LOAD MODE - WITH WORD MARKS - REF. PAGE 18, 3 W, & P43
MOVE " - WITHOUT WORD MARKS REF P43

1410 DATA PROCESSING SYSTEM BULLETIN

IBM 1410 INPUT/OUTPUT CONTROL SYSTEM FOR CARD AND TAPE SYSTEMS PRELIMINARY SPECIFICATIONS

This bulletin is a revision of the previous edition, Form J28-1432-1. It includes additional material and several minor corrections.

To assist the reader in identifying changes from the previous edition, pages have been marked as follows:

A dot (•) in the upper outside corner of a page indicates that the material appearing on this page is new or contains changes.

A dot (•) opposite the heading of a section indicates that the entire section has been added or changed; individual paragraphs are not marked.

A dot (•) opposite the first line of a paragraph indicates that the paragraph has been added or changed.

INTRODUCTION

The purpose of this publication is to enable programmers to avail themselves of the many advantages offered by the 1410 Input/Output Control System. To this end, the bulletin explains the functions and use of the IOCS macro-instructions and the necessary DIOCS, DTF and DA entries.

Related Publications

It is assumed that the reader is thoroughly familiar with the following IBM publications:

IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1,

IBM 1410 Data Processing System Bulletin, "Basic Autocoder for the IBM 1410: Preliminary Specifications," Form J24-1413-2,

IBM 1410 Data Processing System Bulletin, "Autocoder: Preliminary Specifications," Form J24-1433-1.

Machine Requirements

Programs incorporating the IBM 1410 Input/Output Control System can be generated by the 1410 Tape Autocoder processor which requires the following minimum machine configuration:

20,000 positions of core storage
4 IBM 729 II, 729 IV or 7330 Magnetic Tape Units (may be intermixed)
1 IBM 1402 Card Read-Punch, Model 2
1 IBM 1403 Printer with 132-character chain cartridge.

TABLE OF CONTENTS

BASIC PRINCIPLES OF THE 1410 IOCS

Advantages of 1410 IOCS

The IBM 1410 Input/Output Control System consists of a set of tested routines which free the user from all coding of input and output routines for unit record equipment (i.e., the 1402 Card Read-Punch and the IBM 1403 Printer) and magnetic tape.*

Moreover, the Input/Output Control System enables the programmer to handle logical records merely by using GET, PUT and related IOCS macro-instructions. The programmer using the IOCS is no longer concerned with the actual blocking or deblocking of records because this is handled automatically by the Input/Output Control System. The programmer can also instruct the IOCS to provide the coding required for the overlapping of input and output operations with processing if the 1410 is equipped with the Overlap and Priority special features.

Since input and output routines make up nearly half of the average program, the 1410 Input/Output Control System offers users substantial savings in program writing and testing expenses.

Available I/O Routines

The IBM 1410 IOCS provides the programmer with tested routines which will automatically:

- Read or write records or groups of records.
- Block or deblock records.
- Provide for the overlapping of read, write and computing operations (if the 1410 system has the Overlap and Priority special features).
- Check for read and write errors.
- Correct correctable errors.
- Check or write tape labels.
- Write checkpoint records.
- Check end-of-reel conditions.

How to Use IOCS

For each program which is to utilize the IOCS, the programmer must:

1. Use the IOCS macro-instructions in his program.

*A special section of the Input/Output Control System for IBM 1405 Disk Storage can be incorporated into the IBM 1410 IOCS. See IBM bulletin, "Input/Output Control System for IBM 1410 Systems with 1405 Disk Storage – Preliminary Specifications," Form J28-0233.

2. Write one set of DIOCS (Define IOCS) statements.

3. Write one set of DTF (Define The File) statements for each file* used by his program, and

4. Write proper DA (Define Area) statements for each area used by the IOCS.

The DIOCS and DTF entries are punched into IBM cards and must precede the source program during assembly. (See Figure 1)

Assembly of Programs Using IOCS

Before assembling a program using the IOCS, the Autocoder processor analyzes the DIOCS entries to determine whether the Overlap and Priority special features will be used by the object program. If so, the processor creates a "Channel Scheduler" for each channel specified. The channel scheduler(s) will later be used by the program to handle file priorities. If needed, the Channel Schedulers will be written on the intermediate output tape in the form of "one-for-one" symbolic statements. (See Figure 1)

Next, the processor, still using the DIOCS entries, determines which of the IOCS routines will be needed by the object program. The required routines are then also written on the intermediate output tape.

The DTF Entries

Next, the processor uses the DTF entries to produce a "File Scheduler" for each file used by the program. These file schedulers are later used by the object program to arrange for the proper handling of each file.

Linkages

The processor then examines the programmer's source program statements. Each time it encounters an IOCS macro-instruction, the processor generates a routine and/or a linkage to the appropriate IOCS routine (already written on the intermediate output tape) and writes this routine and/or linkage on the output tape.

Assembly of Program

In this manner, the Autocoder processor creates a complete symbolic program. This consists of sections written by the programmer and sections made up of IOCS routines supplied by IBM Applied Programming. The different sections are joined by linkages created by the processor. This symbolic program is converted into the object program, i.e., the machine-language program, by the Autocoder processor (See Figure 1).

*A file is a collection of records which may be arranged in different ways.

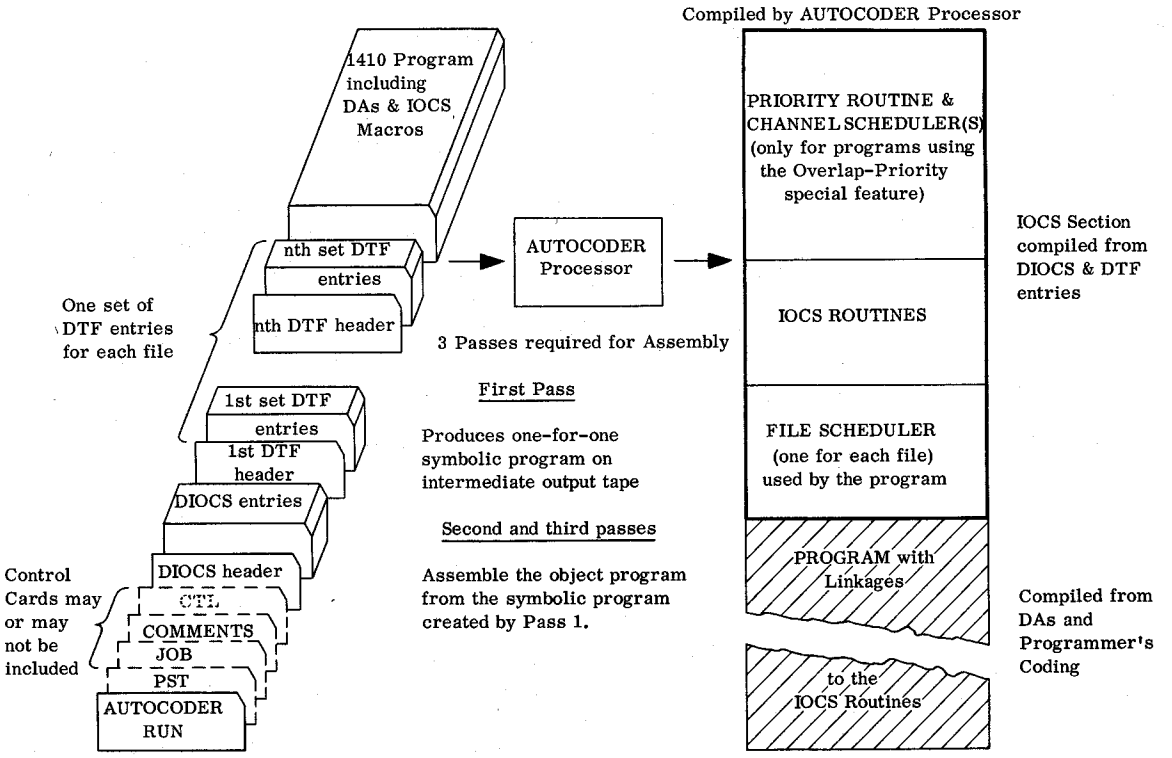


Figure 1. Assembly of a Program Using IOCS

Channel and File Schedulers

The Channel Scheduler(s) (one for each channel used by programs utilizing the Priority Special Feature) and the File Schedulers (one for each file used by the program) are compiled by the Autocoder Processor on the basis of the information supplied in the DIOCS and DTF entries.

The Channel Scheduler

The Channel Scheduler checks all pending requests for input and output operations and assigns the channel to the file of highest priority requesting the channel.

The File Scheduler

Each time the program issues a read or write request, the File Scheduler determines which input or output area of the specified file is to be used. The File Scheduler then passes the request on to the Channel Scheduler.

Remainder of Publication

The remainder of this publication consists primarily of a detailed explanation of the four programming steps required to utilize the 1410 IOCS, namely, the writing of:

1. IOCS macro-instructions
2. DIOCS entries
3. DTF entries
4. DA entries

The final section of the bulletin contains additional information for programmers. The following subjects are covered: error treatment, checkpoint and restart procedure, record additions and deletions, the size of IOCS routines, times required by the GET and PUT macro-instructions, and times required by the different phases of the Priority Routine.

HOW THE 1410 IOCS IS USED

The Twenty-one IOCS Macro-Instructions

The twenty-one IOCS macro-instructions are:

OPEN	(Open File(s))
CLOSE	(Close File(s)))
CLOSD	(Close Dump Tape)
GET	(Get Logical Record)
PUT	(Put Logical Record)
STACK	(Select Stacker and Feed)
SKIP	(Skip Carriage)
CONSL	(Console Operation)
RTLBL	(Read Tape Label)
WTLBL	(Write Tape Label)
RELSE	(Release Block)
FEORL	(Force End of Reel)
CHKPT	(Write Checkpoint Record)
RDLIN	(Read Label Information)
IOBSP	(I/O Backspace)
IORWD	(I/O Rewind)
IORWU	(I/O Rewind and Unload)
IOWTM	(I/O Write Tape Mark)
PSTAC	(Select Punch Stacker)
RTAPE	(Read Tape)
WTAPE	(Write Tape)

Each macro is described in detail below.

OPEN

If he does not use the IOCS, the programmer must determine whether the files to be used are properly mounted on the tape units specified. He must also provide for label processing, i.e., he must provide routines to read and check the labels of input files, to check the retention codes, and to write new header labels for output files.

By using the OPEN macro-instruction, the programmer lets the IOCS provide the routines necessary to handle all of these initializing functions.

The OPEN macro-instruction is written as indicated in Figure 2.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	OPEN	FILE1
0 2			
0 3	ANYLABEL	OPEN	FILE1,FILE2,FILE3
0 4			

Figure 2.

The operand of the OPEN macro-instruction consists of the name or names of the file(s) to be activated. The names must be those used to describe the files in the DTF entries. If more than one file is listed, the names must be separated by commas. (for DTF header line ; 29) 16 DTF-- 21 FILE1

What This Macro Will Do

For each file named in the operand of an OPEN macro-instruction, the IOCS — on the basis of the information contained in the DTF entries — will:

1. Determine whether a reel of tape is available on the tape unit specified.

2. Rewind the tape, if this is specified in the DTF "REWIND" entry.

3. Process tape labels, if the file has labels. (For input files, the IOCS will read and check the header label. For output files, the IOCS will check the retention code and write a new header label.)

   For multi-reel files, the services listed under (1), (2), and (3) will be performed automatically for each subsequent reel. Checks are made at the end of each reel, before records from the next reel are used.

4. Modify the file's channel, drive and priority assignments within the IOCS if such changes were made in the program after assembly. *

NOTE 1: All tape files using two input/output areas must be opened with the same OPEN macro-instruction.

NOTE 2: If the DIOCS "PRIORITY" entry is omitted, all tape files using two input/output areas must be opened by the first OPEN macro-instruction encountered. They may be closed and reopened later in the program using CLOSE and OPEN macro-instructions.

CLOSE

The programmer may use the CLOSE macro-instruction to develop all the coding required to close input and output files.

What This Macro Will Do

For each input file named in the operand of a CLOSE macro-instruction, the IOCS will rewind the tape, if this is specified in the DTF entry.

---

*See the section on "Post-Assembly Modification of Channel, Drive and Priority Assignment."

For each output file named in the operand of a CLOSE macro-instruction, the IOCS will:

1. Determine whether the output area contains partially filled blocks.

2. Write out any partially filled blocks remaining in the output area. (Partially filled blocks will be padded as specified in the DTF "PADDING" entry. If the PADDING entry was omitted, padding will be done with blanks.)

3. Write a Tape Mark, followed (for standard labels) by the trailer label and another Tape Mark.

4. Rewind the tape, if this is specified in the DTF "REWIND" entry.

NOTE: When a card output file is closed, a blank card will be punched. This causes the last card to be selected into the pocket chosen by the programmer.

The CLOSE macro-instruction is written as indicated in Figure 3.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	CLOSE	LABEL1
0 2	ANYLABEL	CLOSE	LABEL1,LABEL2,LABEL3
0 3			

Figure 3.

The operand contains the name or names of the file(s) to be closed. The names must be those used to describe the files in the DTF entries. If more than one file is to be closed, the names must be separated by commas.

CLOSD

The programmer may use the CLOSD macro-instruction to develop all the coding required to close the dump tape specified in the DIOCS "READERROR" entry.

What This Macro Will Do

This macro will cause the IOCS to:

1. Write a Tape Mark (indicating the end of the dump file).

2. Rewind the dump tape, if REWIND is listed as the operand of the macro-instruction.

3. Rewind and unload the dump tape, if UNLOAD is listed as the operand of the macro-instruction.

The CLOSD macro-instruction is written as indicated in Figure 4.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	CLOSD	REWIND
0 2			
0 3	ANYLABEL	CLOSD	UNLOAD
0 4			
0 5	ANYLABEL	CLOSD	
0 6			

Figure 4.

The operand in the first line of Figure 4 specifies that the tape is to be rewound.

The operand in the second line of Figure 4 specifies that the tape is to be rewound and unloaded.

The absence of an operand in the third line of Figure 4 indicates that the tape is not to be rewound.

GET

The programmer may use the GET macro-instructions to develop all coding required to make the next logical record available for processing. This macro will also cause a record count and a hash total to be accumulated and checked against the trailer label if this has been specified by the DTF "TOTALS" entry.

records/block Total records

The two formats of the GET macro-instruction are described below.

Format A

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	GET	INFILE1
0 2			

Figure 5.

The operand contains the name of the file from which records are to be obtained. The name must be that used to describe the file in the DTF header line.

What This Format of The Macro Will Do

The function of the GET macro-instruction depends on record type and the number of input/output areas, as follows:

1. For blocked files using only one input/output area and for all files using two input/output areas*:

*Applies only to programs using the Overlap and Priority special features.

a. If indexing is used, this macro-instruction leaves the logical record in the input area and places the address of the record's high-order position into the specified index register.

b. If indexing is not used, this macro-instruction places the next logical record into the work area specified by the DTF "WORKAREA" entry.

2. For unblocked files using only one input/output area:

   This macro-instruction leaves the next logical record in the input area.

Format B

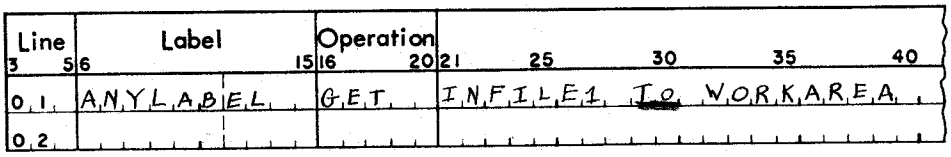


Figure 6.

The first entry in the operand is the name of the file from which records are to be obtained. This name must be that used to describe the file in the DTF header line. The second entry is the name (label) given to the work area to which the record is to be moved.

This format of the GET macro-instruction may be used for all record formats except unblocked records that use only one input area.

PUT

The programmer may use the PUT macro to develop all coding required to include a processed record in an output file.

What This Macro Will Do

Each time the programmer issues a PUT instruction the IOCS will:

1. Place a logical record in the output area.

2. Write a block of records on the output file whenever enough records have been processed to make up an output block.

3. Cause the program to branch to the end-of-reel routine whenever an end-of-reel condition is encountered in the output file.

4. Accumulate record-count and hash-total information for insertion in the trailer label if this has been specified by the DTF "TOTALS" entry.

The three formats of the PUT macro-instruction are described below.

Format A

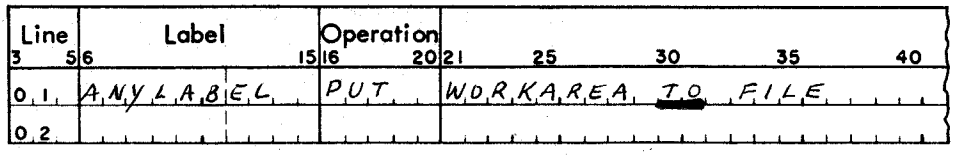


Figure 7.

The first entry in the operand is the name of the work area as defined by the DA statement. The name of the file entered in the operand must be that used to describe the file in the DTF header line.

The macro-instruction causes the logical record in the work area to be included in the specified output file.

Format B

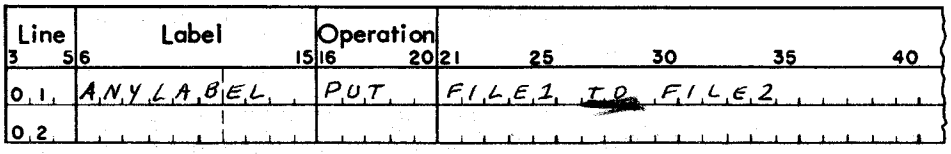


Figure 8.

The first entry in the operand is the name of the file from which the current logical record is taken. The name must be that used to describe the file in the DTF header line.

The last entry in the operand is the name of the file to which the logical record is to be moved. The name must be that used to describe the file in the DTF header line.

The function of this format of the macro-instruction depends on record type and the number of input/output areas, as follows:

1. For blocked files using only one input/output area and for all files using two input/output areas:*

   a. If indexing is used for File 1, the PUT macro-instruction will move the current logical record from the input area to the specified output file. (This corresponds to Case Ia of the GET macro-instruction.)

   b. If indexing is not used for File 1, the PUT macro-instruction will move the current logical record from the work area specified by the DTF "WORKAREA" entry to the specified output file. (This corresponds to Case Ib of the GET macro-instruction.)

*Applies only to programs using the Overlap and Priority special features.

2. For all unblocked files using only one input/output area:

   The PUT macro-instruction will move the current logical record from the input area to the specified output file.

Format C

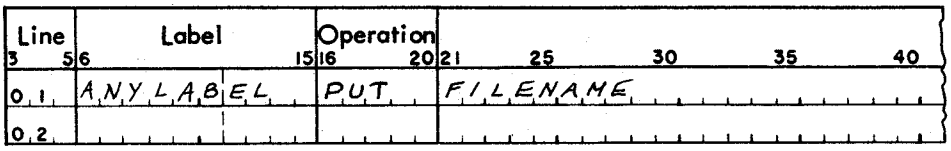


Figure 9.

This format of the PUT macro-instruction is used only if the programmer wishes to place records into the output area by actual move commands. In this case, the programmer must use the PUT FILENAME macro-instruction to enable the IOCS to account for the records so moved. The operand of this macro-instruction is the name of the output file into which records are moved.

The conditions under which the PUT FILENAME macro-instruction may be used for the different record formats are shown in Figure 10.

Record Format	PUT FILENAME Format May be Used if:	When Macro-Instruction is Given
Form-1 Records	a. One output area is used, or b. Two output areas are used, and the output file has been assigned an Indexing Register	The PUT FILENAME macro-instruction is given after the record has been moved into the output area
Form-2 Records	the output file has been assigned an Indexing Register	
Form-3 Records	same as Form-1 Records	
Form-4 Records	See DTF "VARBUILD" Entry	The PUT FILENAME macro-instruction is given before the record is moved into the output area

Figure 10. Conditions for the Use of the PUT FILENAME Macro-Instruction.

STACK

This macro-instruction corresponds to the SSF (Select Stacker and Feed) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.) The macro-instruction is used to select cards into specified pockets when the file uses the DTF "CARDPOC 9" entry.

What This Macro Will Do

This macro-instruction will generate the coding required to:

1. Stack the card that was read on the last card-read cycle into the pocket specified by the operand of the macro-instruction.

2. Check for error conditions.

The operand of this macro-instruction is:

0 if the card is to be selected into the NR pocket;

1 if the card is to be selected into Pocket 1;

2 if the card is to be selected into Pocket 2.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	STACK	1
0 2			

Figure 11.

The example in Figure 11 indicates that the card is to be selected into Pocket 1.

SKIP

This macro-instruction corresponds to the CC (Control Carriage) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.)

What This Macro Will Do

This macro-instruction will generate the coding required to:

1. Skip the tape-controlled carriage of the IBM 1403 Printer to the tape channel specified by the operand of the macro-instruction.

2. Check for error conditions.

The operand of this macro-instruction is the appropriate d-character, as indicated in Figure 12.

d	IMMEDIATE SKIP TO	d	SKIP AFTER PRINT TO	d	IMMEDIATE SPACE
1	Channel 1	A	Channel 1	J	1 Space
2	Channel 2	B	Channel 2	K	2 Spaces
3	Channel 3	C	Channel 3	L	3 Spaces
4	Channel 4	D	Channel 4		
5	Channel 5	E	Channel 5		SPACE AFTER PRINT
6	Channel 6	F	Channel 6		
7	Channel 7	G	Channel 7		
8	Channel 8	H	Channel 8	/	1 Space
9	Channel 9	I	Channel 9	S	2 Spaces
0	Channel 10	?	Channel 10	T	3 Spaces
#	Channel 11	.	Channel 11		
@	Channel 12	¤	Channel 12		

Figure 12. d-character for Control-Carriage Macro

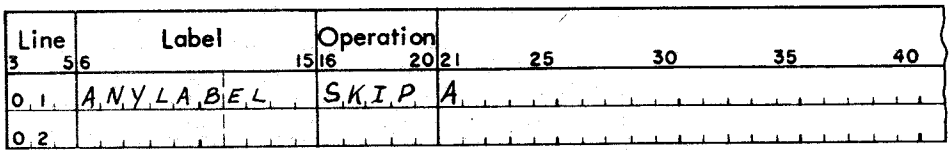


Figure 13.

The example in Figure 13 indicates that the carriage of the 1403 Printer is to skip to Channel 1 after printing.

CONSL

What This Macro Will Do

This macro-instruction will generate the coding required to:

1. Perform the console operation specified by the operands of the macro-instruction.

2. Check for BUSY and NOT READY conditions.

The operands of the CONSL macro-instruction are:

1. A letter code indicating the manner in which the console operation is to be performed, and

2. The label of the area into or from which information is to be read or written.

THE FIRST OPERAND. The first operand of this macro-instruction is:

WCP (Write Console Printer)
or
RCP (Read Console Printer),
if the operation does not use the OVERLAP special feature and takes place in the MOVE mode.

WCPO (Write Console Printer, Overlap)
or
RCPO (Read Console Printer, Overlap), if the operation uses the OVERLAP special feature and takes place in the MOVE mode.

WCPW (Write Console Printer, Word Marks)
or
RCPW (Read Console Printer, Word Marks), if the operation takes place in the LOAD mode and does not use the OVERLAP special feature.

WCPWO (Write Console Printer, Word Marks, Overlap)
or
RCPWO (Read Console Printer, Word Marks, Overlap), if the operation uses the OVERLAP special feature and takes place in the LOAD mode.

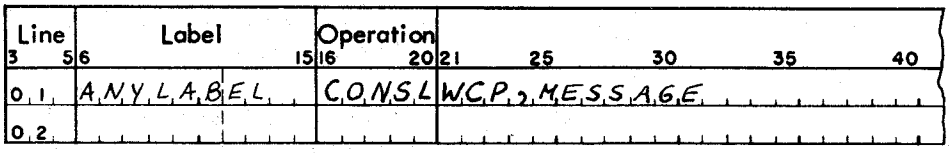


Figure 14.

The example in Figure 14 indicates that the information contained in the area labeled, MESSAGE, is to be written in the MOVE mode on the console printer without use of the OVERLAP special feature.

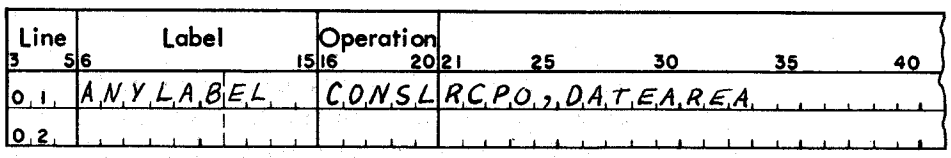


Figure 15.

The example in Figure 15 indicates that information typed out on the console printer is to be read in the MOVE mode, and with use of the OVERLAP special feature, into the area labeled DATEAREA.

RTLBL

The programmer may use the RTLBL macro-instruction to generate all the coding necessary to read non-standard tape labels.

What This Macro Will Do

This macro-instruction will generate the coding required to:

1. Read the non-standard tape label in the area specified by the third operand of the macro.

2. Check for BUSY, NOT READY and DATA CHECK conditions.

The operands of the RTLBL macro-instruction are:

1. The code indicating the manner in which the tape is to be read.

2. A two-digit number indicating the channel and drive of the tape on which a non-standard label is to be read.

3. The label of the area in storage into which the label is to be read.

The first operand is:

RT	if the label is to be read in even parity, the MOVE mode and without use of the OVERLAP special feature.

The following code letters may be added to the RT code:

B	if the label is to be read in odd parity.
W	if the label is to be read in the LOAD mode (i.e., with Word Marks).
O	if the label is to be read using the OVERLAP special feature.

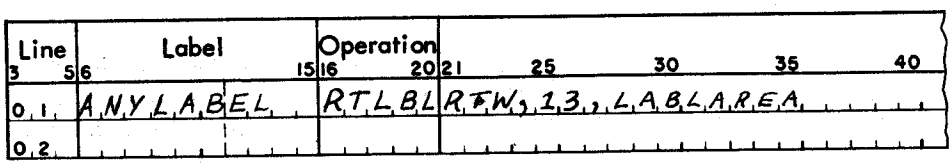


Figure 16.

The example in Figure 16 indicates that the label information is to be read in even parity and in the LOAD mode into the area labeled LABLAREA from Tape Unit 3 on Channel 1, without using the OVERLAP special feature.

NOTE: The following index registers may not be used to index the operands of the RTLBL macro-instruction: X1, X4, X5, X6, X7, X9 and X13.

WTLBL

The programmer may use the WTLBL macro-instruction to generate all the coding necessary to write non-standard tape labels.

What This Macro Will Do

This macro-instruction will generate the coding required to:

1. Write the non-standard tape label in the area specified by the third operand of the macro,

2. Check for BUSY, NOT READY and DATA CHECK conditions.

The operands of the WTLBL macro-instruction are:

1. The code indicating the manner in which the tape is to be written.

2. A two-digit number indicating the channel and drive of the tape on which a non-standard label is to be written.

3. The label of the area in storage from which the label is to be written.

The first operand is:

WT — if the label is to be written in even parity, the MOVE mode and without use of the OVERLAP special feature.

The following code letters may be added to the WT code:

B — if the label is to be written in odd parity;

W — if the label is to be written in the LOAD mode (i.e., with Word Marks);

O — if the label is to be written using the OVERLAP feature.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	WTLBL	WTB0,2,7,TRAILAREA
0 2			

Figure 17.

The example in Figure 17 indicates that the information contained in the area labeled TRAILAREA is to be written in odd parity and the MOVE mode on Tape Unit 7 of Channel 2, using the OVERLAP special feature.

NOTE 1: The RTLBL and WTLBL macro-instructions are intended primarily for the convenience of programmers using EXITS 2, 5, 6 and 7 to process non-standard tape labels.

NOTE 2: The following index registers may not be used to index the operands of the WTLBL macro-instruction: X1, X4, X5, X6, X7, X9 and X13.

RELSE

The programmer may use the RELSE macro-instruction to develop all the coding required to force the release of a partially processed input block or a partially filled output block. Thus, this macro-instruction will cause the next GET or PUT instruction to refer to the next block of the file.

What This Macro Will Do

FOR INPUT FILES. This macro will cause the first logical record of the next block to be obtained when the next GET macro is encountered.

FOR OUTPUT FILES. This macro will cause the block being built in the output area to be written onto the output file. (Fixed-length, blocked records will be padded, as specified in the DTF "PADDING" entry. If this entry was omitted for this file, partially filled blocks will be padded with blanks.)

NOTE: Record counts and hash totals cannot be taken for input files affected by the RELSE macro since records are skipped during processing. However, these counts and totals may be used for output files affected by the RELSE macro-instruction. Block counts, however, are taken automatically for both input and output files affected by RELSE macro-instructions.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	RELSE	INFILENAME
0 2			

Figure 18.

The operand of this macro-instruction is the name of the file to be released. The name must be that used to define the file in the DTF header line.

FEORL

The programmer may use this marco-instruction to develop all the coding required to force the program to branch to an end-of-reel routine. (The end-of-reel-routine options available to the programmer are listed in the description of the DTF "LABELTYPE" entry.)

NOTE: The FEORL macro-instruction can be used only if STANDARD or MIXED has been specified in the DIOCS "LABELDEF" entry.

What This Macro Will Do

FOR OUTPUT FILES. When this macro-instruction is used, the IOCS will:

1. Pad partially filled, fixed-length, blocked records (either with the character specified in the DTF "PADDING" entry or with blanks if the DTF "PADDING" entry was omitted).

2. Write out all records contained in the output area.

3. Cause the program to branch to an end-of-reel routine, with options as specified in the DTF entries.

FOR INPUT FILES. This macro-instruction will:

1. Cause the program to branch to an end-of-reel routine, with options as specified in the DTF entries.

NOTE 1: No check is made to determine whether an end-of-file condition exists.

NOTE 2: The trailer label is not processed.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	FEORL	FILENAME
0 2			

Figure 19.

The operand is the name of the file for which an end-of-reel condition is to be assumed. The name is that used to define the file in the DTF header line. Only one file may appear in the operand.

CHKPT

The programmer may use the CHKPT macro-instruction to develop all the coding required to cause the program to branch to the Checkpoint Routine.

What This Macro Will Do

This macro-instruction will cause a checkpoint record to be written on the tape unit specified by the DIOCS "CHECKPOINT" entry.

NOTE: When a DIOCS "CHECKPOINT" entry has been written, checkpoint records are automatically written each time an end-of-reel condition occurs. The CHKPT macro-instruction permits the programmer to call for additional checkpoint records to be taken at any point of his program. (See DIOCS "CHECKPOINT" entry.)

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	CHKPT	
0 2			

Figure 20.

The operand field of the CHKPT macro-instruction is left blank.

RDLIN

The programmer may use the RDLIN macro-instruction to:

a. Modify the DTF-specified information against which standard input header labels will be compared, and

b. Specify changes in the DTF-specified standard output header labels.

- The RDLIN macro-instruction should be given before the affected file is opened.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	RDLIN	FILE1, FILE2, FILE3
0 2			

Figure 21.

The operand in Figure 21 consists of the name or names of the file(s) for which label handling is to be modified. The names must be those used to define the files in the DTF header lines. If more than one file is named, their names must be separated by commas.

What This Macro Will Do

For each file named in the operand of the RDLIN macro-instruction, the IOCS will cause a card to be read into storage. The information contained in Columns 21-50 of this card will become the information against which standard header labels are to be checked, or from which standard output header labels are to be written. This label-checking information replaces that originally supplied by the DTF entries for the file.

INPUT FILES. Columns 21-50 of this card should contain the information which is to be used to check the file's serial number, reel sequence number, file name, creation date and retention cycle* contained in the header label.

The RDLIN macro-instruction thus enables the programmer to modify the information against which (standard input) header labels will be compared during the running of the program.

OUTPUT FILES. Columns 21-50 of the card should contain the information which is to be inserted into the standard output header labels to be written during the running of the program. This information should be of the same format as listed under INPUT FILES above.

The RDLIN macro-instruction enables programmers to modify the information written on standard output header labels during the running of the program.

NOTE 1: If more than one file is named in the operand of a RDLIN macro-instruction, the cards must be entered into the card reader in the order of the files to which they refer.

NOTE 2: If a RDLIN macro-instruction is used, the programmer must enter the word RDLIN as one of the operands of the DIOCS "LABELDEF" entry.

NOTE 3: The RDLIN macro-instruction may not be used prior to the first OPEN macro-instruction if the DIOCS "PRIORITY" entry is omitted.

---

*i.e., enter File Serial Number in Cols. 21-25
Reel Sequence Number in Cols. 26-30
File Name in Cols. 31-40
Creation Date in Cols. 41-45
Retention Cycle in Cols. 46-50

- IOBSP (I/O Backspace)

This macro-instruction corresponds to the BSP (Backspace Tape) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.)

What This Macro Will Do

This macro-instruction will generate the coding required to:

(1) backspace the tape unit (specified by the operand of this macro-instruction) over one tape record, and

(2) check for error conditions.

NOTE: A Tape Mark is considered a tape record.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	IOBSP	15
0 2			

Figure 22.

The operand in Figure 22 indicates that Tape Unit 5 of Channel 1 is to be backspaced over one tape record.

- IORWD (I/O Rewind)

This macro-instruction corresponds to the RWD (Rewind Tape) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.)

What This Macro Will Do

This macro-instruction will generate the coding required to:

(1) rewind the tape unit (specified by the operand of this macro-instruction), and

(2) check for error conditions.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ANYLABEL	IORWD	27
0 2			

Figure 23.

The operand in Figure 23 indicates that Tape Unit 7 of Channel 2 is to be rewound.

- IORWU (I/O Rewind and Unload)

This macro-instruction corresponds to the RWU (Rewind and Unload) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.)

What This Macro Will Do

This macro-instruction will generate the coding required to:

(1) rewind the tape unit (specified by the operand of this macro-instruction);

(2) unload the reel of tape; and

(3) check for error conditions.

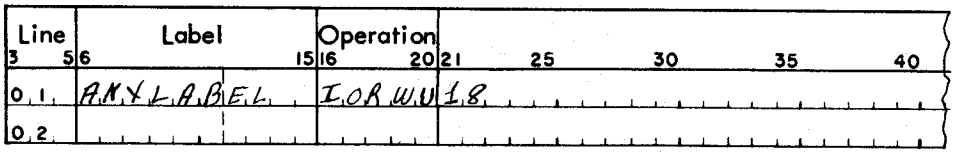


Figure 24.

The operand in Figure 24 indicates that Tape Unit 8 of Channel 1 is to be rewound and unloaded.

- IOWTM (I/O Write Tape Mark)

This macro-instruction corresponds to the WTM (Write Tape Mark) mnemonic operation code. (See IBM Reference Manual, "1410 Data Processing System," Form A24-1407-1.)

What This Macro Will Do

This macro-instruction will generate the coding required to:

(1) write a Tape Mark on the tape specified by the operand of this macro-instruction, and

(2) check for error conditions.

NOTE: The Tape Mark is written as a single-character record in even parity.

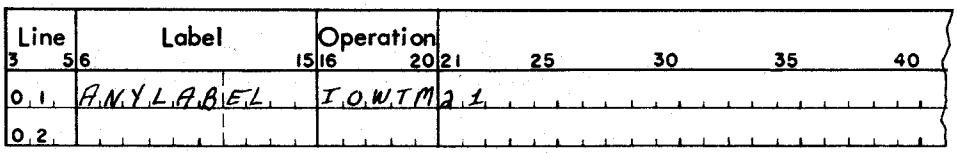


Figure 25.

The operand in Figure 25 indicates that a Tape Mark is to be written on the tape on Tape Unit 1, Channel 2.

PSTAC (Select Punch Stacker)

This macro-instruction enables the programmer to select punched cards into pockets other than those specified by the DTF "CARDPOC" entries.

What This Macro Will Do

This macro-instruction will develop the coding required to select punched cards into the pocket (0, 4 or 8) specified by the operand of the PSTAC macro-instruction instead of the pocket specified by the DTF "CARDPOC" entry.

Thus, in the coding sequence:

```
          ----
          ----
(1)       PUT
(2)       PUT
          ----
          ----
(I)       PSTAC
(3)       PUT
(4)       PUT
(II)      PSTAC
(5)       PUT
(6)       PUT
          ----
          ----
```

cards punched as the result of PUTs (1) and (2) will be selected into the pockets specified by the DTF "CARDPOC" entries of the respective files. Cards punched as the result of PUTs (3) and (4) will be selected into the pocket specified by the first PSTAC macro-instruction (I). Cards punched by PUTs (5) and (6) will be selected into the pocket specified by the second PSTAC macro-instruction (II), etc.

NOTE 1: A punch output file using the PSTAC macro-instruction must have a DTF "CARDPOC" entry.

NOTE 2: Punched cards (except error rejects) are always selected into the pocket indicated by the command that punched them.

The operand of the PSTAC macro-instruction is the pocket into which cards are to be selected. See Figure 26.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	PSTAC	8
0 2			

Figure 26.

The operand in Figure 26 indicates that all cards following this macro-instruction are to be selected into Pocket 8 of the IBM 1402 Card-Read Punch (until another PSTAC macro-instruction is encountered).

RTAPE (Read Tape)

This macro-instruction enables the programmer to read a record from any tape unit. It may be used to read a record from a tape file that uses the DTF "SCHEDULER" entry.

What This Macro Will Do

This macro-instruction will develop the coding required to:

(1) read a record from the tape unit specified in the operand of this macro-instruction; and

(2) check for error conditions.

The first operand of the RTAPE macro-instruction is:

RT — if the tape record is to be read in even parity, the MOVE mode and without use of the OVERLAP special feature.

The following code letters may be added to the RT code:

B — if the record is to be read in odd parity,

W — if the record is to be read in the LOAD mode (i.e., with Word Marks), and

O — if the record is to be read using the OVERLAP special feature.

The second operand is a two-digit number of the form xy, where

x — indicates the channel, and

y — indicates the Tape Unit on that channel from which information is to be read.

The third operand is the label of the area into which the tape record is to be read.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ANYLABEL	RTAPE	RT,B0,,2,3,,AREA1
0 2			

Figure 27.

The operand in Figure 27 indicates that a tape record is to be read from Tape Unit 3 on Channel 2 in the MOVE mode, using odd parity and the OVERLAP special feature. The record will be read into the area labeled AREA1.

NOTE 1: The IOCS will not check for wrong-length records.

NOTE 2: The IOCS will check for end-of-file conditions. If an end-of-file condition is encountered and a DTF "EOFADDR" entry for this file does not exist, an error halt will result.

NOTE 3: The following index registers may not be used to index the operands of the RTAPE macro-instruction: X1, X4, X5, X6, X7, X9 and X13.

WTAPE (Write Tape)

This macro-instruction enables the programmer to write a record onto any tape unit. It may be used to write a record on a tape file that uses the DTF "SCHEDULER" entry.

What This Macro Will Do

This macro-instruction will develop the coding required to:

(1) write a record onto the tape unit specified in the operand of this macro-instruction, and

(2) check for error conditions.

The first operand of the WTAPE macro-instruction is:

WT — if the tape record is to be written in even parity, the MOVE mode and without use of the OVERLAP special feature.

The following code letters may be added to the WT code:

B	if the record is to be written in odd parity,
W	if the record is to be written in the LOAD mode (i.e., with Word Marks), and
O	if the record is to be written using the OVERLAP special feature.

The second operand is a two-digit number of the form xy, where

x	indicates the channel, and
y	indicates the Tape Unit on that channel onto which information is to be written.

The third operand is the label of the area into which the tape record is to be written.

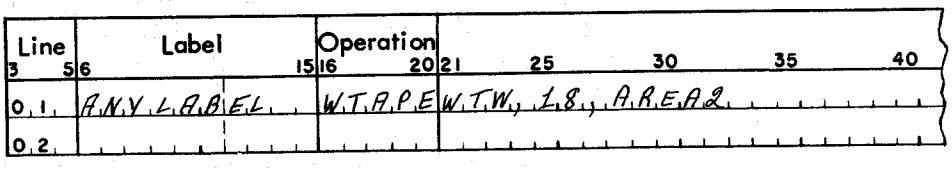


Figure 28.

The operand in Figure 28 indicates that a tape record is to be written onto Tape Unit 8 on Channel 1 in the LOAD mode, using even parity and without the OVERLAP special feature. The record will be written from the area labeled AREA2.

NOTE 1: The IOCS will check for end-of-file conditions. If there is no DTF for this file and an end-of-file condition is encountered, an error halt will result.

NOTE 2: The following index registers may not be used to index the operands of the WTAPE macro-instruction: X1, X4, X5, X6, X7, X9 and X13.

The DIOCS Entries

Before the programmer can use the Input/Output Control System, he must supply the Autocoder processor with the information needed to determine which of the IOCS routines are required by the object program. Depending on installation features and the program, this information consists of two or more card entries listed individually on the IBM 1401/1410 Autocoder coding sheet. These entries define the Input/Output Control System required by the particular object program, and are known collectively as the DIOCS (Define Input/Output Control System) entries. Each entry is described in detail below.

NOTE: The DIOCS entries merely specify the sections of the Input/Output Control System which are to be included in the object program. These sections will be included, but they need not necessarily be used. Thus, although the programmer may have specified EXIT 5 in the DIOCS "EXITS" entry, he is not obliged to use this exit.

GENERAL FORMAT. The first DIOCS entry consists of the code DIOCS in the operation field. Each subsequent DIOCS entry has a blank operation field and must have one of the labels listed below. All DIOCS-entry cards may contain comments which must be separated from the DIOCS entry by at least two adjacent blanks. The DIOCS entries following the header line may be listed in any order. The DIOCS and the IODEVICES entries are mandatory. The remaining entries are not always required.

LIST OF DIOCS ENTRIES. This section describes the purpose of each of the following DIOCS entries:

DIOCS header line
DIOCSORG
FEATURES
IODEVICES
LABELDEF
ALTDRIVE
EXITS
COUNTS
RWDOPTION
READERROR
PRIORITY
CHECKPOINT
CHANCHANGE
INQUIRY
URREQUEST

THE DIOCS HEADER LINE

The first DIOCS entry is mandatory and consists of the entry DIOCS in the operation field. It is known as the "DIOCS header line." This card must be the first card (except for special control cards*) to enter the system during Autocoder assembly.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1		DIOCS	
0 2			

Figure 29.

DIOCSORG

If this entry is not made, the Autocoder processor will begin assembly of IOCS information at core location 500.

If the programmer wants IOCS assembly to begin at another location, the actual address of this location must be listed as the operand of the DIOCSORG entry.

*Such as: AUTOCODER RUN, PST (Print Out Symbol Table), JOB, COMMENTS and CTL (Control) Cards.

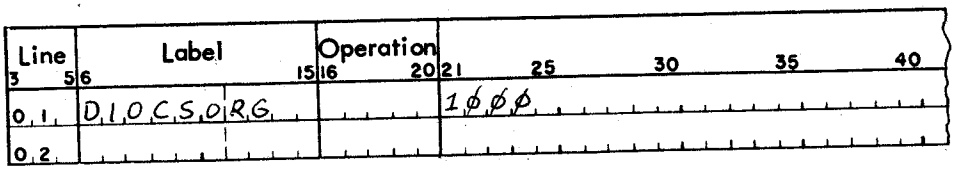


Figure 30.

The entry in Figure 30 will cause the IOCS information to be assembled beginning in location 1000.

Depending on the mode of operation and the use of certain special features, assembly of IOCS routines must begin above certain core storage locations:

1. If the Overlap and Priority special features are used, IOCS assembly must begin above Location 115.

2. If standard labels are used, IOCS assembly must begin above Location 119.

3. If the standard Autocoder loader is used, IOCS assembly must begin above Location 349.

4. If PAT (Procedure for Automatic Testing) is used, IOCS assembly must begin above Location 499.

FEATURES

This entry is not needed if the object program does not utilize Channel 2 or the Overlap and Priority special features.

The operands of the FEATURES entry are:

CHANNEL	if the program uses Channel 2;
OVERLAP	if the program uses the Overlap special feature*;
PRIORITY	if the program uses the Priority special feature.

The operands must be separated by commas and may be entered in any order.

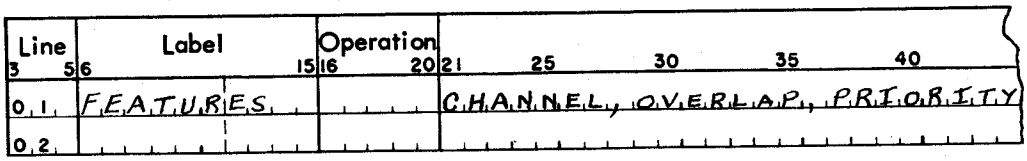


Figure 31.

The entry in Figure 31 indicates that the object program uses the Channel-2 and the Overlap and Priority special features.

*The present IOCS does not provide overlapping without the Priority special feature.

IODEVICES

This entry is mandatory. It indicates the input/output devices used by the program (except for the console printer which is required by all programs using IOCS and need not be specified).

The operands of the IODEVICES entry are:

TAPE	if the program uses magnetic tape;
READER	if the program uses the card reader;
PUNCH	if the program uses the punch;
PRINTER	if the program uses the IBM 1403 Printer;
DISK	if the program uses 1405 Disk Storage files*.

The operands must be separated by commas and may be entered in any order.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	IODEVICES		TAPE,READER,PRINTER
0 2			

Figure 32.

The entry in Figure 32 indicates that the object program uses tapes, the card reader and the printer.

The same entry could have been written as follows since operands of DIOCS entries may be entered in any order:

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	IODEVICES		PRINTER,READER,TAPE
0 2			

Figure 33.

LABELDEF

This entry is not needed if the object program does not use tape files or if the tape files used by the object program have no labels.

The operands of the LABELDEF entry are:

STANDARD	if the program uses only IBM standard tape labels;
NONSTANDARD	if the program uses non-standard tape labels and none of the files use standard labels;

*See the IBM bulletin, "Input/Output Control System for IBM 1410 Systems with 1405 Disk Storage — Preliminary Specifications," Form J28-0233.

MIXED — if some files have standard labels and some files have either non-standard or no labels, or both;

CHECK — if there are one or more files with standard labels which are to be completely checked;

This operand may not be used if the IDENT operand is used;

IDENT — if there are one or more files with standard labels for which only the identification field of the header labels is to be checked and if none of the header labels are to be completely checked. This operand may not be used if the CHECK operand is used.

NOTE: The DIOCS "CHECK" entry develops all the coding necessary for the complete checking of standard labels. Therefore, should the programmer desire to check only the identification fields of some files he can do so — even though he specified the DIOCS "CHECK" entry. The CHECK entry will provide the necessary coding as part of the coding required for the complete checking of standard labels.

TM — if one or more tape files have a Tape Mark between the header label and the first block of records;

RDLIN — if the RDLIN macro-instruction is used in the program.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABELDEF		STANDARD, IDENT
0 2			

Figure 34

The entry in Figure 34 indicates that:

a. The program uses only standard IBM tape labels.

b. The program contains one or more files which require only a check of the header label's identification field.

c. None of the header labels are to be completely checked.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABELDEF		MIXED, TM, RDLIN
0 2			

Figure 35.

The entry in Figure 35 indicates that some files of the program have standard labels and that some files have either non-standard or no labels. The entry further indicates that one or more tape files have a Tape Mark between the header label and the first block of records and that the RDLIN macro-instruction is used in the program.

ALTDRIVE

This entry is not needed if none of the tape files used by the object program are to be provided with alternate tape units.

The operand of the ALTDRIVE entry is

YES	if any of the tape files are to be provided with alternate tape units.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	ALTDRIVE		YES
0 2			

Figure 36.

The entry in Figure 36 indicates that one or more tape files used by the program are to be provided with alternate tape units.

NOTE: The assignment of an alternate tape unit to a multi-reel file can save a considerable amount of processing time.

If no alternate tape unit is assigned to a multi-reel file, processing halts each time an end-of-reel condition occurs so that the operator can mount the next reel of tape.

If an alternate tape unit is assigned, the operator mounts the second reel of the multi-reel file on the alternate unit. The IOCS provides the necessary coding to continue the input or output operation via the alternate tape unit when the first end-of-reel condition occurs, and processing continues without interruption.

Succeeding reels of the multi-reel file are then alternated between the initial and the alternate tape unit assigned to the file. The operator mounts successive reels on the tape unit to which the IOCS will switch the I/O operation at the next end-of-reel condition, and processing continues without interruption.

EXITS

This entry is not needed if no exits are used by the programmer for specialized label-handling.

The operand of the EXITS entry consists of the number(s) of the special exit(s) to be used by any of the files. The operands must be separated by commas and may be listed in any order.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	EXITS		1,5,7
0 2			

Figure 37.

The entry in Figure 37 indicates that the program uses Exits ONE, FIVE and SEVEN.

COUNTS

This entry is not needed if none of the files used by the program requires record counts or hash totals. The operand of the COUNTS entry are:

RECORD	if any files require record counts;
HASH	if any files require hash totals.

The operands must be separated by commas and may be listed in any order.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	COUNTS		RECORD, HASH
0 2			

Figure 38.

The entry in Figure 38 indicates that a record count and a hash total are required by at least one file of the program. *

RWDOPTION

This entry is not needed if:

1. The object program uses no tape files.

2. All tape files used by the program are to be rewound but not unloaded when the file is OPENed and when end-of-reel and end-of-file conditions are encountered.

The operands of the RWDOPTION entry are:

NORWD	if one or more files are not to be rewound when they are OPENed or when end-of-reel conditions are encountered. This operand may not be entered if the UNLOAD operand is listed;
UNLOAD	if one or more files are to be rewound and unloaded. This operand may not be entered if the NORWD operand is listed.

*Block counts are taken and inserted into the trailer label automatically by the IOCS.

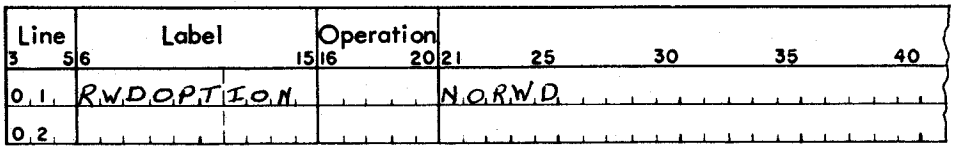


Figure 39.

The entry in Figure 39 indicates that one or more files used by the object program are not to be rewound.

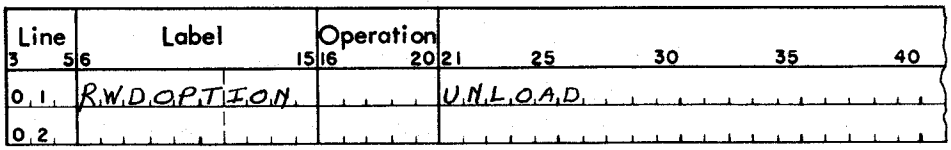


Figure 40.

The entry in Figure 40 indicates that one or more files used by the object program are to be rewound and unloaded.

NOTE: The UNLOAD operand provides the coding required to rewind and to unload tape files, but it leaves the programmer free to utilize only part of, none, or all of the coding provided. Thus, having specified UNLOAD, the programmer can freely use tape files which are to be rewound and unloaded, merely rewound, or neither.

READERROR

This entry is not needed if the object program uses no tape files or if the tape files use the standard read-error procedure.

Standard Read-Error Procedure

The standard read-error procedure is to print the unreadable block on the console printer, read in the next block of information, and continue processing.

The operands of the READERROR entry are:

SCAN — if the IOCS Error Routine is to examine the affected input block, high-order to low-order, and is to type out only the location of asterisks.

(Invalid characters entering storage during read operations are converted into asterisks if the ASTERISK-INSERT switch is set to ON. See the IBM Reference Manual, "1410 DATA PROCESSING SYSTEM," Form A24-1407-1.)

At the completion of the scan, a HALT occurs enabling the operator to correct the error from the console, to proceed, or to reject the record and place it on the error output file.

NOTE: If blocks of records are placed on the output file without processing, the trailer counts will not match for that file.

TAPE,1Y — if blocks containing unreadable information are to be written on a special tape unit specified by the 1Y operand following the TAPE operand. (1 indicates Tape Channel 1; Y indicates the tape unit.) The information will be written in the same mode (LOAD or MOVE) as read and will always be written in even parity.

NOTE: Only Channel 1 may be specified.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	R E A D E R R O R		S C A N , T A P E , 1 3
0 2			

Figure 41.

The entry in Figure 41 indicates that the location of unreadable information is to be printed by the console printer and that the block containing the unreadable information is to be written on Tape Unit 3 of Channel 1 if it cannot be corrected.

NOTE: If both SCAN and TAPE,1Y are specified, SCAN must precede the TAPE,1Y operand. The SCAN and TAPE,1Y operands must be separated by commas.

PRIORITY

This entry may be omitted:

a. if the programmer opens all of his tape files at once, and

b. if the programmer does not wish to modify tape file priorities after his first OPEN macro-instruction.

The operand of the PRIORITY entry is NONOVERLAY.

NOTE: See discussion of the Priority Routine in the section on "Size of the IOCS Routines."

CHECKPOINT

This entry is not needed if no checkpoint records are to be written.

The operand of the CHECKPOINT entry is a two-digit number. The first digit of this number indicates the channel, the second the tape unit on which checkpoint records are to be written each time an end-of-reel condition or a CHKPT macro-instruction is encountered. Checkpoint records may be written on any of the output tapes, including tapes that have been assigned alternate drives.

NOTE: If an input tape contains checkpoint records, the IOCS will recognize and ignore them.

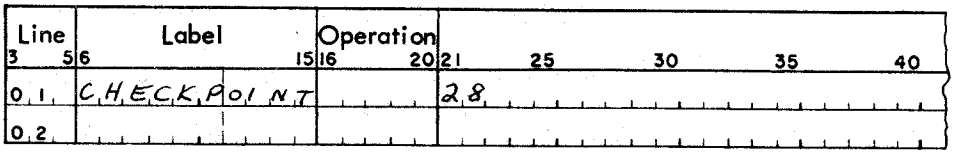


Figure 42.

This entry indicates that checkpoint records are to be taken each time an end-of-reel condition or a CHKPT macro-instruction is encountered, and that these records are to be written on Tape Unit 8 of Channel 2.

- CHANCHANGE

This entry is needed only if the programmer desires to modify a file's channel assignment after assembly. * It is not needed if the programmer desires to modify only the drive, priority, mode or parity assignment.

The operand of the CHANCHANGE entry is YES. See Figure 43.

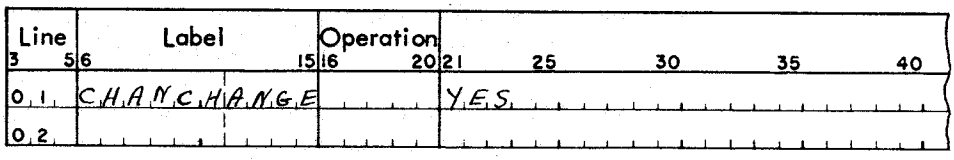


Figure 43.

NOTE 1: If the DIOCS "CHANCHANGE" entry is used with programs using the Overlap and Priority special features, approximately 700 additional storage locations are required. These additional locations and the Priority Assignment Routine occupy one contiguous block of storage locations and will be overlaid by the IOCS if the DIOCS "PRIORITY" entry is omitted.

NOTE 2: The additional storage requirement for programs that do not use the Overlap-Priority special features is negligible.

NOTE 3: The label of the block of coding forming the additional storage locations and the Priority Assignment Routine is IOCSPAHSK.

- INQUIRY

This entry is provided for interrupt programming.

The operand of the INQUIRY entry is the label of the routine used by the programmer to service an inquiry interrupt. The first instruction of this routine should be used to store the contents of the B-Register.

*See the section describing "Post-Assembly Modification of a File's Channel, Drive and Priority Assignment."

The DIOCS "INQUIRY" entry is written as indicated in Figure 44.

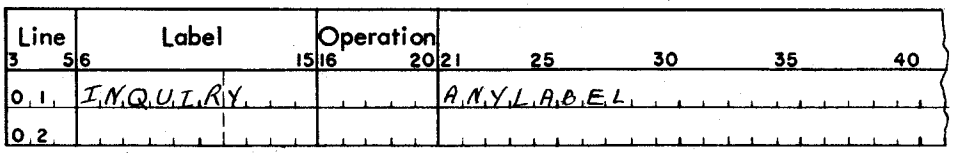


Figure 44.

The sequence of operations following an inquiry interrupt (console inquiry, teletype,* etc.) is as follows:

The IOCS will save the interrupt address and the High- Low- Equal and Zero-Balance Indicator settings, will clear Channel 1, and will branch in the non-interruptable mode to the label listed as the operand of the DIOCS "INQUIRY" entry.

After storing the contents of the B-Register, the programmer's routine will then service the interrupt, and will branch to the address specified by the preserved contents of the B-Register.

The IOCS will then restore all latches, and, after switching to the interruptable mode, will branch to the interrupted point in the program.

NOTE: The programmer may use all IOCS macro-instructions in his routine.

URREQUEST

This entry is provided for interrupt programming.

The operand of the URREQUEST entry is the label of the routine used by the programmer to service a unit record interrupt. The first instruction of this routine should be used to store the contents of the B-Register.

The DIOCS "URREQUEST" entry is written as indicated in Figure 45.

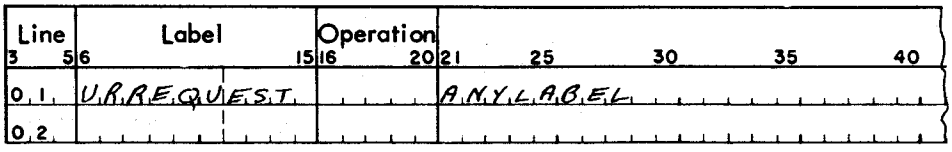


Figure 45.

The sequence of operations following the unit record interrupt is as follows:

The IOCS will save the interrupt address and the High- Low- Equal and Zero-Balance Indicator settings, will clear Channel 1, and will branch in the non-interruptable mode to the label listed as the operand of the DIOCS "URREQUEST" entry.

---

*The present IOCS does not provide a teletype error routine.

After storing the contents of the B-Register, the programmer's routine will then service the interrupt, and will branch to the address specified by the preserved contents of the B-Register.

The IOCS will then restore all latches, and, after switching to the interruptable mode, will branch to the interrupted point in the program.

NOTE 1: A unit-record interrupt takes precedence over all interrupts except inquiry interrupts.

NOTE 2: The programmer may use all IOCS macro-instructions in his routine.

The DTF Entries

In addition to the DIOCS entries, the programmer who wishes to use the Input/Output Control System must write one set of DTF (Define The File) entries for each file (magnetic tape, and unit record file) used by his program. Depending on installation features and the program, this information consists of 3 or more entries listed individually on the IBM 1401/1410 Autocoder Coding Sheet.

Each set of DTF entries describes the characteristics of the file for which it was written and indicates the methods to be used by the IOCS in handling the file. Using the information supplied in the DTF entries, the Autocoder processor develops the File Scheduler and the coding required for the proper handling of each file.

General Format

The first DTF entry consists of the code "DTF" in the operation field followed by the name of the file in the operand field. All subsequent DTF entries have blank operation fields and must have the labels listed below. All DTF entries may be followed by comments which must be separated from the DTF entries by at least two adjacent blanks. The entries following the header line may be listed in any order.

All operands of DTF entries may use address modification provided the label consists of no more than 13 characters (i.e., "LABEL +110" is a valid label if LABEL consists of no more than nine characters).

All symbolic operands of DTF entries, except those of Input/Output areas, may be indexed.

The DTF, FILETYPE and IOAREAS entries are mandatory. The remaining entries are not always required.

The set of DTF cards enters the system immediately after the DIOCS cards during Autocoder assembly. DTF cards without operands are not permitted. Each DTF entry is described below under a sub-heading indicating the label of the entry.

LIST OF DTF ENTRIES

This section describes the function and use of each of the following DTF entries:

DTF header line	TYPELABEL
FILETYPE	CHECKLABEL
MODEPAR	HEADER
CHANDRIVE	SERIALNUM
CARDPOC	REELSEQ
ALTTAPE	REWIND
RECFORM	EX1ADDR
SIZEREC	EX2ADDR
PADDING	EX3ADDR
BLOCKSIZE	EX4ADDR
IOAREAS	EX5ADDR
WORKAREA	EX6ADDR
INDEXREG	EX7ADDR
PRIORITY	EX8ADDR
EOFADDR	VARBUILD
WLRADDR	SCHEDULER
TOTALS	

The DTF Header Line

The first DTF entry is mandatory and consists of the entry DTF in the operation field, followed by the name of the file in the operand field. It is known as the "DTF header line."

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1		DTF	FILENAME
0 2			

Figure 46.

FILETYPE

The FILETYPE entry is mandatory. It specifies the type of input/output device used by the file and whether it is used for input or output. The operands of the FILETYPE entry are:

TAPE	if the file described by the DTF is a tape file;
READER	if the file described by the DTF is a card input file;
PUNCH	if the file described by the DTF is a card output file;
PRINTER	if the file described by the DTF is a printer output file;
INPUT	if the file described by the DTF is a tape input file;
OUTPUT	if the file described by the DTF is a tape output file;
WORK	if the tape file described by the DTF serves as both an input and an output file (i.e., the first time the file is OPENed, it is an output file; the second time it is OPENed, it is an input file; and it alternates with each OPEN thereafter);

CHKPT — if this file also contains checkpoint records from a previous run; this would be the case if this tape had served also as the checkpoint tape on the run during which it was created.

NOTE 1: The operands INPUT and OUTPUT are not required for unit record files.

NOTE 2: A program using the 1410 Input/Output Control System may use only one card input, one printer and one card output file.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	FILETYPE		TAPE, WORK
0 2			

Figure 47.

This entry indicates that the file described by the DTF is a tape file which is alternately used as an output and an input file (as described above).

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	FILETYPE		TAPE, OUTPUT
0 2			

Figure 48.

The entry in Figure 48 indicates that the DTF describes a tape output file.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	FILETYPE		PUNCH
0 2			

Figure 49.

The entry in Figure 49 indicates that the DTF describes a punch (and hence an output) file.

MODEPAR

This entry is not needed for unit record files or for tape files which are to be read or written in the MOVE mode and which use even parity. *

The operands of the MODEPAR entry are:

LOAD — if the tape file described by the DTF is to be read or written in the LOAD mode;

ODD — if the tape file described by the DTF is to be read or written in odd parity.

*See comments regarding the MOVE and LOAD modes and Parity Considerations.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	M,O,D,E,P,A,R,		L,O,A,D, ,O,D,D
0 2			

Figure 50.

The entry in Figure 50 indicates that the tape file described by this DTF is to be read or written in the LOAD mode and in odd parity.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	M,O,D,E,P,A,R,		L,O,A,D,
0 2			

Figure 51.

The entry in Figure 51 indicates that the file is to be read or written in the LOAD mode, using even parity.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	M,O,D,E,P,A,R,		M,O,V,E, ,O,D,D,
0 2			

Figure 52.

The entry in Figure 52 indicates that the file is to be read or written in the MOVE mode and in odd parity.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	M,O,D,E,P,A,R,		O,D,D,
0 2			

Figure 53.

The entry in Figure 53 indicates that the file is to be read or written in the MOVE mode, using odd parity.

NOTE: The operands MOVE and EVEN may be entered but are not needed.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	M,O,D,E,P,A,R,		M,O,V,E,
0 2			

Figure 54.

The entry in Figure 54 indicates that the file is to be read or written in the MOVE mode and in EVEN parity. The entire entry could have been omitted since a file is assumed to be read or written in the MOVE mode and in EVEN parity unless otherwise specified.

Load Mode vs. Move Mode

"LOAD" MODE. The handling of Word Marks and Word Separator Characters varies with the type of operation as follows:

During write or punch operations, each Word Mark is translated into a Word Separator Character which precedes the character with which the Word Mark was associated in storage, and each Word Separator Character is translated into two Word Separator Characters.

During read operations, Word Marks already in the input area are cleared. Each Word Separator Character is translated into a Word Mark, and each Pair of Word Separator Characters is translated into a Word Separator Character.

"MOVE" MODE. When information is read or written in the MOVE mode, all 64 BCD characters (including Word Separator Characters) are read or written.

Read — Word Marks in storage are undisturbed, but they are not written out as Word Separator Characters during write operations.

Write — Word Separator Characters are entered in storage and written out as Word Separator Characters.

Parity Considerations

In order to insure compatability with other IBM systems, the IBM 1410 can read and write tapes in either even or odd parity, with one exception:

On even-parity tapes, the cent-sign (¢) and the blank (b) are represented by the same BCD configuration (i.e., CA). Whenever this CA configuration is read into core storage from an even-parity tape, it will be stored as a C-bit (blank) and not as an A-bit (cent-sign).

In choosing tape parity for a given application, the programmer should consider the effect of the above on his program.

CHANDRIVE

This entry is not needed for unit record files.

The operand of the CHANDRIVE entry is:

xy — where x is the initial channel and y is the initial tape unit described by the DTF.

NOTE: No two DTFs may have the same initial channel and unit number.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	C,H,A,N,D,R,I,V,E		2,7
0 2			

Figure 55.

This entry indicates that the initial tape unit described by this DTF entry is Unit 7 of Channel 2.

NOTE: Files should be assigned to channels in such a manner that the time during which the channels are used during the running of the program is approximately the same for both channels. When reading and writing times are nearly equal, it is usually advisable to assign output files to one channel and input files to the other.

CARDPOC

This entry is needed only for card files.

The operands of the CARDPOC entry are:

x — where "x" is one digit (0, 1, 2, 4 or 8) and indicates the card pocket into which cards from this file are to be selected.

9 — if the programmer uses the STACK macro-instruction for his card file.

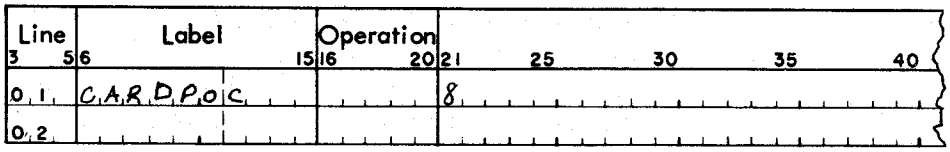


Figure 56.

This entry indicates that all cards from this file are to be selected into Pocket 8 of the card punch.

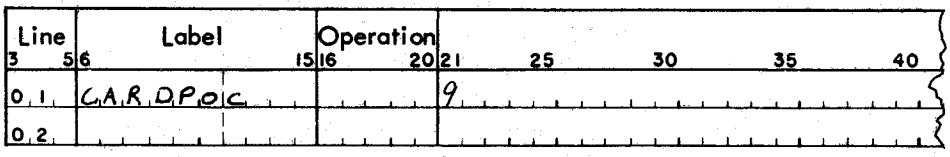


Figure 57.

This entry indicates that the program uses a separate STACK macro-instruction command for each card of this file.

ALTTAPE

This entry is not needed if no alternate tape unit is required for this file. (See section on Alternate Tape Units.)

The operand of the ALTTAPE entry is:

x — where "x" is one digit representing the number of the alternate tape unit.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	ALTTAPE		5
0 2			

Figure 58.

The entry in Figure 58 indicates that Tape Unit 5 of the tape channel specified by the CHANDRIVE entry of this DTF will be used as the alternate tape unit.

RECFORM

This entry is not needed for files containing fixed-length, unblocked records.

The operands of the RECFORM entry are:

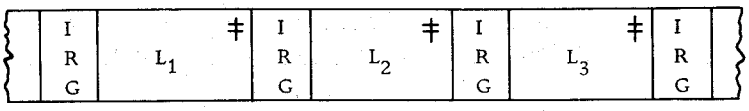
VARIABLE	if the file described by the DTF consists of variable-length records;
BLOCKED	if the records described by the DTF are blocked.

The operands must be separated by a comma and may appear in any order.

NOTE: The operands FIXED and UNBLOCKED, referring to fixed-length and unblocked records, respectively, may be used but are not required.

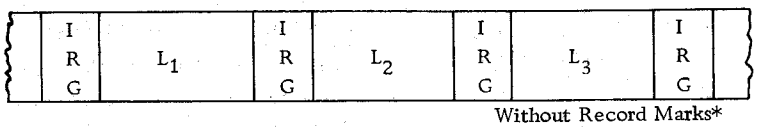
Record Formats which can be handled by the 1410 IOCS

FORM-1 RECORDS. Fixed-length, unblocked — with or without Record Marks.



With Record Marks

$L_1 = L_2 = L_3$



Without Record Marks*

Figure 59.

*Blocked records must always have a Record Mark in the low-order position.
Unblocked records may or may not contain Record Marks.
But unblocked records which are to be moved from one area of storage to another by GET or PUT macro-instructions must have either a Record Mark or a Group Mark with Word Mark immediately to the right of the low-order position.

FORM-2 RECORDS. Fixed-length, blocked - with Record Marks - with padding of short-length blocks*.

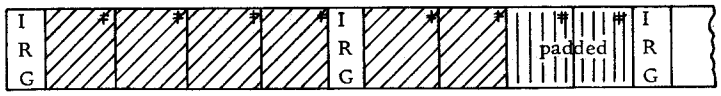


Figure 60.

NOTE: Fixed-length, blocked records which are only partially filled are padded — either with the character specified in the DTF "PADDING" entry or with blanks if the PADDING entry was omitted.

FORM-3 RECORDS. Variable-length, unblocked - with or without Record Marks.

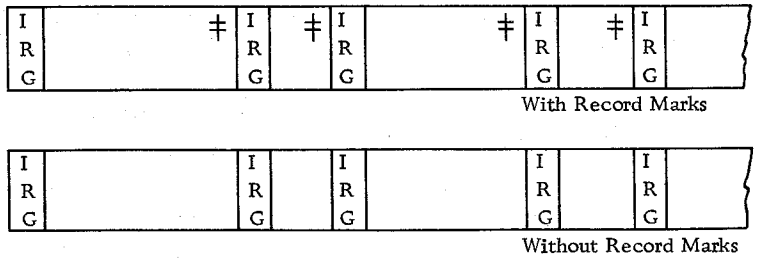


Figure 61.

FORM-4 RECORDS. Variable-length, blocked, with a Block Character-Count Field at the beginning of each block, and a Record Mark and a Record Character-Count Field in each record.

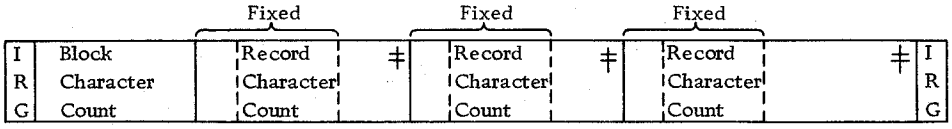


Figure 62.

Each block has a variable number of variable-length records.

Block Character-Count Field. A four-character Block Character-Count Field at the beginning of each block contains a count of the total number of characters in the block including the four-character Block Character-Count Field, itself. The Block Character-Count Field has AB zone bits over the units position. The count is used for checking and correcting wrong-length record conditions.

*Blocked records must always have a Record Mark in the low-order position.
Unblocked records may or may not contain Record Marks.
But unblocked records which are to be moved from one area of storage to another by GET or PUT macro-instructions must have either a Record Mark or a Group Mark with Word Mark immediately to the right of the low-order position.

Record Character-Count Field. A Record Character-Count Field of up to four characters in each record contains a count of the number of characters in that record, including itself and the Record Mark.

	Unblocked	Blocked	BLOCK		RECORD		May use Record Marks	May omit Record Marks	May use indexing registers	May use work Areas
			fixed-Length	Variable-Length	Fixed-Length	Variable-Length				
Form 1	X		--	--	X		Yes[1]	Yes	Needed only if two I/O areas are used	
Form 2		X	X		X		Yes	No	Yes	Yes
Form 3	X		--	--		X	Yes	Yes	Needed only if two I/O areas are used	
Form 4		X		X		X[2]	Yes	No	Yes	Yes

Figure 63. Summary of Record Formats

[1] Record Marks are required only if the output files are to be blocked.
[2] Record character-count is contained in Record Character-Count Field of each record.

NOTE: The maximum size of blocks permissible is 9999 positions.

Examples of RECFORM Entries

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	RECFORM		BLOCKED
0 2			

Figure 64.

The entry in Figure 64 indicates that this file consists of Form-2 (i.e., fixed-length, blocked) Records.

The same DTF statement could have been written:

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	RECFORM		FIXED,BLOCKED
0 2			

Figure 65.

As indicated above, records are assumed fixed-length, unblocked records, unless otherwise specified. The entries FIXED and UNBLOCKED may, therefore, be omitted.

Selecting the Tape Record Format

The time required to complete most data processing operations depends largely on the "tape time," i.e., the time required to read and write tape records.

Since starting and stopping of tape is done mechanically, it is subject to the physical laws of inertia and momentum. Therefore, a certain amount of time is needed to start and stop the tape each time a tape record is read or written. If this starting-and-stopping time is shared by 20 records of a 20-record block, the start-stop time per record is 1/20th of that required if the 20 records were placed on tape individually. An effective method of reducing the total job time, therefore, is to place tape records in groups or "blocks."

Therefore, when selecting the format of tape records, the programmer should give serious consideration to record blocking. Because record blocking and deblocking is handled by the IOCS, blocking of records does not require additional programming effort.

SIZEREC

This entry is not needed for files containing unblocked records.

For Variable-length, Blocked Records

The operand of the SIZEREC entry is:

n — where "n" indicates that the low-order position of each record's character-count field is the "nth" character of each record. (See the example below.)

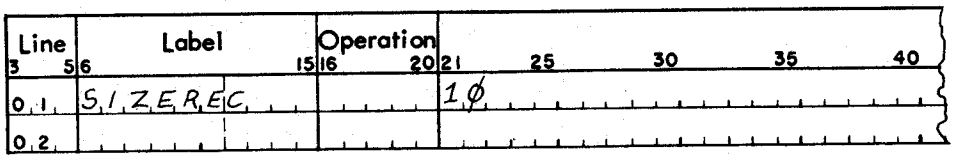


Figure 66.

The entry in Figure 66 indicates that the low-order position of the character-count field in each record of this file is the 10th position of the record. See also Figure 67.

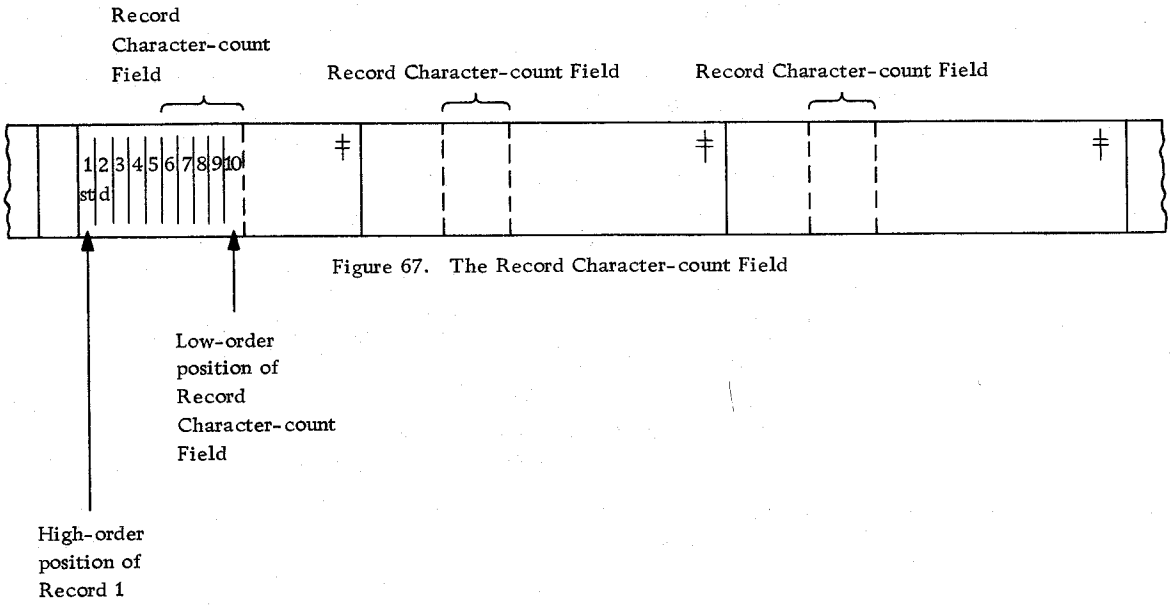


Figure 67. The Record Character-count Field

For Fixed-Length, Blocked Records

The operand is:

m	where "m" is the number of characters in the record, including the Record Mark. (Thus, the operand is "80" for eighty-character records.)

PADDING

This entry is needed only for output and work files containing fixed-length, blocked records.*

The operand of the PADDING entry is:

x	where "x" is the character with which the block is to be padded.

The following characters may not be used for padding: Asterisk, Tape Mark, Word Separator Character, Record Mark, Cent Sign and Group Mark.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	PADDING		7
0 2			

Figure 68.

The entry in Figure 68 indicates that partially filled blocks are to be padded with the digit "7".

NOTE: Record count and hash totals are included as the padding is generated.

*If the PADDING entry is omitted, partially filled blocks of this record type will be padded with blanks.

BLOCKSIZE

This entry is not needed for unblocked files, except variable-length, unblocked work files.

The operand of the BLOCKSIZE entry is:

x — where "x" is the number of positions of the input or output area of the file defined by this DTF.

NOTE: The Group Mark with Word Mark terminating the I/O area is not included in this count.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	BLOCKSIZE		2000
0 2			

Figure 69.

The entry in Figure 69 indicates that the file defined by this DTF has 2000-character input or output area(s). (If two input or output areas are used, each has 2000 characters.)

IOAREAS

This entry is mandatory. The operand of the IOAREAS entry consists of the label(s) assigned to the input (or output) area(s) of the file described by the DTF. If there are two areas*, the two labels of these areas must be separated by a comma.

NOTE: These labels may not be indexed.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	IOAREAS		LABEL1, LABEL2
0 2			

Figure 70.

The entry in Figure 70 indicates that the input (or output) areas of the file described by this DTF are those defined by DAs labeled "LABEL1" and "LABEL2", respectively.

*Overlapping of input/output operations with processing is possible only for tape files that have two input/output areas (i.e., files whose DTF "IOAREAS" entries have two operands). However, the 1410 IOCS does not permit unit record files to have two input/output areas.

Use of Two I/O Areas (only with Overlap and Priority Special Features)

Substantial savings in processing time are possible if the 1410 system is equipped with the Overlap and Priority special features. This permits the use of two input/output areas and makes possible complete overlapping of input/output operations and processing. The following considerations apply:

a. If the Overlap special feature is not used, only one input/output area can be used:

- INPUT. Record blocks are deblocked by the IOCS, i.e., logical records are successively made available for processing, one after the other, until all records in the input area have been processed. Processing then halts while the next block of records is read in, whereupon processing resumes.

- OUTPUT. In the case of output areas, the IOCS blocks the records, and each time the building of a block of records has been completed, processing halts to permit writing out of the completed block. Only after the output area has been emptied can processing — and building of the next block of records — be resumed.

b. If the Overlap special feature is used, two input/output areas can be used:

INPUT. Logical records are processed successively, one after the other. When all records in Input Area I have been processed, the IOCS makes the records in Input Area II available to the processor. Processing does not stop. While logical records are being taken successively from Input Area II, the IOCS arranges for Input Area I to be refilled with records.

When all logical records in Input Area II have been processed, the IOCS makes the new records in Input Area I available to the processor, refills Input Area II — and so forth.

In this manner, input operations and processing take place simultaneously, resulting in greater utilization of the computer system.

Figure 71.

OUTPUT. The same considerations apply to two output areas: processing and building of record blocks continue in one area while a completed block in the other area is written out.

In this manner, output operations and processing also take place simultaneously.

WORKAREA

This entry is not needed for files which do not use a work area or which use the DTF "INDEXREG" entry.

The operand of the WORKAREA entry is the label of the work area used by the input or work file.

INDEXREG

This entry is not needed if no index register has been assigned to the file described by the DTF.

The operands of the INDEXREG entry are:

X1, X2, ..., X14, indicating the index register assigned to the file.

NOTE: Index Register 15 may not be assigned to a file, although it may be used for other purposes.

PRIORITY

This entry is not needed for files of programs which do not use the Priority special feature.

The operands of the PRIORITY entry are:

0 to indicate highest priority
1 to indicate next-to-highest priority
2
.
.
.
9 to indicate lowest priority

The operands must be separated by a comma and may appear in any order.

The operand indicates the relative priority of the file described by the DTF on the channel specified by the CHANDRIVE entry of the DTF.

NOTE 1: Files of greatest activity should be assigned highest (i.e., "0") priority.

NOTE 2: Files for which the DTF "PRIORITY" entry is omitted will be assigned lowest (i.e., "9") priority.

EOFADDR

This entry is needed only for input files, including work files used as input files.

The operand of the EOFADDR entry is the label of the end-of-file routine written by the programmer.

WLRADDR

This entry is not needed for output files and for variable-length unblocked input files.

The operand of the WRLADDR entry is the label of the wrong-length-record routine written by the programmer.

NOTE: 1. If this entry is omitted, a wrong-length-record check will not be made.

2. Wrong-length-record checks will not be made for variable-length, unblocked files.

3. In his own routine, the programmer may give any I/O commands except GET macro-instructions.

Each time a wrong-length-record condition occurs, the IOCS will retry reading the record nine times. If the block remains in error, the IOCS will branch to the programmer's routine labeled WLRADDR (in which the programmer may either attempt to correct the block or dump it.)*

NOTE: The programmer should use the first instruction of his WLRADDR routine to store the contents of the B-Register. This will enable him to return control to the IOCS at the end of his WLRADDR routine. Assume the contents of the stored B-Register were X:

If the programmer branches control to X, the IOCS will process the corrected block.

If the programmer branches control to X + 7, the IOCS will bypass processing of the faulty block.

TOTALS

This entry is not needed if a record count or a hash total is not to be inserted in the trailer label.

The operands of the TOTALS entry are:

RECORD	if a record count is desired, and
x	where "x" is the address of the low-order position of the count field (from which hash-total counts are to be taken) relative to the high-order position of the record. See Figures 72 and 73.

*If a wrong-length-record condition occurs in the first area of a two-area file, the IOCS will branch control to the programmer's Wrong-Length-Record Routine. If a wrong-length-record condition occurs in the second area, the IOCS will branch control to the programmer's Wrong-Length-Record Routine PLUS FOURTEEN.

Thus, if the operand of the file's DTF "WLRADDR" entry is WLROUTINE, the IOCS will branch control to WLROUTINE for a wrong-length-record condition in Area 1, and to WLROUTINE +14 for a wrong-length-record condition in Area 2. (Area 1 is defined by the first operand of the DTF "IOAREAS" entry; Area 2 by the second operand of this entry.)

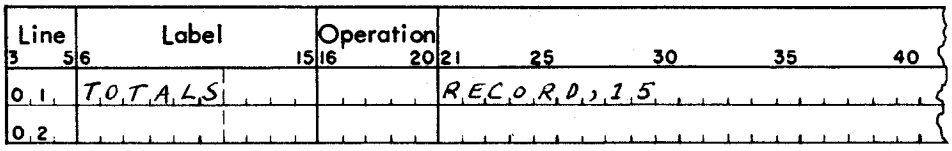


Figure 72.

The entry in Figure 72 indicates that hash-total counts are to be taken from the field shown in Figure 73.

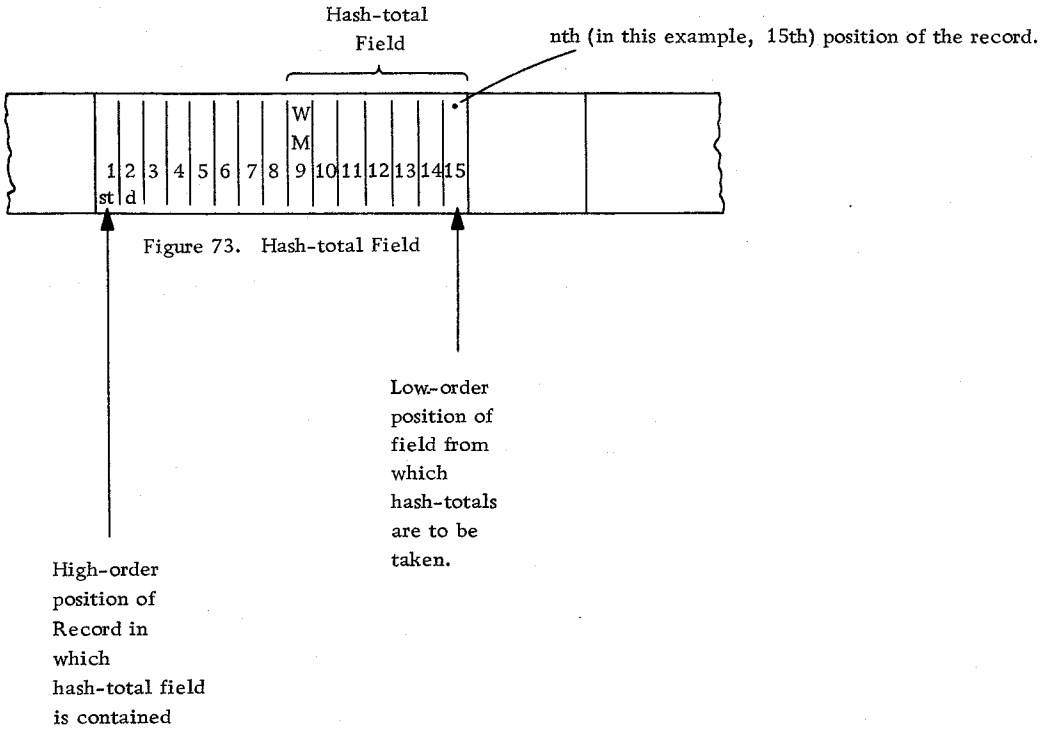


Figure 73. Hash-total Field

NOTE: Hash-total fields may contain any alphanumeric information but must not be larger than 10 characters. A Word Mark must define the high-order position of the field.

TYPELABEL

This entry is not needed if the file described by the DTF has no labels.

The operands of the TYPELABEL entry are:

STANDARD	if the file described by the DTF uses IBM standard labels;
NONSTANDARD	if the file described by the DTF uses non-standard labels;
TM	if the file has a Tape Mark between the header label and the first block of records.

If two operands are used, they must be separated by a comma. They may be entered in any order.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	TYPELABEL		NONSTANDARD,TM
0 2			

Figure 74.

CHECKLABEL

This entry is not needed if standard labels are not used or if labels are not to be checked.

The operands of the CHECKLABEL entry are:

ALL — if header and trailer labels are to be completely checked.

IDENT — if trailer labels are to be completely checked, but if only the ten-character file identification name of the header label is to be checked.

IBM 1410 Tape Labels

This section describes:

1. The Purpose of Tape Labels
2. Recommended Tape-Labeling Practices
3. Standard IBM Tape Labels
4. Use of Tape Labels by the IOCS

1. THE PURPOSE OF TAPE LABELS. The 1410 Input/Output Control System uses standard IBM header and trailer labels to insure proper handling of input and output files during the running of 1410 programs.

Header Labels. The 1410 IOCS can check input header labels to insure that the object program uses the proper files.

Before writing a new output file, the IOCS checks the output tape's retention cycle (i.e., the period of time following its creation during which the file may not be destroyed). If the current date falls within the retention cycle, the IOCS informs the operator via an appropriate message (printed on the console printer) that an attempt was made to write on a file protected by an unexpired retention cycle.

Trailer Labels. The IOCS can check input trailer labels to insure that all information in the file has been processed and to indicate end-of-reel and end-of-file conditions. The IOCS can also write trailer labels for output files to provide the checking functions just described when the output files are later used as input files. (A Tape Mark, the Trailer Label, and another Tape Mark form the last three records of each tape, in that order.)

IOCS Label-Checking and Label-Writing Functions. All tape-handling errors discovered by the IOCS are called to the attention of the operator by appropriate messages printed on the console printer. If necessary, the program is halted to enable the operator to take corrective action. The label-checking and label-writing functions offered by the IOCS are summarized in Figure 73.

2. RECOMMENDED TAPE-LABELING PRACTICES

Separation by Density. As tapes enter the system, they should be divided into two groups:

a. Tapes to be written in high density (556 characters per inch), and
b. Tapes to be written in low density (200 characters per inch).

Temporary Labeling. Next, a temporary header label followed by a Tape Mark should be placed on each tape until it is used as a data or program tape. This labeling may be done by means of the IBM 1410 Tape File Generator utility program, or by off-line equipment.

Temporary Header Labels. Temporary header labels should have the format indicated in Figure 75.

Field No.	Positions	Contents	Description
1	1-5	1HDRb	Header-Label Identifier
2	6-10	xxxxx	Tape Serial Number
3	11-80	blank	May be used as desired by the programmer

Figure 75. Temporary Header Label

Tape Serial Number. The tape serial number (Field 2) is a five-digit number (00001-99999) assigned consecutively to tapes entering the 1410 system.

High-density tapes should be numbered 00001-49999, and

Low-density tapes should be numbered 50000-99999.

High- and low-density tapes should be stored separately.

After tapes have been labeled, permanent physical labels should be placed on each reel indicating the date the tape entered the system, the serial number assigned, and the density in which the tape is to be written. Different-colored labels should be used for high- and low-density tapes so that they may be easily distinguished. The physical label should not be removed until the tape is retired from the system.

3. IBM STANDARD TAPE LABELS. Standard IBM tape labels have eighty characters. They are written in even parity and in the MOVE mode, and they are read in even parity and in the LOAD mode.

Standard Header Label. The IBM standard header label consists of eight fields containing:

1. A five-character header flag (1HDRb), Cols. 1-5
2. A five-character tape serial number, Cols 6-10
3. A five-character file serial number, Cols. 11-15
4. A five-character reel sequence number (-xxxb), Cols. 16-20
5. A ten-character file name, Cols. 21-30*
6. A five-character creation date, Cols. 31-35 (See NOTE 1)
7. A five-character retention cycle, (-000b), Cols. 36-40 (See NOTE 2).
8. A forty-position field that is blank if not filled in by the user, Cols. 41-80. This field may be used in any way desired by the programmer.

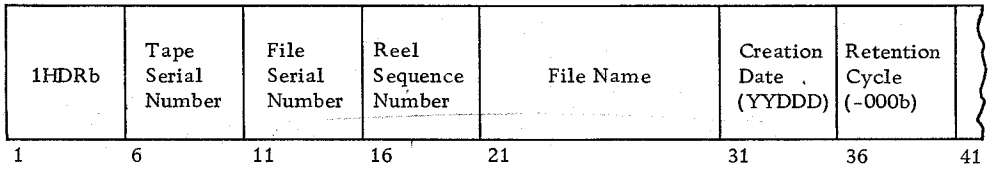


Figure 76. The Standard Header Label.

NOTE 1: The two high-order digits of the creation date indicate the year (00-99) and the remaining 3 digits the (nth) day of that year (001-366) on which the file was created.

NOTE 2: In the Retention-Cycle Field, the three-digit number following the minus-sign indicates the number of days the file is to be preserved after the creation date. Files should be preserved until all output data created from them has been used successfully as new input. This insures that any record which requires the file as input can be reconstructed by means of a single run. Standard header labels provide for retention cycles from 1 to 365 days. For files which are to be protected indefinitely, the programmer can insert the digits "99" in the two high-order positions of the creation date.

Standard Trailer Label. The standard IBM trailer label consists of the following fields:

1. A five-character trailer flag, "1EOFb" or "1EORb", depending on whether the label indicates end-of-file or end-of-reel, Cols. 1-5

2. A five-character Block Count, Cols. 6-10

3. A ten-character Record Count, (optional), Cols. 11-20

4. A ten-character Hash Total, (optional), Cols. 21-30 (or Cols. 11-20 if no Record Count is taken).

5. A fifty-position field that is blank if not filled in by the user, Cols. 31-80. (This field may be used in any way desired by the programmer.)

*The file name must consist of exactly 10 alphameric characters.

The 4 Formats of the Standard Trailer Label.

1. If both Record Count and Hash Total are specified by the DIOCS "TOTALS" Entry

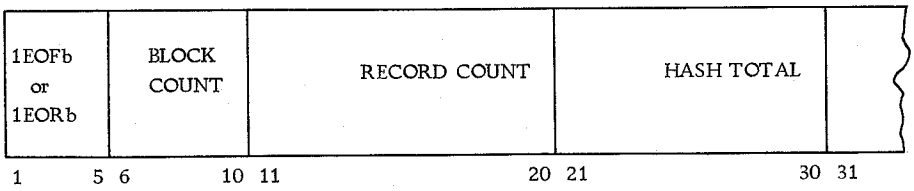


Figure 77. Format 1 of Standard Trailer Label

2. If neither Record Count nor Hash Total are specified by the DIOCS "TOTALS" Entry

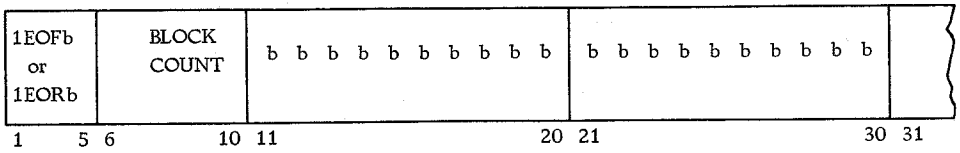


Figure 78. Format 2 of Standard Trailer Label

The third and fourth fields (Cols. 11-30) contain blanks.

3. If only a Hash Total is specified by the DIOCS "TOTALS" Entry

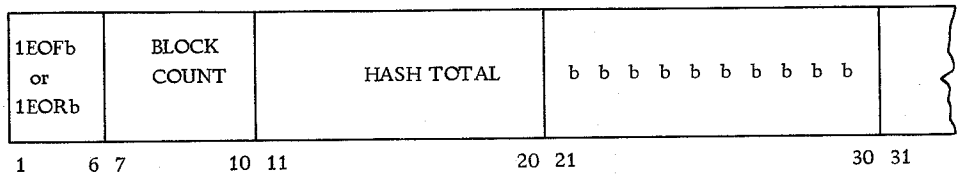


Figure 79. Format 3 of Standard Trailer Label

The third field (Cols. 11-20) contains the current HASH TOTAL, beginning with all zeros. The fourth field (Cols. 21-30) contains blanks.

4. If only a Record Count is specified by the DIOCS "TOTALS" Entry

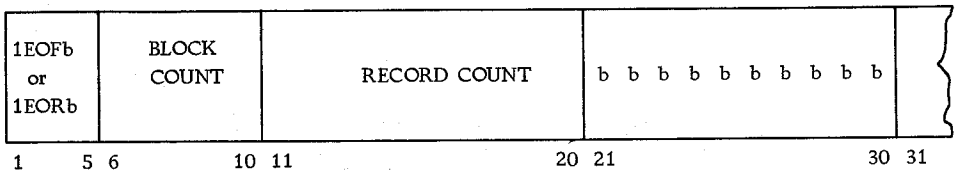
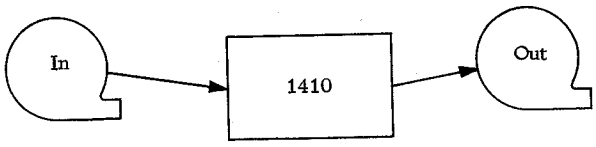


Figure 80. Format 4 of Standard Trailer Label

The third field (Cols. 11-20) contains the current RECORD COUNT, beginning with all zeros. The fourth field (Cols. 21-30) contains blanks.

4. USE OF TAPE LABELS BY THE IOCS. The label-checking and label-writing functions provided by the 1410 IOCS are summarized below.



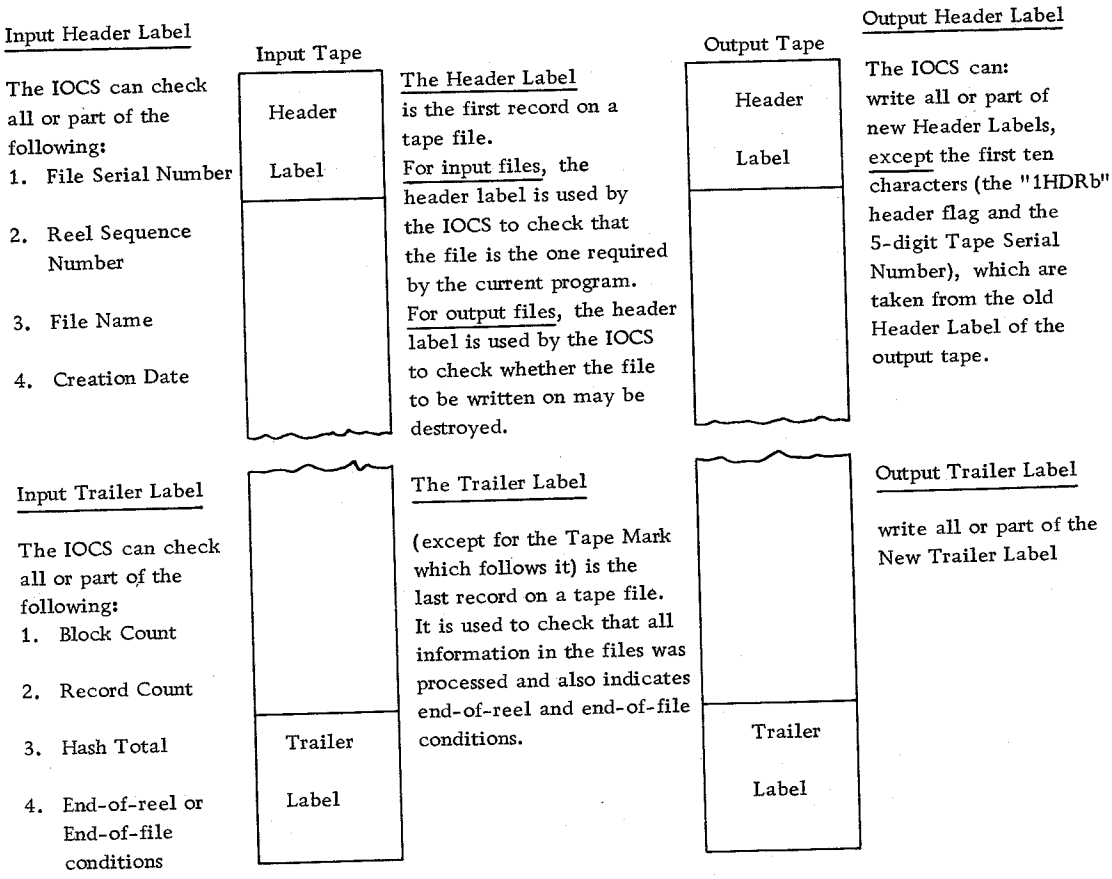


Figure 81. Use of Tape Labels by the IOCS

HEADER

This entry is not needed if standard labels are not used or if labels are not to be checked.

The operands of the HEADER entry are:

For Input Files and Work Files

1. The ten-character file identification name.
2. The five-digit creation date.

For Output Files*

1. The ten-character file identification name.

2. The three-digit retention cycle (in days).

The operands must be listed in the order indicated and must be separated by a comma.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	HEADER		FILENAME,6Ø6
0 2			

Figure 82.

The entry in Figure 82 indicates that this (input) file is named FILENAME and was created on the 6th day of 1961.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	HEADER		FILENAME1,Ø6Ø
0 2			

Figure 83.

This entry indicates that this (output) file is named FILENAME1 and has a retention cycle of 60 days.

SERIALNUM

This card is not needed if:

1. The file described by the DTF has no file serial number, or
2. The file serial number is not to be checked, or
3. The file serial number (of an output file) is equal to the tape serial number.

The operand of the SERIALNUM entry is

xxxxx — where "xxxxx" is the standard, five-digit file serial number.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	SERIALNUM		12345
0 2			

Figure 84.

*For output files, the creation date is taken from storage positions 115-119 where it should be entered each day by means of a load card.

REELSEQ

This entry is not needed if the first reel of the standard-label file described by this DTF is numbered "001".

The operand of the REELSEQ entry is

xxx	where "xxx" is the three-digit alternate reel sequence number assigned to the reel by the programmer.

NOTE: If the programmer assigns his own reel sequence number, he must modify it himself (by using an exit).

REWIND

This entry is not needed:

1. if the file described by this DTF is not a tape file, or
2. if the programmer desires to rewind the file whenever an OPEN macro-instruction or an end-of-reel condition is encountered.

The operands of the REWIND entry are:

UNLOAD	if the file is to be rewound and unloaded at the end of the reel;
NORWD	if the file is not to be rewound. (i.e., neither on encountering an OPEN macro-instruction nor at the end of the reel.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	REWIND		UNLOAD
0 2			

Figure 85.

The entry in Figure 85 indicates that the file described by this DTF is to be rewound and unloaded at the end of the reel.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	REWIND		NORWD
0 2			

Figure 86.

The entry in Figure 86 indicates that the file described by this DTF entry is not to be rewound during the running of the program.

The 8 IOCS Exits

None of the following eight EX. ADDR entries is needed if the programmer does not wish to modify the standard treatment of labels provided by the 1410 IOCS.

The eight exits are located between different sections of the IOCS label-handling routines. The exits make it possible to bypass all or part of the IOCS label-handling routines, enabling the programmer to write his own label-handling routines. (For example, the programmer can use one such exit (i.e., EX6ADDR) to perform additional checking of (standard) input trailer labels.) After completing his own specialized routine, the programmer in each but one case (EX8ADDR below) returns to the program by branching to the re-entry location, labeled "IOCSRENTRY". Standard labels are read into and written from an 80-character area labeled, "IOCSLBAREA".

The operand of each of these DTF entries is the label of the specialized label-handling routine written by the programmer. The following eight exits are provided:

EX1ADDR

This exit permits the programmer to branch to his own routine in order to enter additional information into the output trailer label.

NOTE: The programmer using Exits 1 through 7 may not use any IOCS macro-instructions in his own subroutine, except the RTLBL, WTLBL and CONSL macro-instructions.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	EX1ADDR		OWNLABEL
0 2			

Figure 87.

The entry in Figure 87 indicates that the programmer wishes to enter additional information into output trailer labels by branching to his routine, OWNLABEL. The last instruction of this routine will be a branch to IOCSRENTRY. The IOCS will then continue all necessary subsequent label handling.

EX2ADDR

This exit permits the programmer to branch to a routine written to process non-standard output trailer labels or to write labels that are in addition to the standard output trailer label.

EX3ADDR

This exit permits the programmer to branch to his own routine for the checking of standard output header labels. (This checking will not be done by the IOCS when Exit 3 is used.)

EX4ADDR

This exit permits the programmer to branch to his own routine which will modify the standard (output) header labels to be written by the IOCS.

EX5ADDR

This exit permits the programmer to branch to his own routine which will write non-standard header labels or write output labels in addition to the standard header label.

EX6ADDR

This exit permits the programmer to branch to his own routine which will perform additional checking of standard input trailer labels and/or additional input trailer labels or non-standard input trailer labels.

NOTE: This exit may be used to modify the reel-sequence count and/or the DTF "REWIND" entry.

EX7ADDR

This exit permits the programmer to branch to his own routine which will perform additional checking of standard input header labels and/or additional input header labels or non-standard input header labels.

EX8ADDR

This exit permits the programmer to ignore the end-of-reel reflective spot which precedes the first Tape Mark and enables him to write additional records.

NOTE: This is the only IOCS exit which does not permit return to the IOCS error routine via IOCSRENTRY. In order to close the reel, the programmer must give either a CLOSE or a FEORL macro-instruction.

Summary of DTF Exits

Each exit makes it possible to branch to a routine written by the programmer to modify the standard label-handling treatment provided by the 1410 IOCS. In each case except EX8ADDR, the last instruction of the modifying routine is a branch to IOCSRENTRY which returns the program to the appropriate section of the IOCS label-handling routines.

VARBUILD

The VARBUILD entry enables the programmer to build variable-length records in the output area.

The operand of the VARBUILD entry is the label of a five-position area in storage reserved by the programmer.

Before building the record, the programmer must place the length of the record into the area defined by the VARBUILD operand. He then issues the PUT FILENAME macro-instruction. The IOCS will then insert the high-order address of the area into which the record will be moved by the programmer into the area specified by the DTF "VARBUILD" entry.

NOTE 1: The label provided by the user may designate an index register.

NOTE 2: The IOCS will not accumulate hash totals, but the programmer is free to do so. (The label of the file hash-total field is IOCScuTHT, where

c = number of the channel (1 or 2), and

u = number of the tape unit (0-9).

- SCHEDULER

This special DTF entry may be used to specify a file subject to certain restrictions in exchange for a reduction in core-storage requirements.

Records contained on files using this DTF entry cannot be processed by means of GET, PUT, or RELSE macro-instructions, and they may be read or written only by means of the RTAPE and WTAPE macro-instructions. OPEN, CLOSE, FEORL and RDLIN macro-instructions may be used for such files.

These restrictions permit use of a small file scheduler and consequent savings of 250-1,000 core-storage locations for each file using the DTF "SCHEDULER" entry.

Because of the limitations imposed by the reduced file scheduler, the following DTF entries may not be used for files using the DTF "SCHEDULER" entry:

RECFORM	INDEXREG
SIZEREC	PRIORITY
PADDING	WLRADDR
BLOCKSIZE	TOTALS
IOAREAS	VARBUILD
WORKAREA	CARDPOC

The operand of the DTF "SCHEDULER" entry is NO. See Figure 88.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	S C H E D U L E R		N O
0 2			

Figure 88.

NOTE 1: The DTF "SCHEDULER" entry may be used only for tape files.

NOTE 2: Files using reduced file schedulers may be used to create or check labels on checkpoint tapes, dump tapes or error tapes.

NOTE 3: The IOCS will indicate both end-of-file and end-of-reel conditions. If an end-of-reel condition occurs, the programmer must use an exit. In his EXIT routine, he must use a FEORL macro-instruction for switching to the new reel. (Output files require Exit 8; input files require Exit 6. In neither of these cases can the user return to his program via IOCSRENTRY.)

NOTE 4: The programmer may utilize a file using the DTF "SCHEDULER" entry as a work file by modifying the contents of Location IOCSCUTFL1, where C indicates the Channel Number (1 or 2) and U indicates the Tape Unit Number (0-9).

The contents of this location should be:

1 if the tape file is an input file, and

0 if the tape file is an output file.

DA (Define Area) Entries Needed to Support the IOCS

All areas used by the Input/Output Control System (i.e., input, output and work areas) must be reserved by the programmer by means of appropriate DA entries. (A general discussion of how DAs are written may be found in the bulletin, "Basic Autocoder for the IBM 1410: Preliminary Specifications," Form J24-1413-2.) All such areas must be terminated by a Group Mark with Word Mark immediately to the right of the low-order position of the area. The label of the DA entry is used to describe the area in the DTF "IOAREAS" entry.

The remainder of this section describes how DA entries are written for the different I/O areas and different types of records. The examples cover input areas, output areas and work areas, in that order, and illustrate DAs for all types of records which can be handled by the 1410 IOCS (both with and without indexing, wherever applicable).

INPUT AREAS

DAs for the input areas required by the IOCS fall into the following four major categories, depending on record type and the number of input/output areas:

1. Unblocked Records Using Only One I/O Area.
2. Unblocked (fixed- or variable-length) Records Using Two I/O Areas.
3. Blocked, fixed-length Records.
4. Blocked, variable-length Records.

These major categories and their sub-divisions are discussed below.

1. Unblocked Records Using Only One I/O Area.

a. For data read in the MOVE mode

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABEL	DA	1X80,G
0 2	FIELD1		1,5
0 3	FIELD2		6,10
0 4	FIELD3		11,15
0 5			

Figure 89.

The LABEL of the DA is used to describe the area in the DTF "IOAREAS" entry. The "1x80" operand in Figure 89 indicates that one area of 80 locations is to be reserved. The "G" operand indicates that a Group Mark with Word Mark is to be placed one position to the right of the entire area defined by the DA.

b. For data read in the LOAD mode

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABEL	DA	1X80,G
0 2	FIELD1		5
0 3	FIELD2		10
0 4	FIELD3		15
0 5			

Figure 90.

NOTE: The labels of these DA areas must not be used as operands of macro-instructions.

2. Unblocked (fixed- or variable-length) Records Using Two I/O Areas.

a. With indexing, for data read in the MOVE mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1X80,X2,0,G
0 2	FIELD1		1,5
0 3	FIELD2		6,10
0 4	FIELD3		11,15
0 5			
0 6	AREA2	DA	1X80,G
0 7			1,5
0 8			6,10
0 9			11,15
1 0			

Figure 91.

NOTE 1: The "X2" operand in Figure 91 indicates that the labels of the fields and subfields entered under this DA entry, when referred to in symbolic instructions, are indexed by Index Register 2.

NOTE 2: The "0" operand indicates that storage assignment of fields and subfields entered under this DA header line is to be relative to zero.

b. With indexing, for data read in the LOAD mode.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	AREA1	DA	1X80,X2,0,G
0 2	FIELD1		5
0 3	FIELD2		10
0 4	FIELD3		15
0 5			
0 6	AREA2	DA	1X80,G
0 7			

Figure 92.

NOTE: The "AREA1" and "AREA2" labels of these DAs are the names used to describe these areas in the DTF "IOAREAS" entry.

c. Without indexing, in either MOVE or LOAD mode. (Processing is done in a work area.)

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	AREA1	DA	1X80,G
0 2	AREA2	DA	1X80,G
0 3			

Figure 93.

3. Blocked, fixed-length Records.

a. With indexing, for data read in the MOVE mode.

Line	Label	Operation	
3 5	6 15	16 20	21 25 30 35 40
0 1	AREA1	DA	10X80,X2,0,G
0 2	FIELD1		1,5
0 3	FIELD2		6,10
0 4	FIELD3		11,15
0 5			
0 6	AREA2	DA	10X80,G
0 7			1,5
0 8			6,10
0 9			11,15
1 0			

Figure 94.

b. With indexing, for data read in the LOAD mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	10X80,X2,0,G
0 2	FIELD1		5
0 3	FIELD2		10
0 4	FIELD3		15
0 5			
0 6	AREA2	DA	10X80,G
0 7			

Figure 95.

c. Without indexing, for data read in either the MOVE or LOAD mode. (Processing is done in a work area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	10X80,G
0 2	AREA2	DA	10X80,G
0 3			

Figure 96.

4. Blocked, variable-length Records.

a. With indexing, for data read in the LOAD mode.*

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1X2000,X2,0,6
0 2	FIELD1		5
0 3	FIELD2		10
0 4	FIELD3		15
0 5			
0 6	AREA2	DA	1X2000,G
0 7			

Figure 97.

*Indexing not available for MOVE mode.

b. Without indexing, for data read in the MOVE mode. (Processing is done in a work area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1x200,G
0 2			1,1
0 3	AREA2	DA	1x200,G
0 4			1,1
0 5			

Figure 98.

NOTE: The "1,1" entries are needed by the IOCS.

c. Without indexing, for data read in the LOAD mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1x200,G
0 2	AREA2	DA	1x200,G
0 3			

Figure 99.

OUTPUT AREAS

DAs for IOCS output areas fall into the following four major categories, depending on record type and the number of input/output areas:

1. Unblocked Records Using Only One Input/Output Area.
2. Unblocked (fixed- or variable-length) Records Using Two Input/Output Areas.
3. Blocked, fixed-length Records.
4. Blocked, variable-length Records.

These major categories and their sub-divisions are discussed below.

1. Unblocked Records Using Only One Input/Output Area.

a. For data written in either the MOVE or LOAD mode. (The programmer can construct records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA	DA	1x8Ø,G
0 2	FIELD1		1,5
0 3	FIELD2		6,1Ø
0 4	FIELD3		15
0 5			

Figure 100.

NOTE 1: The fields indicated in the DA are needed only if the programmer refers to records in the output area.

NOTE 2: Word Marks and labels are needed only if the programmer does processing (or builds records) in the output area.

2. Unblocked (fixed- or variable-length) Records Using Two I/O Areas.

a. With indexing, for data written in either the MOVE or LOAD mode. (The programmer can build records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1x8Ø,X2,Ø,G
0 2	FIELD1		1,5
0 3	FIELD2		6,1Ø
0 4	FIELD3		15
0 5			
0 6	AREA2	DA	1x8Ø
0 7			1,5
0 8			6,1Ø
0 9			

Figure 101.

NOTE 1: The fields indicated in the DA are needed only if the programmer refers to these records in the output area.

NOTE 2: Word Marks and labels are needed only if the programmer does processing in the output area or builds records in the output area.

b. Without indexing, for data written in either the MOVE or LOAD mode. (The programmer cannot construct records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1x8Ø,,G
0 2	AREA2	DA	1x8Ø,,G
0 3			

Figure 102.

NOTE: The programmer may not use any fields in the output area.

3. Blocked, fixed-length Records

a. With indexing, for data written in either the MOVE or LOAD mode. (The programmer can build records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1Øx79,,x2,,Ø,,≠,,G
0 2	FIELD1		1,,5
0 3	FIELD2		6,,1Ø
0 4	FIELD3		11,,15
0 5			
0 6	AREA2	DA	1Øx79,,≠,,G
0 7			1,,5
0 8			6x10
0 9			11,,15
1 0			

Figure 103.

b. Without indexing, for data written in either the MOVE or LOAD mode. (The programmer cannot build records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1Øx79,,≠,,G
0 2	AREA2	DA	1Øx79,,≠,,G
0 3			

Figure 104.

NOTE: The records of examples 103 and 104 are 80 characters in length, including the Record Mark.

When hash totals are to be taken, the high-order Word Mark for the hash-total field must be included in the DAs for this type of file.

4. Blocked, variable-length Records, Without Indexing

a. Without the DTF "VARBUILD" entry, for data which is written in either the MOVE or LOAD mode. (The programmer cannot build records in the output area.)

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1X2000,G
0 2			1,1
0 3	AREA2	DA	1X2000,G
0 4			1,1
0 5			

Figure 105.

NOTE: The "1,1" entries are needed by the IOCS.

b. With the DTF "VARBUILD" Entry, where the VARBUILD entry does not designate an Index Register. For data written in either the MOVE or LOAD mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA1	DA	1X2000,G
0 2			1,1
0 3	AREA2	DA	1X2000,G
0 4			1,1
0 5			

Figure 106.

NOTE 1: The "1,1" entry is needed by the IOCS.

NOTE 2: The programmer may not specify any fields or Word Marks in the output area. When building records in the output area, the programmer must use "A-field control" for all moves, and he must generate all his addresses from the record address placed in the VARBUILD location.

c. With the DTF "VARBUILD" Entry, where the VARBUILD entry designates an Index Register. For data written in either the MOVE or LOAD mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	AREA 1	DA	1X2,6Ø,Ø,X2,Ø,G
0 2			1,1
0 3	FIELD1		5
0 4	FIELD2		1Ø
0 5	FIELD3		15
0 6			
0 7	AREA2	DA	1X2ØØØ,G
0 8			

Figure 107.

NOTE 1: By letting the VARBUILD entry designate an Index Register, the programmer can enter in the DA all the fields needed for the building of records in the output area. All moves must use "A-field control".

NOTE 2: The "1, 1" entry is needed by the IOCS.

NOTE 3: The "X2" entry is used as the DTF "VARBUILD" entry.

NOTE 4: The file does not use a DTF "INDEXREG" entry.

WORK AREAS

a. For data moved in the LOAD mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABEL	DA	1X8Ø,G
0 2	FIELD1		5
0 3	FIELD2		1Ø
0 4	FIELD3		15
0 5			

Figure 108.

b. For data moved in the MOVE mode.

Line 3 5	Label 6 15	Operation 16 20	21 25 30 35 40
0 1	LABEL	DA	1X8Ø,G
0 2	FIELD1		1,5
0 3	FIELD2		6,1Ø
0 4	FIELD3		11,15
0 5			

Figure 109.

NOTE: The "LABELs" of the work areas are used in the DTF "WORKAREA" entries and as operands of the GET TO WORKAREA macro-instructions.

ADDITIONAL INFORMATION FOR PROGRAMMERS

1. Error Treatment

COMPOSITION OF THE ERROR ROUTINE

The IOCS error routines for a particular program are generated by the Autocoder processor during assembly of the object program. IOCS error routines provide complete error checking for all input/output devices specified in the DIOCS entries. Based on these entries, the error routine will be generated for one or two channels, for non-overlap or for overlap-priority processing.

REACTION TO ERRORS

Errors are brought to the operator's attention by means of appropriate error messages printed out on the console printer. Certain conditions cause a programmed halt to enable the operator either to ignore the error or to take corrective action. Read and write commands are retried nine times on DATA CHECK indications before an error message is typed out.

2. Checkpoint and Restart

A Checkpoint-and-Restart Routine will be generated by the Autocoder processor whenever this is specified by the DIOCS "CHECKPOINT" entry. The routine will cause the writing of checkpoint records each time an end-of-reel condition or a CHKPT macro-instruction is encountered.

The checkpoint records contain the information required to restart program which were stopped before completion. The records are written on the tape specified by the DIOCS "CHKPT" entry. Partially run programs can be restarted at any point of the program at which a checkpoint record was written.

3. Record Additions and Deletions

ADDITIONS

Records which were not contained on an input file but were created in a work area can be moved to an output file by the (Format A) PUT macro-instruction.

DELETIONS

Input records may be omitted from output files by omitting the PUT macro-instruction which would have placed these records into the output file. In this case, two GET macro-instructions would appear in a line of coding without an interim PUT macro-instruction.

4. The Size of the IOCS Routines

The number of storage positions occupied by the IOCS routines of a given object program varies with the DIOCS and DTF entries written by the programmer. The size of the IOCS routines depends on both the entries, themselves, and on their combination, and it is therefore difficult to predict the exact size of the IOCS routines for a given program. The information given below is intended to guide the user in judging his core storage requirements. The figures indicate the number of storage positions required by the different IOCS routines.

A. WITH OVERLAP AND PRIORITY SPECIAL FEATURES

1. The Priority Routine: 1,350 storage positions

NOTE: This routine assigns file priorities. If all files are opened at the same time, this routine may be overlaid once the files have been opened. The overlaying will be done automatically by the IOCS if the DIOCS "PRIORITY" entry is omitted.

2. Routines Based on the DIOCS Entries

Maximum: 6,300

Minimum: 2,700

Normal requirement with standard labels: 5,150

Normal requirement with non-standard labels: 4,200

3. Routines Based on the DTF Entries (i.e., File Schedulers)

Normal requirement for blocked records using two I/O Areas: 600-900

Minimum: 300

Maximum: 1,550 (for two-area work files containing variable-length, blocked records that are checked for wrong-length-record conditions).

B. WITHOUT OVERLAP OR PRIORITY SPECIAL FEATURES

1. Routines Based on the DIOCS Entries

Maximum: 5,000

Minimum: 1,300

Normal requirement with standard labels: 4,200

Normal requirement with non-standard labels: 2,600

2. Routines Based on the DTF Entries (i.e., File Schedulers)

Normal requirement for blocked records: 280-350

Minimum: 54

Maximum (work files containing variable-length, blocked records): 580

5. Times Required by the GET and PUT Macro-Instructions

The approximate operating times of GET and PUT macro-instructions for different record formats are shown in Figures 110 and 111. The figures indicate the time required for each record, except the first record of each block (when the End-of-Block Read-Write Routine is entered).

Record Type and Handling	GET times in micro-seconds	
Fixed-length, blocked records.	Maximum*	670
Logical records remain in input area-indexing is used	Minimum	290
Fixed-length, blocked records.	Maximum*	770 + 11.3 x number of characters in record
Logical Records are moved to work area - indexing is not used.	Minimum	435 + 11.3 x number of characters in record
Variable-length, blocked records.	Maximum*	710
Logical Records remain in input area-Indexing is used.	Minimum	330
Variable-length, blocked records	Maximum*	765 + 11.3 x number of characters in record
Logical Records are moved to work area Indexing is not used.	Minimum	430 + 11.3 x number of characters in record

Figure 110. Operating times in micro-seconds for GET macro-instructions.

Record Types and Handling	PUT times in micro-seconds	
Fixed-length, blocked Records	Maximum*	750 + 11.3 x number of characters in record
Logical Records are moved to output area	Minimum	300 + 11.3 x number of characters in record
Fixed-length, blocked Records	Maximum*	610
Records were built in output area	Minimum	230
Variable-length, blocked Records	Maximum*	915 + 11.3 x number of characters in record
Logical Records are moved to output area	Minimum	490 + 11.3 x number of characters in record

Figure 111. Time of PUT macro-instructions in micro-seconds.

*The major and sometimes the only factor determining the difference between maximum and minimum times is the time required to take the Trailer Record and Hash-Total Counts.

6. Times Required by Different Phases of the Priority Interrupt Routine

The approximate times in micro-seconds required by different phases of the Priority Interrupt Routine are indicated in Figure 112. It is assumed that five files have been assigned to the channel on which the priority signal occurs and that the second file interrogated by the Priority Interrupt Routine is found to have an I/O request. The letters in the fifth column of Figure 112 refer to the corresponding letters in Figure 113.

Phase of Priority Interrupt Routine					Time	
No.	FROM	TO	Conditions	Line of Coding in Fig. 113	Maximum	Minimum
1	Interrupt of Main Routine (A)	Return to Main Routine (F)	5 Files (See NOTE 1). No Input/Output Operation Pending	A-B-C-D'-D-E-F	1050	820
2	Interrupt of Main Routine (A)	Start of execution of Input/Output Command (C')	5 Files; The second file interrogated has an I/O request	A-B-C-C'	1390	690
3	Interrupt of Main Routine (A)	Return to Main Routine (F)	5 files; Second file interrogated has an I/O request	A-B-C-C'-D-E-F	1920	1170

Figure 112. Approximate times in Micro-Seconds Required by Different Phases of the Priority Interrupt Routine.

NOTE 1: Add or subtract 50 micro-seconds for each file difference from five. Applies to Phase No. 1 only.

NOTE 2: Above figures assume no read or write errors.

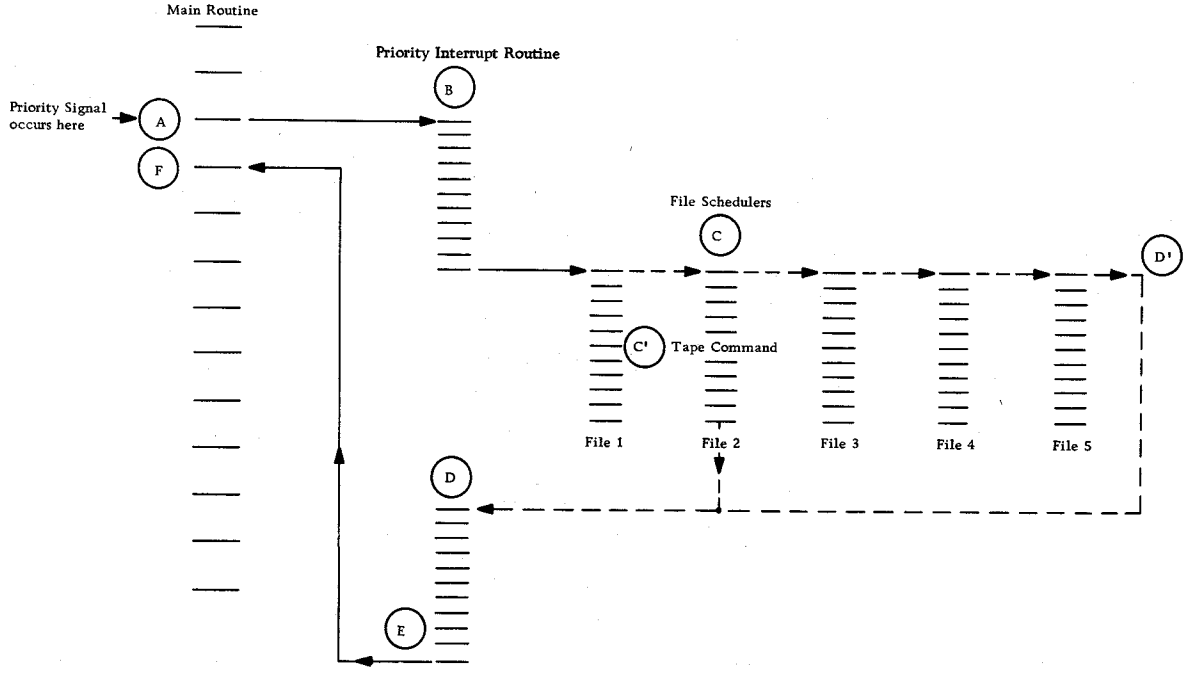


Figure 113. Approximate times Required by Different Phases of the Priority Interrupt Routine.

7. Post-Assembly Modification of Channel*, Drive and Priority Assignment

The channel, drive and priority assignment of a tape file can be modified after assembly as follows:

a. If No Alternate Tape Unit Was Specified

Enter the new	into the location labeled
priority symbol (0...9)	"IOCScuACT-1"
channel symbol (%, ¤, @, *)	"IOCScuACT+1"
drive number (0...9)	"IOCScuACT+2"
mode symbol (M, L)	"IOCScuACT+3"
channel symbol (%, ¤, @, *)	"IOCScuACT+4"
parity symbol (B, U)	"IOCScuACT+5"
drive number (0...9)	"IOCScuACT+6"
channel symbol for error testing (R, X)	"IOCScuACT+7"
channel symbol (%, ¤)	"IOCScuACT+8"
drive number (0...9)	"IOCScuACT+10"

NOTE 1: The channel symbols @ and * are used only for programs using the Overlap and Priority special features.

NOTE 2: A file must be OPENed or re-OPENed after its priority has been modified.

NOTE 3: C = channel number (1 or 2); U = Tape Unit number (0-9).

b. If An Alternate Tape Unit Was Specified

The new priority symbol is entered as described under (a) above.

The channel and drive assignments are changed in the 16-character area labeled IOCSALTD, as indicated in Figure 114 below:

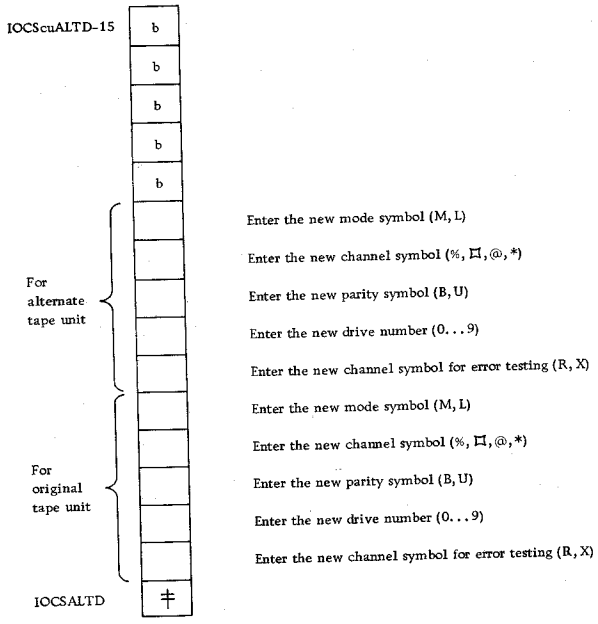


Figure 114. Channel and Drive Modification for tape files with alternate drives.

*See the DIOCS "CHANCHANGE" entry.

INDEX

IBM
International Business Machines Corporation
Data Processing Division, 112 East Post Road, White Plains, N.Y.

Printed in U.S.A. J28-1432-2 11/61