1
2G.  // DVC 80
2H.  // VOL DSP004
2I.  // LBL LABEL2
2J.  // LFD OUTPUT1,,INIT
2K.  // EXEC DATA
3.   /$
4.   UDD B=(80,800),V=(6),W=(30),ORY=(I)
5.   /*
6.   /&
7.   // FIN
```

Line	Explanation
1	Same as for example 5a
2A–2B	These lines assign the printer to the job.
2C–2F	Indicates the input file has a file identifier of LABEL2 and a file name of INPUT1, and is located on the disk with volume serial number DSP003. This information was obtained from the *IN* keyword parameter in line 2 of example 5a. Volume serial number DSP003 is assigned to logical unit number 60.
2G–2J	Indicates output file also has a file identifier of LABEL2 and is located on the disk with volume serial number DSP004, which is assigned to logical unit number 80. This information is obtained from the *OUT* keyword parameter in line 2 of example 5a. The file name is OUTPUT1.
2K	Indicates the DATA routine is to be loaded from YLOD
3–7	Same as for example 5a

5.4.2. UDT Job Control Procedure

Function:

UDT generates the job control statements for your device assignment sets that are required by the data utility routine to copy or compare a disk file to a tape file (disk-to-tape operation).

Format:

```
//ignored UDT  IN=((vol-ser-no|RES|RUN),label[,noext][,ACCEPT]),
                OUT=(vol-ser-no,label)[,PRNTR=(N|((lun[,vol-ser-no])|ZB|N))]
                [,PUNCH=(NO|YES)][,COMPARE=(NO|YES)]
```

Label:

The label field is ignored.

Parameters:

IN=

Supplies the information required to define the input file. This information is coded within the parentheses.

where:

vol-ser-no

Specifies the volume serial number of the disk volume containing the input file.

RES

Specifies that the input file is located on the SYSRES disk.

RUN

Specifies that the input file is on the volume containing the job's YRUN file.

label

Specifies the file identifier for the input file.

noext

Specifies the number of extents in the file for which space must be reserved in the prologue. The default value is 8.

ACCEPT

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

OUT=

Supplies information required to define the output file for a copy operation or a secondary input file for a compare operation. The information is coded within parentheses.

where:

vol-ser-no

Indicates the volume serial number of the disk containing the output (or secondary input) file.

label

Specifies the file identifier for the output (or secondary input) file.

PRNTR=

Indicates whether a device assignment set is to be generated for the printer and defines the print file.

where:

N

Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of LCB, SPL, and VFB job control statements.

{*lun* / **20**}

Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no

Provides a 1- to 6-alphanumeric-character remote destination identifier for the print file when dealing with remote job entry.

PUNCH=YES

Generates a device assignment set for the punch. This function is required when the card punch dual output feature (DC) is selected on the data control card.

PUNCH=NO

No device assignment set is generated for the punch. This is the default value and need not be specified.

COMPARE=YES

Specifies that the compare operation option K2 will be selected on the data control card. This keyword causes the file name INPUT2 (secondary input) to be generated for the file specified by the OUT keyword parameter.

COMPARE=NO

Specifies that this is a copy operation. This is the default value and need not be specified.

Example 1a:

This example illustrates a SAM disk-to-tape copy operation using the UDT jproc call statement for copying a SAM disk to a standard labeled tape. The first 80 characters of each record are to be punched.

```
      1         10        20        30        40        50
1.  // JOB DCTPØ1
2.  // UDT IN=(DSPØØ1,LABEL1),OUT=(SPØØØ1,LABEL2),PUNCH=YES
3.  /$
4.  UDT DC,OR,LO
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCTP01
2	Indicates that the name of the jproc called is UDT. The *IN* parameter indicates the file identifier (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the file is located. The *OUT* keyword parameter indicates the tape file identifier is LABEL2 and tape volume serial number is SP0001. The *PUNCH* parameter indicates that the *DC* keyword parameter (Table 2-1) is used on the data utility control statement; therefore output is also punched.
3	Indicates the start of data
4	This is a data utility control statement indicating the disk input and tape output parameters. The *DC* parameter indicates punched output in addition to tape output. The *OR* parameter indicates the tape is to be rewound before and after processing. The *LO* parameter indicates standard tape labels are used. The default block size and record size for input and output are 80.
5	Indicates the end of data
6	Indicates the end of the job
7	Terminates the card reader operation

Example 1b:

This example illustrates the expanded job stream generated by example 1a.

```
     1         10        20
1.   // JOB DCTPØ1
2A.  // DVC 2Ø
2B.  // LFD PRNTR
2C.  // DVC 5Ø
2D.  // VOL DSPØØ1
2E.  // LBL LABEL1
2F.  // LFD INPUT1
2G.  // DVC 9Ø
2H.  // VOL SPØØØ1
2I.  // LBL LABEL2
2J.  // LFD OUTPUT1
2K.  // DVC 4Ø
2L.  // LFD OUTPUT2
2M.  // EXEC DATA
3.   /$
4.   UDT DC,OR,LO
5.   /*
6.   /&
7.   // FIN
```

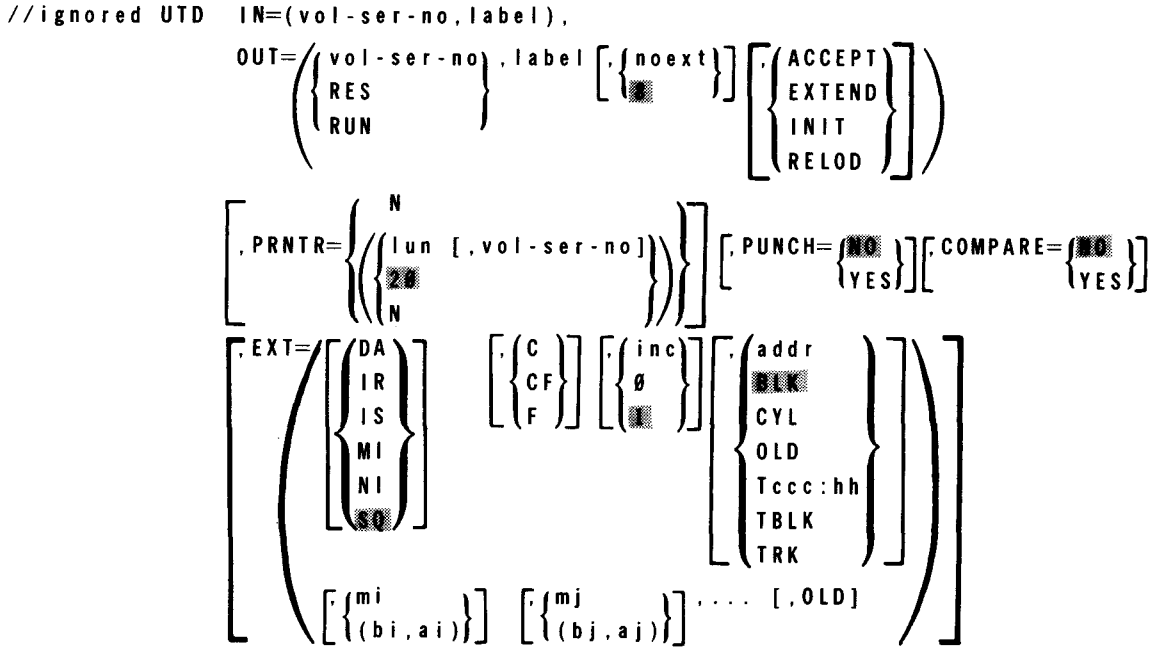
Line	Explanation
1	Same as example 1a
2A–2B	Assigns the printer to this job. The default logical unit number 20 is used.
2C–2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk having volume serial number DSP001. The volume (DSP001) is assigned to logical unit number 50. This information was obtained from the *IN* keyword in line 2 of example 1a.
2G–2J	Indicates the output file has a file identifier of LABEL2, a file name of OUTPUT1, and a tape volume serial number of SP0001. The volume is assigned to logical unit number 90. This information was obtained from the *OUT* keyword in line 2 of example 1a.
2K–2L	Assigns the card punch (logical unit number 40) to the job. The file name is OUTPUT2.
2M	Loads the DATA routine from YLOD
3–7	Same as example 1a

5.4.3. UTD Job Control Procedure

Function:

UTD generates the job control statements for your device assignment sets that are required by the data utility routine to copy or compare a tape file to a disk file (tape-to-disk operation).

Format:



Label:

The label field is ignored.

Parameters:

IN=

Supplies the information required to define the input file. This information is coded within the parentheses.

where:

vol-ser-no

Specifies the volume serial number of the volume containing the input file.

label

Specifies the file identifier for the input file.

OUT=

Supplies the required information to define the output file for a copy operation or secondary input for a compare operation.

where:

vol-ser-no
: Provides the volume serial number of the disk containing the output file (or secondary input).

RES
: Indicates the output (secondary input) file is located on the SYSRES disk.

RUN
: Indicates the output (secondary input) file is located on the volume containing the job's YRUN file.

label
: Specifies the file identifier for the output (or secondary input) file.

no,ext
: Specifies the number of extents in the file to be reserved in the extent table storage for use by the data access method. The default value is 8.

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

EXTEND
: Indicates that a sequential (SQ) file is to be extended. The information is appended to the present end of the file.

INIT
: Indicates that the specified file is to initialize starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file closure time.

RELOD
: Indicates that the specified IRAM or ISAM file is not to be reformatted at file open time when the file is being loaded.

PRNTR=
: Indicates whether a device assignment set is to be generated for the printer and defines the printer file.

where:

N
: Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of SPL, LCB, and VFB job control statements.

{lun / 20}
: Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no
Provides a 1- to 6-character remote destination alphanumeric identifier for the print file when dealing with remote job entry.

PUNCH=YES
Generates a device assignment set for the punch. This is required when the punch dual output feature (DC) is selected on the data utility control card.

PUNCH=NO
No device assignment set is generated for the punch. This is the default value and need not be specified.

COMPARE=YES
Indicates that compare operation option K2 is selected on the data utility card. This keyword generates a file name of INPUT2 (secondary input) for the file specified by the *OUT* parameter.

COMPARE=NO
Specifies that this is a copy operation. This is the default value and need not be specified.

EXT=
Supplies the extent specifications to be used when reserving system resources for an output file. The information is supplied as a subparameter list enclosed in parentheses.

where:

DA
Indicates this is a direct access file.

IR
Indicates this is an IRAM or a MIRAM file.

IS
Indicates this is an ISAM file.

MI
Indicates this is a MIRAM file.

NI
Indicates this is a nonindexed (DA or SQ) file.

SQ
Indicates this is a sequential file and is the default option.

C

Indicates the file must be allocated contiguously.

F

Indicates the file is to be formatted at allocation time. Subparameter 4 must be *BLK*.

CF

Indicates both of the preceding parameters apply to subparameter 2.

NOTE:

If this parameter is omitted (C, F, or CF), none of the previous options apply.

inc

Specifies the secondary increment (in cylinders) by which the file is to be extended if automatic extension is required.

Ø

Indicates that there can be no dynamic extension of the file.

1

Indicates an extension of one cylinder increment of the file.

NOTE:

If there is no EXT keyword parameter for this file, the value of the most recently specified secondary increment is used.

addr

Specifies the absolute cylinder address where the file is to begin. The allocation is in terms of cylinders.

BLK

Indicates that the allocation is in terms of blocks. This is the default parameter.

CYL

Indicates that the allocation is in terms of cylinders.

OLD

Indicates that the secondary allocation increment is changed (subparameter 3). No additional space is allocated by job control. If specified, this must be the last subparameter in the sublist.

Tccc:hh

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

TBLK

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

TRK

Allocates space in tracks.

mi and mj

Specify the number of cylinders to allocate for this file. Subparameter 4 must be *CYL* or *addr.* If split cylinder allocation is made, remember:

- If subparameter 4 is *PRI* and this is not the first *EXT* keyword parameter for this file, this option specifies the number of cylinders for this extent. The number of tracks is assumed to have been specified on the first *EXT* keyword parameter for the primary member.
- If subparameter 4 is *SUB,* this option specifies the number of tracks to be allocated to this secondary member of a split cylinder.

(bi,ai) and (bj,aj)

These options are used when allocation is in terms of blocks, where *bi* and *bj* are the average block length, and *ai* and *aj* are the number of blocks to allocate. If split cylinder allocation is made, remember:

- If subparameter 4 is *PRI* and this is the first *EXT* keyword parameter for this file, this option specifies the number of tracks per cylinder to allocate to this member and the number of cylinders for all members in the extent.
- If subparameter 4 is *SUB,* this option sacrifices the number of tracks per cylinder allocated to this member. The number of cylinders is omitted.

Subparameters 6 through n:

Specified in the same format as subparameter 5 and used to describe additional extents, where $n \leqslant 20$. This allows you to specify up to 16 extents.

Subparameter n+1:

OLD

Indicates this extent applies to a previously allocated file. Primary members specified by this keyword parameter are allocated by job control and added to the existing file. If this parameter is omitted, the request is for a new extent.

Example 1a:

This example illustrates a SAM tape to ISAM disk copy using the UTD procedure call statement. The operation uses variable-length records. It is assumed the disk area has been previously allocated.

```
      1         10        20        30        40        50
1.  // JOB TPDCØ1
2.  // UTD IN=(SPØØ1,LABEL1),OUT=(DSPØØ1,LABEL2,,INIT)
3.  /$
4.  UTD A=(5,89),OI,FV,LO
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is TPDC01
2	Indicates the name of the jproc being called is UTD. The *IN* parameter indicates the tape file identifier is LABEL1 and the tape volume serial number is SP001. The *OUT* keyword parameter indicates that the disk file identifier is LABEL2 and the volume serial number is DSP001. Since the file is assumed to be previously allocated, *INIT* is used to indicate that any information that may be on the file is to be overwritten by the new information.
3	Indicates the start of data
4	Specifies a data utility control statement indicating tape input and disk output. The *FV* parameter indicates that records are of variable length. The *A* parameter indicates the maximum block size is 89, and the minimum record size is 5. With the *B* parameter not specified, maximum block size defaults to the maximum block size for input. The *OI* parameter indicates the output is ISAM. The *LO* parameter indicates the tape has standard labels.
5	Indicates the end of data
6	Indicates the end of the job
7	Terminates the card reader operation

Example 1b:

This example illustrates an expanded job stream generated by example 1a.

```
     1         10        20
1.   // JOB TPDCØ1
2A.  // DVC 2Ø
2B.  // LFD PRNTR
2C.  // DVC 9Ø
2D.  // VOL SPØØ1
2E.  // LBL LABEL1
2F.  // LFD INPUT1
2G.  // DVC 5Ø
2H.  // VOL DSPØØ1
2I.  // LBL LABEL2
2J.  // LFD OUTPUT1,,INIT
2K.  // EXEC DATA
3.   /$
4.   UTD A=(5,89),OI,FV,LO
5.   /*
6.   /&
7.   // FIN
```

Line	Explanation
1	Same as for example 1a
2A–2B	Assigns the printer to the job. The default logical unit number 20 is used.
2C–2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and the tape volume serial number is SP001. It is assigned logical unit number 90. This information is obtained from the *IN* keyword parameter in line 2 of example 1a.
2G–2J	Indicates the output file has a file identifier of LABEL2, a file name of OUTPUT1, and a volume serial number of DSP001. It is assigned logical unit number 50. This information is obtained from the *OUT* parameter in line 2 of example 1a.
2K	Loads the DATA routine from YLOD
3–7	Same as for example 1a

PART 3. FILE PROCESSING IN AN INTERACTIVE ENVIRONMENT

6. Interactive Data Utility

6.1. USING THE UTILITY

The interactive data utility allows you to interactively execute the DATA routine by using mixed or consolidated data management files. This interactive execution consists of a dialog (question and answer session) that you conduct with the DATA routine via a workstation. During the dialog, questions are displayed on the workstation screen and you answer them by typing in the requested information via the workstation keyboard. The answers define the file processing requirements for the DATA routine.

To use the interactive data utility, you should become familiar with keyins and workstation responses. See 7.4 of the screen format services concepts and facilities, UP-8802 (current version), for workstation operator considerations.

When using the interactive data utility on a remote communications terminal, you must have the field protect feature. If this feature is not used, an SF16 error message is issued.

To conduct an interactive dialog, you must first log on the system by entering the LOGON command. After you have successfully logged on, enter the RV I@DATA command at your workstation keyboard. The format of the I@DATA command is:

```
RV  I@ DATA(new-name)[,,MEM=nnnnn][,ACT=act-no][,DBG={Y}]
                                                   {N}
```

where:

RV
Indicates the job is to be run without the use of a card reader (interactively).

I@DATA(new-name)
Specifies the job name for the interactive data utility. If you specify the optional new-name parameter, this name becomes the job name for the utility. Since every job name must be unique, the use of the new-name allows for multiple users.

MEM=nnnnn
Specifies the required amount of main storage (hexadecimal). The default value 8000_{16} ($32{,}767_{10}$) is sufficient for most processing; however, certain functions requiring tape or disk files may require more storage. The formula for determining the minimum amount of main storage is described in 3.4.

ACT=act-no
Indicates a 1- to 4-character alphanumeric function assigned to you for job accounting purposes.

DBG={Y / N}
Specifies debug mode, which is used to provide documentation in reporting a software user report (SUR). Specify Y to run the DATA routine in debug mode. If omitted, N is assumed. The debug mode increases interactive response time.

Once you have successfully initiated the execution, your workstation screen clears and the first menu selection screen is displayed. This screen asks you what you want to do. Depending on your response, the next menu selection screen or a HELP screen (a detailed explanation of the options used in the menu screen) appears.

To make a choice or to request HELP, you must enter the requested information via the workstation keyboard and then press the transmit (XMIT) key.

6.2. OUTPUT LISTINGS

The same types of output listings (printer formats and termination information) that are provided in the batch environment (2.3) are also provided in the interactive environment. The only difference is that the termination information is displayed on the workstation that initiated the dialog.

6.3. TERMINATION INFORMATION

When you execute a program in this environment, the format of the termination information for card, tape, and printer files is:

- INPUT1/OUTPUT1

```
INPUT1/OUTPUT1...(FILENAME).....{CARD  }
                                {TAPE  }
                                {PRINTER}
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
RECORD FORMAT.....{FIXBLK}
                  {FIXUNB}
                  {VARBLK}
                  {VARUNB}
FILE ORG.....SAM
```

- INPUT2

```
INPUT2...(FILENAME)..... {CARD   }
                         {TAPE   }
                         {PRINTER}
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
RECORD FORMAT..... {FIXBLK}
                   {FIXUNB}
                   {VARBLK}
                   {VARUNB}
FILE ORG..SAM   RCDS UNEQUAL.....nnnnnnnn
```

The format of the termination information for MIRAM disk and diskette files is:

- INPUT1/OUTPUT1

```
INPUT1/OUTPUT1..(FILENAME).....DISK
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
MIRAM KEY     LOC      LEN      CHG     DUP
  KEY1       nnnnn    nnnnn    Y/N     Y/N
  KEY2       nnnnn    nnnnn    Y/N     Y/N
  KEY3       nnnnn    nnnnn    Y/N     Y/N
  KEY4       nnnnn    nnnnn    Y/N     Y/N
  KEY5       nnnnn    nnnnn    Y/N     Y/N
INDEX BUFFER SIZE.....nnnnnnn
RECORD CONTROL BYTE.....{YES}
                        {NO }
RECORD SLOT SIZE.....nnnnn
DISK SECTOR SIZE.....nnnnn
VOLUME MOUNT SETTING.....{SINGLE}
                         {MULTI }
RECORD FORMAT.....{FIXBLK}
                  {VARBLK}
FILE ORG.....MIRAM{CONSEC}
                  {INDEX }
                  {MIXED }
```

- INPUT2

```
INPUT2.....(FILENAME).....DISK
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
MIRAM KEY     LOC      LEN      CHG     DUP
  KEY1       nnnnn    nnnnn    Y/N     Y/N
  KEY2       nnnnn    nnnnn    Y/N     Y/N
  KEY3       nnnnn    nnnnn    Y/N     Y/N
  KEY4       nnnnn    nnnnn    Y/N     Y/N
  KEY5       nnnnn    nnnnn    Y/N     Y/N
INDEX BUFFER SIZE.....nnnnn
RECORD CONTROL BYTE.....{YES}
                        {NO }
RECORD SLOT SIZE.....nnnnn
DISK SECTOR SIZE.....nnnnn
VOLUME MOUNT SETTING.....{SINGLE}
                         {MULTI }
RECORD FORMAT.....{FIXBLK}
                  {VARBLK}
FILE ORG...MIRAM{CONSEC}...RCDS UNEQUAL...nnnnnnnn
               {INDEX }
               {MIXED }
```

6.4. ERROR MESSAGES

Error messages are handled the same as in the batch environment (2.4), except that they also are displayed on the workstation that initiated the dialog. They assume the default value of // PARAM DISPLAY= that is on the printer and system log file, as well as the initiating workstation.

6.5. SAMPLE INTERACTIVE DIALOG

A typical tape-to-tape copy operation is provided in the following description. First, we must log on and ensure that the workstation is in system mode and that the input tape file is a single-volume file with standard labels in EBCDIC mode. We assume that in this example the file has a fixed block length of 800 bytes, with each record having 80 bytes; the volume serial number is TAPE01; and the file identifier is TFILE1.

The output tape file is a single-volume file having standard labels; it is in EBCDIC mode. Also, the block and record sizes are the same as for the input tape file. We have a variable-length block of 800 bytes and a record length of 80 bytes. The volume serial number is TAPE02, with the file identifier of TFILE2. The tape is to be rewound before and after processing.

Once you know your input and output file specifications, you can start the interactive file processing utility via the following command:

```
RV I@DATA(DATAUT1)[,,MEM=A000,ACT=PUBS]
```

This command calls the interactive data utility and assigns 40,960 decimal bytes (A000) for the job. The job name is DATAUT1, allowing other users to use the utility. The account number is PUBS. (We did not specify the debug mode; thus, no snap dumps are provided in case of a hardware or software failure.)

We now go through the tape-to-tape copy operation in the following steps:

NOTE:

In the workstation screens used for this description, the entries that appear in shaded typeface (1) are default values. The entries that appear in reverse print (2) are entered by the user. Terms in parentheses are protected fields and are not keyed in by the user.

Step 1: Indicating the Operation

After the RUN command is processed, the first menu selection screen is displayed:

```
SCREEN 1 DUS01
DO YOU WISH TO
1.  COPY OR PRINT A FILE
2.  COMPARE TWO FILES
3.  CONVERT OS/4 FILES TO OS/3 DISK FILES
4.  CONVERT A S/32-34 $COPY DISKETTE
5.  HELP
ENTER (1 THRU 5) (1)
```

Since you want a tape-to-tape copy operation, press the transmit key (XMIT) (because 1 is the default).

If you want a compare operation, a conversion operation, or need HELP, enter a 2, 3, 4, or 5 (overwriting 1) and press the transmit key.

Step 2: Choosing Primary File Type

After you have indicated the type of operation, the following screen is displayed:

```
INPUT SCREEN 1 DUSØ2
PLEASE ENTER THE TYPE OF YOUR PRIMARY FILE
1.  CARD
2.  TAPE
3.  DISKETTE
4.  DISK
ENTER (1 THRU 4) (2)
```

Because your input file type is a tape, overwrite the default (1) with a 2 and press the transmit key.

Step 3: Defining Input Record and Block Characteristics

After you choose the primary file type, the following screen is displayed.

```
TAPE SCREEN 2 DUSØ5
PLEASE ENTER RECORD LENGTH, BLOCK LENGTH, RECORDING MODE,
AND RECORD FORMAT
1.  RECLEN (ØØØ8Ø)
2.  BLKLEN (Ø08ØØ)
3.  RECORDING MODE (E=EBCDIC OR A=ASCII) (E)
4.  RECORD FORMAT (FIXBLK, FIXUNB, VARBLK, OR VARUNB) (VARUNB)
5.  HELP (ENTER ITEM NUMBER OR 5 FOR ALL) (_)
```

Since you have already chosen the input tape file specifications, enter the appropriate specifications in items 1 through 4. Since you are using the variable unblocked record format, choose the smallest record length (80) and the largest block length (800).

Step 4: Indicating Input Tape Rewind Option

After you choose the record and block characteristics, the following screen is displayed:

```
TAPE SCREEN 3 DUSØ6
PLEASE ENTER YOUR TAPE REWIND OPTION
1.  REWIND BEFORE AND AFTER PROCESSING
2.  REWIND BEFORE AND UNLOAD TAPE AFTER PROCESSING
3.  REWIND BEFORE BUT NOT AFTER PROCESSING
4.  DO NOT REWIND BEFORE BUT REWIND AFTER PROCESSING
5.  DO NOT REWIND BEFORE BUT UNLOAD TAPE AFTER PROCESSING
6.  DO NOT REWIND BEFORE OR AFTER PROCESSING
7.  HELP
ENTER(1)
```

Because you want to rewind the input tape before and after processing, overwrite the default (6) with a 1 and press the transmit key.

NOTE:

Only press the transmit key after making a selection, or a series of selections, on a menu selection screen.

Step 5: Selecting the Input Tape Label Specification

After indicating your tape rewind option, the following screen appears:

```
TAPE SCREEN 4 DUSØ7
PLEASE ENTER TAPE LABEL SPECIFICATION
1.  STANDARD LABELS
2.  STANDARD AND USER LABELS
3.  USER LABELS
4.  NO LABELS
5.  HELP
ENTER(1)
```

Since you are using standard labels, enter a 1 and press the transmit key.

Step 6: Choosing Tape and File Specifications

After you choose the input tape label specification, the following screen is displayed:

```
TAPE SCREEN 1 DUSØ4
PLEASE ENTER TAPE VOLUME SERIAL NUMBER, FILE NAME,
AND FILE SEQUENCE NUMBER
1.  VSN (TAPEØ1)
2.  FN (TFILE1)
3.  FSN(1)
4.  HELP (ENTER ITEM NUMBER OR 4 FOR ALL)(_)
```

Here, you enter the tape volume serial number TAPE01 and its file name TFILE1. Since it is a single volume file, 1 is assumed. Press the transmit key for the next screen.

Step 7: Specifying Input File Catalog Password

After you have defined your tape file characteristics, this screen appears:

```
CATALOGUE SCREEN 1 DUS54
1.  IF THE INPUT1 FILE IS CATALOGUED AND HAS
    A READ PASSWORD, PLEASE SPECIFY THE
    PASSWORD.
       (______)
2.  NEED HELP (Y OR N)   (_)
```

Since there is no password associated with your file, press the transmit key.

Step 8: Specifying Multiple Input Volume Serial Numbers

After you specify your catalog password, the following screen is displayed:

```
MULTI-VOLUME SCREEN 1    DUS55
 PLEASE SPECIFY IN ORDER OF PROCESSING THE VSN NAME(S) OF THE
 ADDITIONAL VOLUMES YOU WISH DATA UTILITIES TO PROCESS.
  VSN2:  (______)      VSN3:  (______)     VSN4:  (______)
  VSN5:  (______)      VSN6:  (______)     VSN7:  (______)
  VSN8:  (______)      VSN9:  (______)    VSN10:  (______)
 VSN11:  (______)     VSN12:  (______)    VSN13:  (______)
 VSN14:  (______)     VSN15:  (______)    VSN16:  (______)
IF YOU NEED HELP PLEASE SPECIFY H FOR HELP. (_)
```

Since you wish to process only one volume, press the transmit key.

Step 9: Selecting Output File Type

After you specify your multiple volume serial numbers, this screen appears:

```
OUTPUT SCREEN 1 DUS24
PLEASE ENTER YOUR OUTPUT FILE TYPE
1.  PRINTER
2.  CARD
3.  TAPE
4.  DISKETTE
5.  DISK
ENTER(1)
```

Since you want a tape output file, overwrite the default (1) with 3 and press the transmit key.

Step 10: Specifying File Option

After you specify your output file type, the following screen is displayed:

```
OUTPUT SCREEN 2 DUS50

ENTER FILE EXTENSION OR INITIALIZATION
OPTION (APPLIES TO TAPE,DISKETTE, OR
DISK OUTPUT ONLY):
1. FILE TO BE INITIALIZED
2. FILE TO BE EXTENDED
ENTER (1 OR 2) (_)
```

Since your file is to be initialized, enter a 1 and press the transmit key.

Step 11: Defining Output Record and Block Characteristics

After you indicate your file option, this screen appears:

```
TAPE SCREEN 2 DUS05

        PLEASE ENTER RECORD LENGTH, BLOCK LENGTH, RECORDING MODE,
        AND RECORD FORMAT
        1. RECLEN 00080
        2. BLKLEN 00800
        3. RECORDING MODE (E=EBCDIC OR A=ASCII)  E
        4. RECORD FORMAT (FIXBLK, FIXUNB, VARBLK, OR VARUNB)  VARUNB
        5. HELP (ENTER ITEM NUMBER OR 5 FOR ALL) _
```

Since you have chosen these characteristics earlier, you may now enter them: Assume the default of 80 for item 1. Overwrite the default in item 2 with 800 (since these are the smallest record length and largest block length). Overwrite the default in item 3 with E. Indicate VARUNB in item 4. Press the transmit key.

Step 12: Indicating Output Tape Rewind Option

After you have input the record and block characteristics of your output tape, the following screen is displayed:

```
TAPE SCREEN 3 DUSØ6

          PLEASE ENTER YOUR TAPE REWIND OPTION
          1. REWIND BEFORE AND AFTER PROCESSING
          2. REWIND BEFORE AND UNLOAD TAPE AFTER PROCESSING
          3. REWIND BEFORE BUT NOT AFTER PROCESSING
          4. DO NOT REWIND BEFORE BUT REWIND AFTER PROCESSING
          5. DO NOT REWIND BEFORE BUT UNLOAD TAPE AFTER PROCESSING
          6. DO NOT REWIND BEFORE OR AFTER PROCESSING
          7. HELP
          ENTER 6
```

Because you want to rewind the output tape before and after processing, overwrite the default with a 1 and press the transmit key.

Step 13: Selecting the Output Tape Label Specification

After you indicate your tape rewind option, the following screen appears:

```
TAPE SCREEN 6 DUSØ9

          PLEASE ENTER YOUR OUTPUT TAPE LABEL SPECIFICATION
          1. STANDARD LABELS
          2. NO LABELS
          3. HELP
          ENTER 1
```

Since you are using standard labels on your output tape, enter a 1 and press the transmit key.

Step 14: Choosing Output Tape and File Specifications

After you choose the output tape label specification, the following screen is displayed:

```
TAPE SCREEN 1 DUSØ4
PLEASE ENTER TAPE VOLUME SERIAL NUMBER, FILE NAME,
AND FILE SEQUENCE NUMBER
1.  VSN(______)
2.  FN(_________________)
3.  FSN(1)
4.  HELP (ENTER ITEM NUMBER OR 4 FOR ALL)(1)
```

Assuming that you request HELP on item 1, enter a 1 in item 4. The following HELP screen is displayed:

```
                    * * * TAPE HELP DUHØ411 * * *
'VSN' SPECIFIES THE VOLUME NUMBER (VOL1 LABEL) ON THE TAPE VOLUME TO
BE USED AS YOUR PRIMARY/SECONDARY FILE.  THE VOLUME SERIAL NUMBER
UNIQUELY IDENTIFIES THE TAPE REEL TO THE OPERATING SYSTEM.  IT IS
WRITTEN INTERNALLY (ON THE TAPE SURFACE) AND POSSIBLY EXTERNALLY
(GENERALLY ON A GUMMED LABEL).  THE VSN CANNOT BE MORE THAN SIX
ALPHANUMERIC CHARACTERS AND THE FIRST CHARACTER MUST BE ALPHANUMERIC.
IF THERE ARE LESS THAN SIX, TRAILING BLANKS WILL BE ADDED ON THE
RIGHT.
NEED MORE(IF ANY) HELP SCREENS? (Y/N=CONTINUE NORMAL PROCESSING)(_)
```

This HELP screen gives you a description of the volume serial number and defines its purpose. If you understand the explanation, press the transmit key to continue normal processing. The screen prior to the HELP screen is again displayed. If you need more help, enter Y and another HELP screen is displayed. Here, we take the default by pressing the transmit key.

```
TAPE SCREEN 1 DUSØ4
PLEASE ENTER TAPE VOLUME SERIAL NUMBER, FILE NAME,
AND FILE SEQUENCE NUMBER
1.  VSN (TAPEØ2)
2.  FN (TFILE2)
3.  FSN (1)
4.  HELP (ENTER ITEM NUMBER OR 4 FOR ALL) (_)
```

Now, we can enter the information for the output tape and file specifications. In item 1, TAPE02 is entered for the volume serial number; in item 2, the file name TPFILE2 is entered. In item 3, because it is a single volume file, the default (1) is accepted. Press the transmit key for the next screen.

Step 15: Specifying Output File Catalog Password

After you have chosen your output tape and file specifications, this screen appears:

```
CATALOGUE SCREEN 1 DUS54

1.  IF THE OUTPUT1 FILE IS CATALOGUED AND HAS
    A WRITE PASSWORD, PLEASE SPECIFY THE
    PASSWORD.
       (______)
2.  NEED HELP (Y OR N)   (_)
```

Since there is no password associated with your file, press the transmit key.

Step 16: Specifying Multiple Output Volume Serial Numbers

After you specify your catalog password, the following screen is displayed:

```
MULTI-VOLUME SCREEN 1    DUS55
 PLEASE SPECIFY IN ORDER OF PROCESSING THE VSN NAME(S) OF THE
 ADDITIONAL VOLUMES YOU WISH DATA UTILITIES TO PROCESS.
  VSN2:  (______)      VSN3:  (______)     VSN4:  (______)
  VSN5:  (______)      VSN6:  (______)     VSN7:  (______)
  VSN8:  (______)      VSN9:  (______)    VSN10:  (______)
 VSN11:  (______)     VSN12:  (______)    VSN13:  (______)
 VSN14:  (______)     VSN15:  (______)    VSN16:  (______)
IF YOU NEED HELP PLEASE SPECIFY H FOR HELP. (_)
```

Since you wish to output to only one volume, press the transmit key.

Step 17: Selecting Additional Options

After you specify your multiple output volume serial numbers, this screen appears:

```
OPTION SCREEN 1 DUS25
PLEASE SELECT (Y OR N) ANY OPTIONS YOU REQUIRE
1.  FILE REPOSITIONING                          (N)
2.  HALT ON RECORD COUNT                        (N)
3.  CORRECTION BY RECORD NUMBER                 (N)
4.  SELECTION/DELETION BY FIELD VALUE           (N)
5.  SEQUENCE CHECKING                           (N)
6.  REARRANGE/CONVERT FIELDS                    (N)
7.  SEQUENCE NUMBERING                          (N)
8.  HELP (ENTER ITEM NUMBER OR 8 FOR ALL). (_)
```

Since you don't require any additional options, press the transmit key.

Step 18: Selecting Data Card Option

After you have indicated you do not desire additional options, this screen appears:

```
DUAL CARD SCREEN 1 DUS45
DO YOU WISH DATA TO PUNCH A CARD FILE CONTAINING THE
FIRST 80 BYTES OF EACH SECONDARY FILE RECORD


ENTER (Y OR N) (N)
```

Assume the default value (N) and press the transmit key.

Step 19: Indicating Print Option

After you have selected your data card option, the print option screen appears.

The following screen is always the last screen before the end-of-job screen. It asks you whether or not your output file is to be printed.

```
DUAL PRINT SCREEN 1 DUS46
DO YOU WISH DATA TO PRINT YOUR SECONDARY FILE
AS IT IS CREATED

ENTER (Y OR N) (N)
```

To take the default (N), press the transmit key.

Step 20: End-of-Job Information

After you indicate whether or not you would like your output file printed, this screen appears. Since the interactive dialog was successful, the interactive data utility displays the following end-of-job information screen:

```
EOJ SCREEN 1 DUS49
CONVERSATIONAL PHASE OF DATA UTILITIES COMPLETED
....EXECUTION STARTED....
```

Once this screen is displayed, file processing begins and the workstation returns to system mode and is ready for the next user.

PART 4. LOADING LARGE, MULTIKEYED MIRAM FILES IN A BATCH ENVIRONMENT

7. MILOAD Utility – Special Purpose Loader for MIRAM Files

7.1. MILOAD – WHY YOU NEED IT

Until now, loading large, multikeyed MIRAM files by using conventional loading techniques resulted in long load times due to the number of I/O operations required to create the dynamic indexes for these files. You can significantly increase performance by using MILOAD, the more efficient fast loader utility program, whenever you need to load a large, multikeyed MIRAM file. There are, however, some considerations, restrictions, and trade-offs to consider before using this utility.

7.1.1. Considerations before Using MILOAD

- Your Operating Environment

  MILOAD only operates in a consolidated data management operating environment (CDM). It uses CDM to read and write the files it processes. Therefore, as a Series 90 user, your system must support a CDM or MIXED mode environment to use the MILOAD utility.

- User Responsibilities for Input Files

  As a user, you are responsible for preparing the input files processed by MILOAD. You must make certain that these files consist only of records that can be successfully added to the MIRAM file being created. For example, if you instruct MILOAD not to accept records that contain duplicate keys, then the input file to MILOAD must not contain records having duplicate keys. In addition, records containing illegal duplicate keys must be removed (sorted) from the input to MILOAD. To do this, use the standard OS/3 SORT program. You are also responsible for defining the criteria for record selection. Your input files can reside on disk or tape; disk files must be IRAM or MIRAM characteristic type files, and tape files must be created by OS/3 data management in EBCDIC mode.

- Output File Structure

  The output file created by MILOAD is always a MIRAM characteristic file. It must include one or more of the following criteria:

  - More than one key
  - Duplicate keys
  - Keys that may be changed during update operations
  - A variable file record format
  - A record control byte (RCB)

  When creating the output file MILOAD, we recommend that you avoid having data management automatically compute the physical sector size (PSS) for data partitions (VSEC=YES). The reason for this recommendation is that MILOAD uses large buffers to achieve performance; by specifying VSEC=YES, you create a situation where every program that later accesses the output file must contend with the inconvenience of large buffers. Therefore, performance is degraded. You may, however, specify a value for the PSS (VSEC=n), but make certain that you choose a value that is comfortable to work with throughout the life of the file. For additional information about VSEC, see the consolidated data management concepts and facilities manual, UP-8825 (current version).

- Need for Temporary Work Files

  MILOAD requires temporary disk work files to perform the sort process, which is an integral part of its operation. The amount of work file space required depends on the number of records involved and the key characteristics of the output file. The algorithm for calculating the amount of work file space needed for a given job is found in a later part of this section.

- Hardware Requirements

  MILOAD supports the following disk devices: 8418, 8430, and 8433.

7.1.2. Restrictions

MILOAD will only run in a CDM or MIXED mode environment.

The output file is not usable until MILOAD successfully completes. Logical errors, hardware errors, and asynchronous termination leaves the output file that is created by MILOAD in a compromised state. It is your responsibility to ensure the successful termination of MILOAD.

Sharing the output file with other jobs is not permitted. Use of the output file must be exclusive until MILOAD successfully loads all the records of a file.

7.1.3. Trade-Offs

The primary objective of MILOAD is to increase performance when loading large, multikeyed MIRAM files. Achieving this goal affects flexibility, temporary disk file space, and main storage requirements. For example, MILOAD does not provide for record selection criteria and terminates abnormally for any hardware or logical error condition. Data-related errors such as illegal duplicate keys are not recoverable when using MILOAD. The input file cannot contain any records with duplicate keys for those output key structures that do not permit duplicates.

MILOAD requires temporary disk file space for use by the SORT routine. This temporary disk file space could become a significant amount if the ratio of key data to record length is high. It could approach the size of the file being loaded.

MILOAD requires 65k of main storage plus an additional amount for the sort work space and buffers. The total amount of main storage needed to execute MILOAD must be 85k bytes or greater.

7.2. USING MILOAD FOR CREATING MIRAM CHARACTERISTIC FILES

After reading Section 7.1, if you determine that you have the necessary resources and main storage to use MILOAD, you can easily put it to work loading your large, multikeyed MIRAM files. Your interface with MILOAD is through the OS/3 job control language (JCL).

The JCL assignment sets for your input, output, scratch, and print file devices are consistent with those used for running any job in an OS/3 environment. The control statements that define your file characteristics and execute MILOAD are compatible with the input and output statements used by the DATA routine. (See 3.1 and Table 3–1.) Examples are provided to assist you in preparing device assignment sets and run control statements.

7.2.1. Device Assignment Sets for Defining Your Files

Your MILOAD job stream must include device assignment sets for all your input, output, scratch, and print files.

- Input File

  The input file to MILOAD, as previously mentioned, can reside on disk or tape. You are required to identify the device type, its volume serial number, and the name of the file containing the input records for processing; you must also define the logical file name as INPUT1. The following examples illustrate the device assignment sets for disk and tape input files of your job stream:

Disk Input	Tape Input
// DVC 50	// DVC 90
// VOL vsn	// VOL vsn
// LBL filename	// LBL filename
// LFD INPUT1	// LFD INPUT1

- Output File

The output file created by MILOAD can only be a MIRAM characteristic file and must reside on disk. As with the input file, you must identify the device type to which the output is written, its volume serial number, and the name of the file to contain the output records. You must also specify the logical name of the file as OUTPUT1. The device assignment set for your output file is:

```
// DVC 50
// VOL vsn
// EXT MI,,,CYL,mi
// LBL filename
// LFD OUTPUT1,,INIT
```

In this example, the INIT parameter indicates that the output file is a new file. You would omit this parameter from the LFD statement if the output file is an existing file. The EXT statement is required only if file space has not been previously allocated.

If you want the output file created with recovery included, add a // DD RECV=YES to your assignment set.

- Scratch File

MILOAD requires temporary work file space on disk for use by the SORT routine. The device assignment set for the scratch file not only identifies the device type, vsn, filename, and logical description; it also allocates the amount of temporary space required. This is a typical example of the device assignment set to be included in your job stream for a scratch file:

```
// DVC 50
// VOL vsn
// EXT ST,,,CYL,mi
// LBL $SCR1
// LFD DM01
```

Note that the file label is specified as $SCR1, and the logical file description is DM01. The extent statement (EXT) is required to allocate temporary space for the scratch file, and the mi parameter specifies the number of cylinders to be allocated. Finally, the specific amount of work space you need for MILOAD depends on:

- number of key structures involved;

- maximum (longest) key length; and
- total number of records being loaded.

Use the following algorithm to determine the amount of work space you need to run MILOAD:

$$S = \frac{(A)\ (B)\ (C)}{D}$$

where:

S
: Is the amount of space needed in terms of 256-byte sectors.

A
: Is the size (maximum length) of the largest key plus 5.

B
: Is the number of key structures.

C
: Is the total number of records being loaded.

D
: Is the physical sector size (256) of the scratch disk.

To convert the space (S) value into cylinders, divide S by the number of blocks per track. Then divide the resultant quotient by the number of tracks per cylinder for your particular scratch disk. Add 20 percent as a size factor.

- Printer File

  You must assign a printer to your job in order to run MILOAD. The device assignment set for the printer is as follows:

```
// DVC 20
// LFD PRNTR
```

7.2.2. Control Statements for Running MILOAD

The control statements for running MILOAD are compatible with those used by the DATA routine for copying MIRAM files. That is, MILOAD uses the input and output utility statement (Uio) and several of its keyword parameters (A, B, FV, L_c, MK_n, and OM) to accomplish the load operation you want performed. These statements define the type of load operation, the input and output device types, and the characteristics of the files being processed.

The Uio statement and its parameters must be included between the start-of-data (/$) and end-of-data (/*) statements, which follow // EXEC MILOAD in your job stream.

A brief description of the Uio statement and its parameters follows. If you need a more detailed explanation of their function, refer to Section 3.

- Uio Statement

  The Uio statement identifies the device types you require for input and output. It is a mandatory statement that you specify in one of two forms:

  UDD
  Indicates that MILOAD is to perform a disk input to disk output operation.

  UTD
  Indicates that MILOAD is to perform a tape input to disk output operation.

  This statement must begin in column 2 or greater and need only be specified once in your job stream. At least one keyword parameter must be included on the same line of code as the Uio statement. For example:

```
UDD A=(r,b)
    B=(r,b)
```

  Otherwise, MILOAD generates an error.

- A=(r,b) Keyword Parameter

  The A parameter specifies the record size (r) and the buffer size (b) of your input file. This parameter is mandatory if your input file resides on tape. You need not specify this parameter for disk input files. In which case, the record size from the format labels is used; the buffer size is essentially ignored because MILOAD optimizes buffer size to achieve better performance.

  When your input tape file contains variable-length records, you must specify r as the minimum record length and b as the maximum block length.

- B=(r,b) Keyword Parameter

  The B parameter is optional. When used, it defines the record size (r) and the buffer size (b) for your output file. If omitted, MILOAD uses the record size of your input file as the default value, essentially ignoring the buffer size since it optimizes buffer size for performance. When specifying r, do not include the record control block (RCB) as part of this value.

- FV=(n) Keyword Parameter

  The FV parameter is also optional. When specified, it indicates two things:

  - The record format is variable length.
  - The file slot size (n)

Make certain that the slot size (n) you specify is large enough to hold the largest expected input record.

If omitted, MILOAD assumes a fixed record length format.

- L Keyword Parameter

The L keyword parameter provides MILOAD with tape input label information. It is mandatory only when your input is on tape. You can specify this parameter in one of two forms:

`LØ`
Indicates standard tape labeled input.

`LC`
Indicates unlabeled tape input.

- MKn = (len,loc[,DUP][,CHG]) Keyword Parameter

The MK parameter is mandatory. It defines the key specifications for your output file. Use it to identify the length and location of each key and to specify whether duplicate keys and key changes are permitted for particular key structures.

You must include at least one key specification in your MILOAD job stream, but you are limited to a maximum of five specifications (MK1 through MK5) for any one MILOAD run. If you specify more than one key specification, you must define them consecutively in ascending order beginning with MK1, MK2, and so on.

You can specify a key length (len) from 1 to 80 bytes. The location of that field (loc) is also expressed in bytes. However, the value specified represents the number of bytes preceding the key field in the record, relative to zero. The value you specify for loc must be within the specified record length and must be less than 32,767 (maximum for MIRAM files). Keep in mind that if your input records are of variable length, the 4-byte header must also be considered to determine the actual key location.

If you want to allow duplicate keys for the particular structure defined, include the DUP subparameter in your specification. Likewise, if you want key changes permitted during updates, include the optional CHG parameter.

- OM=(I,n[,V][,R]) Keyword Parameter

The OM parameter is mandatory. You use it to define the characteristics of your output file, i.e., via its subparameters, define the size of the index buffer needed to build your multikeyed output file, define the output as a multivolume file, and specify whether the output file is to contain a record control byte (RCB) for each record.

You must always include the I subparameter as part of this specification. It identifies your output file as being indexed.

The n subparameter is required to specify the size of the index buffer used to build your multikeyed output file. Buffer size is expressed as the number of 256-byte sectors you need to build your file. You can indicate from 1 to 10 sectors.

The V subparameter indicates that the output file created is a multimount file (all volumes must be online). Because MILOAD only creates multimount files, V (the default) is always used. Therefore, you can ignore this subparameter.

The R subparameter determines whether the output file contains an RCB for each record. When R is specified, MILOAD creates the output file *without* the RCB included. When R is omitted, the output file is created *with* the RCB included.

The RCB should be included in the output file if the programs that later access this newly created file are concerned with record deletion.

7.2.3. Sample Job Stream for Running MILOAD

In the job stream that follows you will note that the input, output, and scratch files are located on separate devices. This is not a requirement, but is a performance consideration because it reduces unnecessary disk-head movement.

```
// JOB MILOAD,,18000                    Job statement
// DVC 20 // LFD PRNTR                  Printer file
// DVC 61 // VOL DMTST1
// LBL MILOAD.INPUT                     Input file
// LFD INPUT1
// DVC 62 // VOL DMTST2
// LBL MILOAD.OUTPUT                    Output file
// LFD OUTPUT1
// DVC 50 // VOL REL071
// EXT ST,,,CYL,30                      Scratch file
// LBL $SCR1
// LFD DM01
// EXEC MILOAD                          MILOAD execution call
/$
 UDD     A=(255,1024)                  Input file record and buffer sizes
         B=(255,1024)                  Output file record and buffer sizes
         OM=(I,1)                      Output file index buffer size
         MK1=(4,8,DUP,CHG)             Primary (first) key specification
         MK2=(13,6,DUP,CHG)
         MK3=(22,10,DUP,CHG)           Additional key specification
/*
/&
// FIN
```

7.2.4. Output Listing Produced by MILOAD

In addition to creating your MIRAM characteristic output file, MILOAD generates an output listing after it successfully terminates. This listing provides you with:

- a statistical summary of the number of records loaded and the number of duplicate keys created for each key structure;
- the characteristics of the output file created including key specifications for all key structures, record size, and buffer size; and
- device and file identification that includes the input device type, output disk type, input and output device volume serial numbers, and input and output file identifiers.

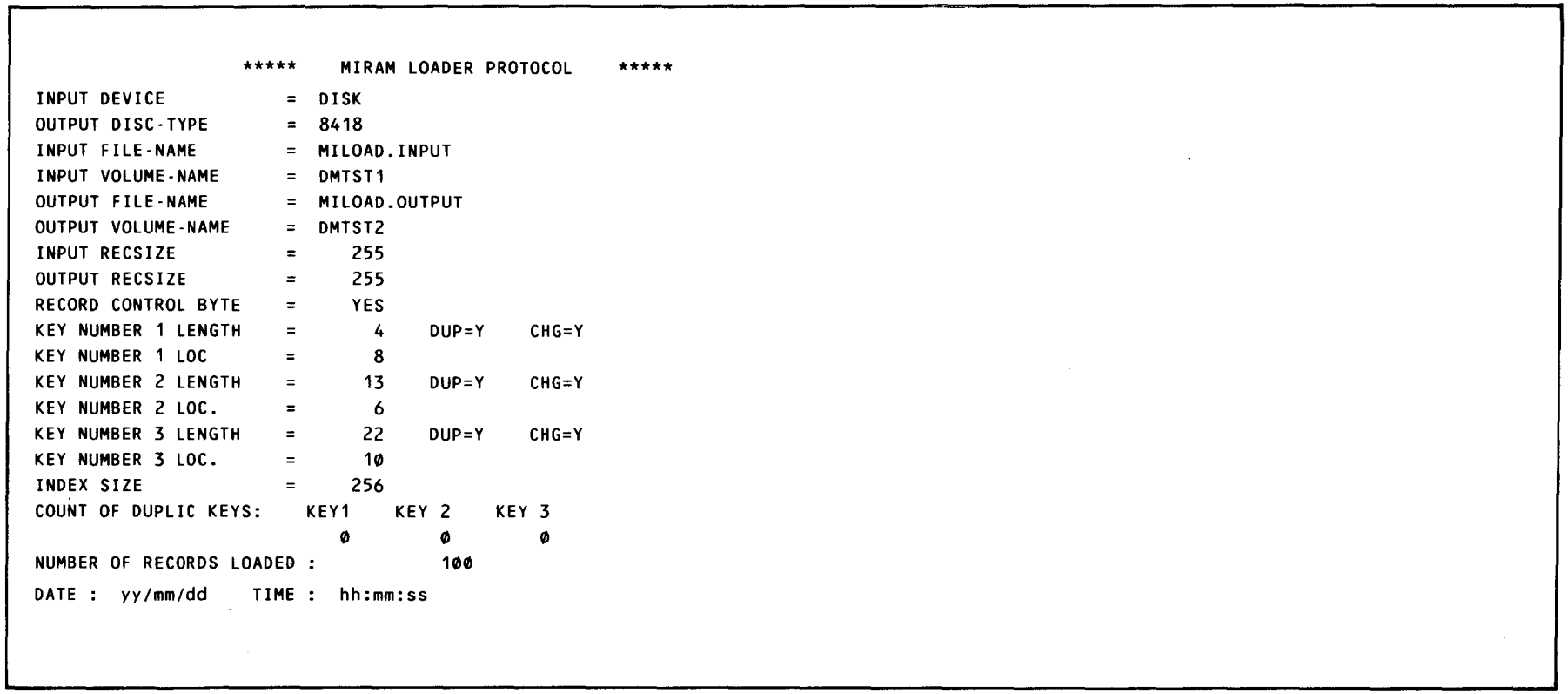
Figure 7–1 is a typical example of an output listing produced when MILOAD successfully terminates.

```
JC06    USING    DEV=301  VSN=UMTST1
JC01    JOB MILOAD3  EXECUTING JOB STEP MILOAD00 #002 00:06:19
START INDEX SORT
END OF INDEX-SORT
01. KEY INSERT STARTED
02. KEY INSERT STARTED
03. KEY INSERT STARTED
 SORT MI00 END OF SORT
SORT A186 RECORDS IN          300 RECORDS DELETED        0
****** NUMBER OF RECORDS LOADED BY MILOAD :        100
AC10  LFD - PRNTR  , FORM NAME - STAND1  , COPIES - 0001, PAGES - 00000001, STEP =002
AC11 STEP #002  (MILOAD00) USED 00064478 BYTES   ELAPSED WALL CLOCK TIME=00:00:53#336  TOTAL SVC CALLS=00001660
AC12    TERM CODE=000       SWITCH-PRIORITY=10   CPU TIME USED          =00:00:40#671  TRANSIENT CALLS=00000079
AC13 UPSI SETTING X'00'
AC19            DEVICE EXCP's          =00000486       =00000497
AC21  JOB  TOTALS      USED  00064478  BYTES        TOTAL  ELAPSED  WALL  CLOCK  TIME=00:01:13#128   AC22   TOTAL  JOB  SVC  CALLS=00002331

AC22                                WALL CLOCK TIME OF ALL STEPS =00:01:06#319  JOB TRANSIENT CALLS=00000131
AC23                                   TOTAL CPU TIME OF ALL STEPS  =00:00:47#227   TOTAL JOB EXCP'S =00001232

JC02 JOB MILOAD  TERMINATED NORMALLY                hh:mm:ss
A=(255,1024),B=(255,1024)
OM=(I,1)
MK1=(4,8,DUP,CHG)
MK2=(13,6,DUP,CHG)
MK3=(22,10,DUP,CHG)
```

Figure 7–1. Typical Output Listing for MILOAD (Part 1 of 2)



```
                *****    MIRAM LOADER PROTOCOL    *****
INPUT DEVICE           =  DISK
OUTPUT DISC-TYPE       =  8418
INPUT FILE-NAME        =  MILOAD.INPUT
INPUT VOLUME-NAME      =  DMTST1
OUTPUT FILE-NAME       =  MILOAD.OUTPUT
OUTPUT VOLUME-NAME     =  DMTST2
INPUT RECSIZE          =     255
OUTPUT RECSIZE         =     255
RECORD CONTROL BYTE    =     YES
KEY NUMBER 1 LENGTH    =       4    DUP=Y    CHG=Y
KEY NUMBER 1 LOC       =       8
KEY NUMBER 2 LENGTH    =      13    DUP=Y    CHG=Y
KEY NUMBER 2 LOC.      =       6
KEY NUMBER 3 LENGTH    =      22    DUP=Y    CHG=Y
KEY NUMBER 3 LOC.      =      10
INDEX SIZE             =     256
COUNT OF DUPLIC KEYS:    KEY1    KEY 2    KEY 3
                            0        0        0
NUMBER OF RECORDS LOADED :           100
DATE :  yy/mm/dd    TIME :  hh:mm:ss
```

Figure 7–1. Typical Output Listing for MILOAD (Part 2 of 2)

7.2.5. Examples of Typical MILOAD Jobs

The following examples give three variations of using MILOAD for loading large multikeyed MIRAM files.

- Example 1: Disk Input to Disk Output (Multidrive)

  In this example, MILOAD creates a 2-key output file with an RCB. Duplicate key and key changes are not allowed. The file contains fixed-format records. The record size is 256, and the input and output files are on separate volumes.

```
// JOB MILOAD1,,18000                          Job statement
// DVC 20    // LFD PRNTR                      Print file
// DVC 50    // VOL VOL742                     }
// LBL INFILE    // LFD INPUT1                 } Input file
// DVC 51    // VOL VOL692                     } Output file
// LBL OUTFILE   // LFD OUTPUT1,,INIT          }
// DVC 50    // VOL VOL742                     }
// EXT ST,,,CYL,50                            } Scratch file - temporary
// LBL $SCR1    // LFD DM01                   } work space
// EXEC MILOAD
/$
    UDD    A=(255,256),B=(255,256)             }
           OM=(I,1)                            }
           MK1=(8,4)                           } Utility input and output statement
           MK2=(6,13)                          }
/*
/&
// FIN
```

- Example 2: Disk Input to Disk Output (Single Drive)

In this example, MILOAD creates a 2-key file with no RBC. Duplicate keys and key changes are allowed. The MIRAM file produced contains variable-size records, and the slot size for these records is 100 bytes.

```
// JOB MILOAD2,,18000                          Job statement
// DVC 20          // LFD PRNTR                Print file
// DVC 50          // VOL VOL742   }
// LBL INFILE      // LFD INPUT1   }            Input file
// DVC 50          // VOL VOL742   }
// LBL OUTFILE   // LFD OUTPUT1,,INIT }        Output file
// DVC 50        // VOL VOL742      }
// EXT ST,,,CYL,35                  }
// LBL $SCR1     // LFD DM01        }           Scratch file – temporary
// EXEC MILOAD                      }           work space
/$
    UDD     A=(100,512              }
            B=(100,512)             }
            OM=(I,1,V,R)            }
            FV=(100)                }           Utility input and output
            MK1=(8,4,DUP,CHG)       }           statement
            MK2=(16,8,DUP,CHG)      }
/*
/&
// FIN
```

- Example 3: Tape Input to Disk Output

  In this example, MILOAD creates a 5-key MIRAM output file with a record size of 50.

  No RCB, fixed format, duplicate keys, and key changes are allowed. The tape input is standard labelled.

```
// JOB MILOAD3,,18000                                  Job statement
// DVC 20          // LFD PRNTR                        Print file
// DVC 90          // VOL TAP001          }
// LBL TAPFILE     // LFD INPUT1          }            Input tape file
// DVC 50          // VOL VOL742          }
// LBL OUTFILE     // LFD OUTPUT1,,INIT   }            Output disk file
// DVC 50          // VOL VOL742          }
// EXT ST,,,CYL,70                        }            Scratch files - temporary
// LBL $SCR1       // LFD DM01            }            work space
/$
    UTD     A=(50,50)                      }
            B=(50,512)                     }
            OM=(I,1,,R)                    }         Utility input and output
            L0                             }         statement
            MK1=(6,14,DUP)                 }
            MK2=(10,4,DUP)                 }
            MK3=(17,6,DUP)                 }
            MK4=(4,12,DUP)                 }
            MK5=(5,27,DUP)                 }
/*
/&
// FIN
```

7.3. MESSAGES – INTERFACE FOR USERS AND OPERATORS

MILOAD provides both informational and error messages to assist you in using and understanding it. All messages are displayed on your console screen and recorded on your printer listings.

7.3.1. Informational Messages

Informational messages inform you of the current status of MILOAD during execution. They indicate when an important process begins. For example, MILOAD uses the SORT routine to sort the scratch file containing all the file keys. When this process begins, a SORT called informational message is displayed. An informational message is also displayed indicating that MILOAD has completed the sorting and is beginning construction of an indexed key structure. No action is required after these messages. They are strictly for your information, so you know the status of MILOAD and when it completes.

7.3.2. Error Messages

If your job does not run successfully and is terminated by MILOAD, one or more error messages are generated and displayed explaining why the run was unsuccessful. The messages are self-explanatory and can be used to remedy the error condition. The error messages are listed directly below the run statements on the output listing generated by MILOAD and are formatted as follows:

```
DUFnn -{W}- message text
       {S}
```

where:

nn
: Is the message number.

W
: Indicates that the message is a user warning.

S
: Indicates that the error is serious and that the UPSI byte is set to X'40'.

For a complete listing of all MILOAD error messages, refer to the system messages programmer/operator reference, UP-8076 (current version).

7.3.3. Unrecoverable Error Conditions

In addition to the error messages, there are five possible CANCEL exit paths that signify unrecoverable errors. They are identified as follows:

Open error	Results in a DUF45 error message and CANCEL termination
Error writing index	Results in a DUF28 error message and CANCEL termination
Error reading INPUT1	Results in a DUF29 error message and CANCEL termination
Error writing OUTPUT1	Results in a DUF30 error message and CANCEL termination
Error in sort	Results in a DUF45 error message and CANCEL termination

7.4. ADDITIONAL FEATURES OF THE MILOAD UTILITY

In addition to loading large, multikeyed MIRAM files, MILOAD can bulk extend (add large numbers of records to) an existing MIRAM file. The same restrictions and constraints previously noted for file creation apply to this added capability of MILOAD. Performance varies for bulk extending and is dependent upon the relationship between the key values of the records being added to those of the records already existing in the file.

Before attempting to bulk extend an existing file, we recommend that you do two things. First, make certain that the records being added to the existing file are not in conflict with that file. That is, make sure that you have not defined file specifications to MILOAD that differ from those that exist for the existing MIRAM file. An example of these differences could be format differences, RCSZ differences, key differences, etc. Second, make a backup copy of the existing file. This will allow you to restore the original contents of the file, should MILOAD terminate your job due to an error condition. Use the DUMP/RESTORE facility to back up your existing file. Output file shareability during bulk extending is not permitted. MILOAD requires the file exclusively until it has successfully completed.

PART 5. APPENDIXES

Appendix A. Statement Conventions

This appendix contains descriptions of the conventions that you must follow to properly code the statements to express the DATA routine operations you desire. Coding conventions for subparameters, optional and required keyword parameters, and punctuation marks are given. The use of capital and lowercase letters in the DATA routine statements is explained.

The conventions used to present the DATA routine statements are as follows:

- Information that must be coded exactly as shown is presented in uppercase letters. This information denotes control statement mnemonics or keyword parameters of a DATA routine utility statement.

  Examples:

```
UCD A=(80,80),DC,I1,K1,OS
SEL D=1
COR N=(1),A=(5),B=(6)
```

- Information that must be supplied by the user is presented in lowercase letters.

  Examples:

```
Q=(c,s,n,i)
X=(r,s)
```

- Field select (FS) statement parameters are positional and must be coded in the order shown in the description of the FS statement in 3.3.1.

  Examples:

```
FS      a,b,c/CV/a,b,c
FS      CV/a,b,c/a,b,c
FS      a,b,c/a,b,c/CV
```

- Information contained within braces represents alternate choices, only one of which may be chosen.

  Examples:

  ```
  IN=({vol-ser-no})
     ({RES       })
     ({RUN       })
  {OS }
  {OSY}
  ```

- Information contained within brackets represents optional entries that may be included or omitted. Braces within brackets signify that one of the specified entries must be chosen if that parameter is specified.

  Examples:

  ```
  [,{DC}]
  [,{DP}]
  [,{FF}]
  [,{FV}]
  ```

- Commas, virgules (/), and parentheses must be coded exactly as shown.

  Examples:

  ```
  UCC A=(r,b),B=(r,b)
  FS a,b,c/CV/a,b,c
  ```

- An ellipsis (a series of three periods) indicates the omission of a variable number of entries.

  Example:

  ```
  A<(sssss,aa...a)
  ```

- A keyword parameter may contain a sublist of parameters called subparameters, which are separated by commas and are positional within parentheses.

  Examples:

  ```
  Q=(c,s,n,i)
  FS a,b,c/a,b,c/a,b,c
  ```

- Commas are required when positional subparameters are omitted, except after the last subparameter specified.

  Examples:

  ```
  Q=(c,s,,i)
  Q=(c,s)
  ```

- Subparameters contained within brackets are optional. If included, they must be coded in the order shown. If omitted, a comma is required in their place.

  Example:

  ```
  OR=(I[,p][,S],[,V])
  ```

  can be coded as

  ```
  OR=(I,,V) or OR=(I,2)
  ```

- When a default specification occurs in the format description, it is printed on a shaded background. If, by parameter omission, the operating system performs processing other than parameter insertion, it is explained under an "if omitted" heading in the parameter description.

  Examples:

  {FF / FV} {K1 / K2}

- Entries made by the user are indicated in this document in reverse print (00180)

- Parameters may be coded through column 71 and must be followed by at least one blank column.

- The statement identifier (Uio, FS, SEL, DEL, COR, and PAR) may be coded in any columns between 1 and 71. It must be followed by at least one blank space and, unless starting in column 1, preceded by at least one blank space. The Hn statement must start in column 2.

- With the exception of the Hn statement, statements cannot be continued. Additional parameters are coded by repeating the statement identifier. Parameters or fields must not be split between two cards.

  Examples:

  ```
  UDD B=80,132,DP,G=10,H=5000
  UDD MY,OI,Q,R=1000,S2,TD
  PAR P2P3,P1P2,P3P1,P1=(72,720,FIXBLK,50,20)
  PAR P2=(100,100,FIXUNB,30,100),P3=(9,80,KEYED,20,50)
  ```

  Hn statements are continued by coding a nonblank character in column 72 and resuming the character string in column 2 of a second card. (See 3.3.3 for coding rules for the Hn statement.)

Index

SPERRY UNIVAC

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

_____

(Document Title)

_____ _____ _____

(Document No.) *(Revision No.)* *(Update No.)*

Comments:

Cut along line.

From:

_____

(Name of User)

_____

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424

CUT

FOLD

SPERRY UNIVAC

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

_____

(Document Title)

_____ _____ _____

(Document No.) *(Revision No.)* *(Update No.)*

Comments:

Cut along line.

From:

_____

(Name of User)

_____

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424

CUT

FOLD

SPERRY UNIVAC

USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

Please note: This form is not intended to be used as an order blank.

_____

(Document Title)

_____ _____ _____

(Document No.) *(Revision No.)* *(Update No.)*

Comments:

Cut along line.

From:

_____

(Name of User)

_____

(Business Address)

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19424

CUT

FOLD