File No. 1410-30
Form C28-0334-1

IBM Systems Reference Library

IBM 1410 Input/Output Control System for Card and Tape Systems

This publication is intended to familiarize users of the IBM 1410 with the many advantages of the 1410 Input/Output Control System for Card and Tape Systems. It includes information for programmers on assembling an IOCS with an object program. It also explains the functions and use of IOCS macro-instructions, and provides details on writing the necessary DIOCS, DTF, and DA entries.

Readers should have a thorough knowledge of the IBM 1410 Data Processing System and IBM 1410 Autocoder.

Preface

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 text explains the functions and use of the IOCS macro-instructions and the necessary DIOCS, DTF, and DA entries.

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

IBM 1410 Principles of Operation, Form A22-0526
IBM 1410 Autocoder, Form C28-0309

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, Model 2

MINOR REVISION (June, 1964)

This publication replaces *IBM 1410 Input/Output Control System for Card and Tape Systems,* Form C28-0334-0; however, it does not make that publication obsolete. The revision incorporates Technical Newsletters N28-1075, N28-2007, and N28-1138. Additional changes are indicated by a vertical bar to the left of the text and a dot (•) before the figure number.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.
Address comments concerning the contents of this publication to: IBM Corporation, Programming Systems Publications, Dept. 637, Neighborhood Road, Kingston, N.Y. 12401

Contents

Advantages of the 1410 IOCS

The IBM 1410 Input/Output Control System (hereafter referred to as the IOCS) is a set of macro-instructions and subroutines that schedule and execute the efficient transfer of data from external storage to core storage, and from core storage to external storage or an on-line output device. The use of these macro-instructions and subroutines frees the programmer from coding his own input/output routines for unit record devices (e. g., the 1402 Card Read Punch and the 1403 Printer) and magnetic tape.*

Merely by using GET, PUT, and related macro-instructions, the user can handle the input and output of logical records. In addition to performing input/output operations, the IOCS automatically blocks and deblocks records, provides the coding required to overlap read/write operations with processing, if the 1410 is equipped with the Overlap and Priority special features, and allows a limited amount of interrupt programming.

Since Input/Output routines constitute approximately half the average program, the IOCS offers users substantial savings in the area of program writing and testing.

Available Input/Output 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.
Schedule the overlapping of read, write, and processing operations – if the 1410 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.

IOCS Check List

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

1. Write one set of DIOCS (Define IOCS) statements.
2. Write one set of DTF (Define The File) statements for each file used by his program. (A file is a collection of records which may be arranged in different ways.)
3. Write proper DA (Define Area) statements for each area used by IOCS.

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

Assembly of Programs Using IOCS

CHANNEL SCHEDULER

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) are to be used by the program to schedule overlapping of input/output and processing operations. If needed, the Channel Schedulers will be written on the intermediate output tape in the form of "one-for-one" symbolic statements (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.

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

*Special sections of the Input/Output Control System for IBM 1405 Disk Storage and IBM 1301 Disk Storage can be incorporated into the IBM 1410 IOCS. See *IBM 1410 Input/Output Control System for 1405 Disk Storage,* Form J28-0233 and *IBM 1410 Input/Output Control System for 1301 Disk Storage,* Form J28-0251.

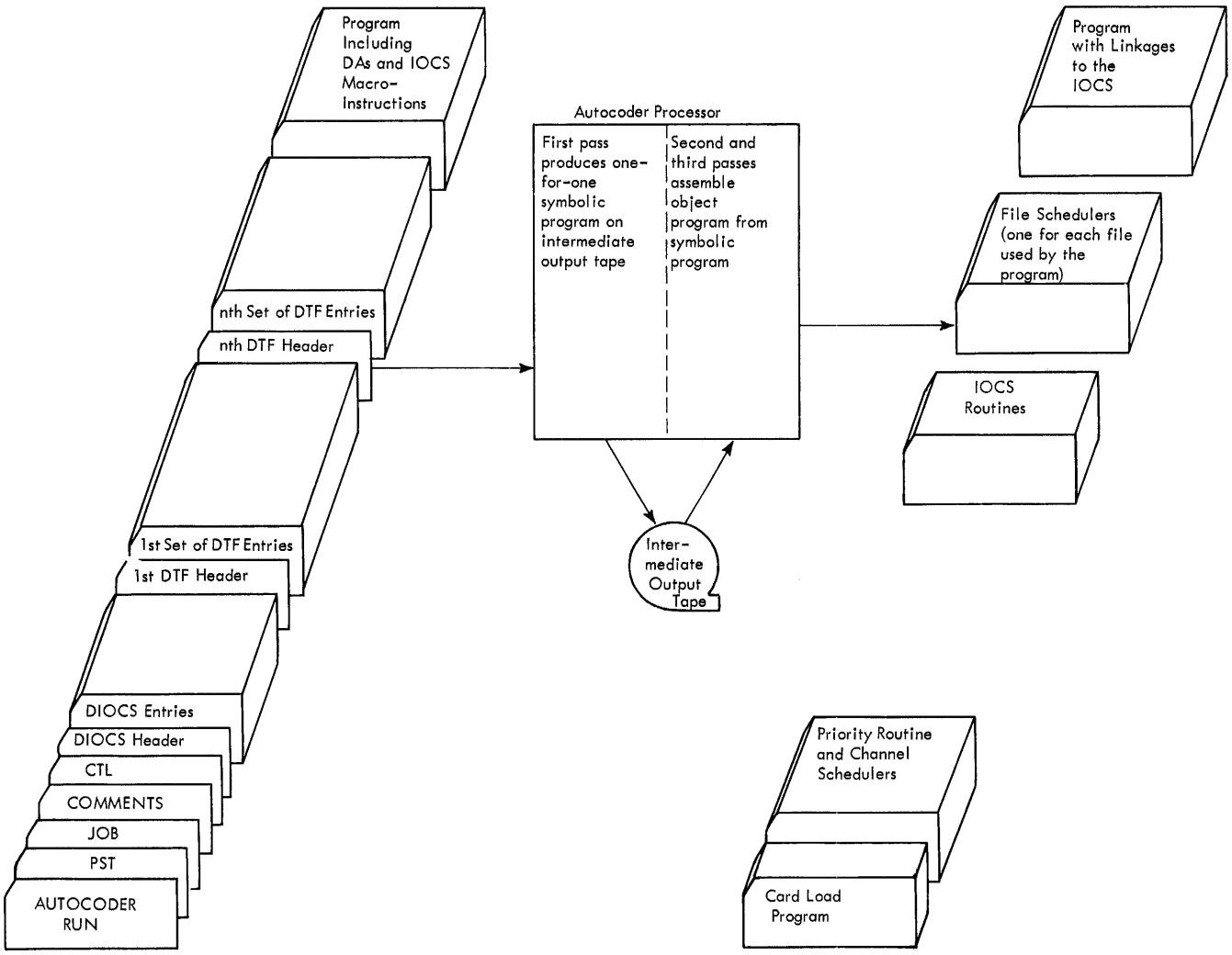

Figure 1. Autocoder Assembly of a Program Using the IOCS

IOCS routines supplied by IBM Programming Systems. 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 (Figure 1).

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.

CHANNEL SCHEDULER

When an interrupt occurs, the Channel Scheduler senses the cause of the interrupt and provides an exit to the appropriate IOCS read/write routine to service the interrupt. Once the interrupt has been serviced, the Channel Scheduler checks to see if any I/O operations are pending. Pending I/O operations are executed according to the priority assigned to them by the user (see "File Scheduler"). The Channel Scheduler then provides a return to the interrupted sequence of instructions.

FILE SCHEDULER

Each time a read/write request is encountered, the File Scheduler determines which I/O area of the specified file will be involved in the operation, and notifies the Channel Scheduler of the request. When the channel specified is free, the Channel Scheduler notifies the File Scheduler that this is the case and the I/O operation is executed within the File Scheduler.

This section 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 22 IOCS Macro-Instructions

This section describes in detail the 22 macro-instructions:

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
IOSYS	Clear Channel

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 OPEN macro-instructions, 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.

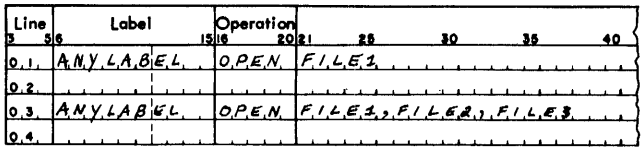

Figure 2

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 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 (unless otherwise directed by 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.
   (See the section on Tape File Tables, under "Additional Information for Programmers.")

Tape files using only one input/output area may be opened at any time.

If the programmer uses either the DIOCS PRIORITY NONOVERLAY or PRIORITY ASSEMBLE entries, two-area tape files may be opened at any time.

If the programmer uses the PRIORITY NONOVERLAY, origin DIOCS entry, two-area tape files can be opened *only* before the user's program overlays the Priority Assignment Routine.

If the programmer does not write *any* DIOCS PRIORITY entry, all two-area tape files should be opened by the first OPEN macro-instruction. They may, however, be closed and reopened later in the program. (Two-area tape files *initially opened after the first* OPEN *macro-*

instruction operate with only the efficiency of a one-area file.)

Use of this macro is not mandatory with unit-record files. (For these files, the OPEN initializes card or line counts.)

IBM 1014 Remote Inquiry Unit Files cannot be opened by an OPEN macro-instruction which is used to open one or more files that are not Remote Inquiry Unit Files.

CLOSE

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

For each *input file* named in the operand of a CLOSE macro-instruction, the IOCS will rewind the tape (unless otherwise directed by the DTF REWIND entry).

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 any partially filled blocks of fixed-length records remaining in the output area. (Partially filled blocks are padded as specified in the DTF PADDING entry. If this entry is omitted, padding is done with blanks.)
3. Write a tape mark, followed (for standard labels) by the trailer label and another tape mark.
4. Rewind the tape (unless otherwise directed by 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. 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.

Line	Label	Operation	
0.1	ANYLAB EL	CLOSE	LAB EL 1
0.2	ANYLAB EL	CLOSE	LAB EL 2,LAB EL 2,LAB EL 3
0.3			

Figure 3

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.

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. The operand in the first line specifies that the tape is to be rewound. The operand in the second line specifies that the tape is to be rewound and unloaded. The absence of an operand in the third line indicates that the tape is not to be rewound.

Line	Label	Operation	
0.1	ANYLAB EL	CLOSD	REWIND
0.2			
0.3	ANYLAB EL	CLOSD	UNLOAD
0.4			
0.5	ANYLAB EL	CLOSD	
0.6			

Figure 4

GET

Use of the GET macro-instruction makes the next logical record available for processing, begins processing end-of-reel conditions, or indicates "wrong-length-record" conditions by branching to the user's wrong-length-record routine. (See DTF WLRADDR entry.) Use of this macro-instruction also causes a record count and hash total to be accumulated and checked against the trailer label, if this has been specified by the DTF TOTALS entry.

The four formats of the GET macro are described below.

FORMAT 1A

Line	Label	Operation	
0.1	ANYLAB EL	GET	INFILE1
0.2			

Figure 5

The operand in Figure 5 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.

For unblocked files using one I/O area, this format of GET will make the next logical record available in the area whose label was defined in the file's DTF IOAREAS entry.

NOTE: The DTF WORKAREA and INDEXREG entries may not be used to define an unblocked file using one I/O area.

For blocked files using one I/O area and all files using two I/O areas, this format will make the next logical record available in one of two areas. (If a file uses two I/O areas, the DIOCS FEATURES entry for the program must include the OVERLAP and PRIORITY operands.)

If an index register has been specified in the file's DTF INDEXREG entry, the record is made available in the area whose label was specified in the file's DTF IOAREAS entry and the address of the record's high-order position is placed in the index register specified.

If the label of a work area has been specified in the file's DTF WORKAREA entry, the next logical record is made available in the area which that label defines.

NOTE: The DTF INDEXREG and WORKAREA entries cannot both be used to define the same file.

FORMAT 1B

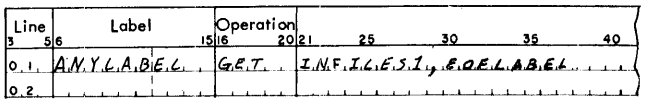

Figure 6

In 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, EOFLABEL, is the label of the user's end-of-file routine.

In addition to the functions performed by Format 1A, this format of the macro-instruction allows the programmer to enter the label of his end-of-file routine at any point in the program where he uses Format 1B of the GET macro-instruction.

FORMAT 2A

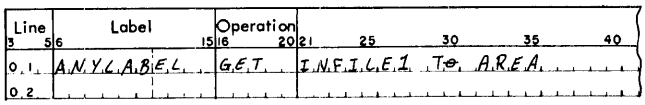

Figure 7

In Figure 7, 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 label given to the area to which the record is to be moved.

For all files *except* unblocked files using one I/O area, this format of GET will make the next logical record available in the area whose label appears as the second entry in the operand.

NOTE: The DTF INDEXREG and WORKAREA entries have no effect when this format of the macro is used.

FORMAT 2B

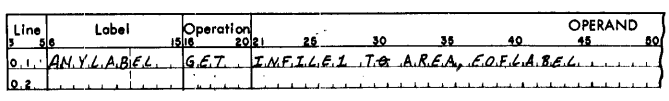

Figure 8

In Figure 8, 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 label given to the area to which the record is to be moved. The third entry (EOFLABEL) is the label of the user's end-of-file routine.

In addition to the functions performed by Format 2A, this format of the macro-instruction allows the programmer to enter the label of his end-of-file routine at any point in the program where he uses Format 2B of the GET macro.

PUT

Use of the PUT macro-instruction causes a processed record to be written on the output file, or begins processing of an end-of-file condition.

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

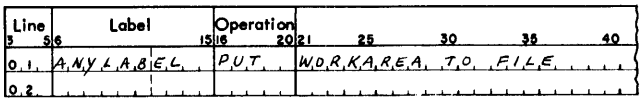

Figure 9

The first entry in the operand in Figure 9 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.

NOTE: Word marks in the work area are moved with the logical record to the output area of the specified file.

FORMAT B

In Figure 10, the first entry in the operand is the name of the file from which the current logical record is taken.

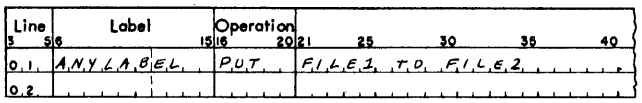

Figure 10

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.
   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.
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

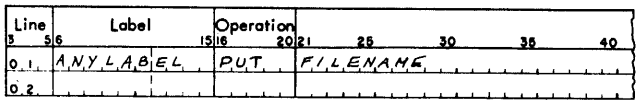

Figure 11

Format C of the PUT macro-instruction, shown in Figure 11, 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 12.

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 12. 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 1410 Principles of Operation,* Form A22-0526.) The macro-instruction is used to select cards into specified pockets when the file uses the DTF CARDPOC entry.

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 first operand of the macro-instruction.
2. Check for error conditions.

The *first 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.

The *second operand* of this macro- instruction is:

x or CHANX where x is the channel to which the 1402 Card Read Punch is attached.

NOTE: If the 1402 Card Read Punch is attached to channel 1 the second operand may be omitted.

The example on line 1 of Figure 13 indicates that the card just read by the 1402 Card Read Punch attached to channel 1 is to be selected into pocket 1.

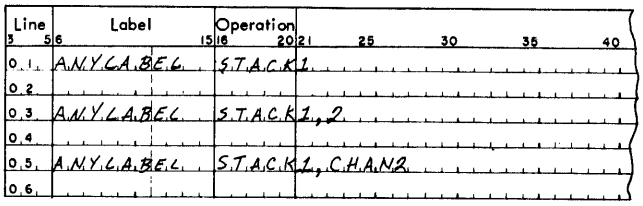

Figure 13

The examples on lines 3 and 5 of Figure 13 indicate that the card just read by the 1402 attached to channel 2 is to be selected into pocket 1.

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

SKIP

This macro-instruction corresponds to the CC (Control Carriage) mnemonic operation code. (See *IBM 1410 Principles of Operation*, Form A22-0526.)

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 first operand of the macro-instruction.
2. Check for error conditions.

The *first operand* of this macro-instruction is the appropriate d-character, as indicated in Figure 15.

The *second operand* of the macro-instruction is:

X or CHANX where X is the channel to which the printer is attached.

NOTE: if the printer is attached to channel 1, the second operand may be omitted.

The example on line 1 of Figure 14 indicates that the carriage of the 1403 Printer attached to channel 1 is to skip to (carriage control tape) channel 1 after printing.

The examples on lines 3 and 5 indicate that the carriage of the 1403 Printer attached to channel 2 is to skip to (carriage control tape) channel 1 after printing.

Line	Label	Operation	25	30	35	40
0.1	ANYLABEL	SKIP	A			
0.2						
0.3	ANYLABEL	SKIP	A,2			
0.4						
0.5	ANYLABEL	SKIP	A,CHAN2			
0.6						

Figure 14

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 15. d-Character for Control Carriage Macro

CONSL

This macro-instruction will generate the coding required to:

1. Perform the console operation specified by the operands of the macro-instruction.
2. Check NOT READY, BUSY, DATA CHECK, and CONDITION indicators.

NOTE: A CONSL macro-instruction with an RCP operand is executed only after the console Inquiry Request key has been pressed. If this key has not been pressed before this CONSL macro-instruction is encountered, the IOCS enters a waiting loop until the console Inquiry Request key is pressed.

The operands of the CONSL macro-instruction are:

1. A letter code indicating the manner in which the console operation is to be performed.
2. The label of the area into or from which information is to be read or written.

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.

The example in Figure 16 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.

Line	Label	Operation	25	30	35	40
0.1	ANYLABEL	CONSL	WCP,MESSAGE			
0.2						

Figure 16

The example in Figure 17 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.

Line	Label	Operation	25	30	35	40
0.1	ANYLABEL	CONSL	RCPO,DATEAREA			
0.2						

Figure 17

NOTE: Message areas must have a group mark with word mark immediately to the right of the low-order position.

RTLBL

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

This macro-instruction will generate the coding required to:

1. Read the nonstandard 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 nonstandard 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 and the MOVE mode, and the use of the Overlap special feature has *not* been specified in the operand of the DIOCS FEATURES entry.

The following code letters *must* be added to the RT code, if the conditions described below apply.

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 use of the Overlap special feature *has* been specified in the operand of the DIOCS FEATURES entry.

NOTE: These code letters are added to the RT code in the order in which they are discussed above.

Line	Label	Operation	
0 1	ANYLABEL	RTLBL	RTW,1,3,,LABLAREA
0 2			

Figure 18

The example in Figure 18 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.

WTLBL

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

This macro-instruction will generate the coding required to:

1. Write the nonstandard 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 nonstandard 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 and the MOVE mode, and the use of the Overlap special feature has *not* been specified in the operand of the DIOCS FEATURES entry.

The following code letters *must* be added to the WT code, if the conditions described below apply.

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 use of the Overlap special feature has been specified in the operand of the DIOCS FEATURES entry.

NOTE: These code letters are added to the WT code in the order in which they are discussed above.

Line	Label	Operation	
0 1	ANYLABEL	WTLBL	WTBO,2,7,,TRAILAREA
0 2			

Figure 19

The example in Figure 19 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: The RTLBL and WTLBL macro-instructions are intended primarily for the convenience of programmers using Exits 2, 5, 6 and 7 to process nonstandard tape labels.

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.

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-instruction will cause the block being built in the output area to be written onto the output file. Blocks consisting of Form 2 Records will be padded, if necessary, as specified in the DTF PADDING entry. (If this entry was omitted for this file, partially filled blocks will be padded with blanks.) For padded

blocks, the output area must contain a record mark to insure proper padding. See sections on DA Entries for IOCS, Output, "Blocked, Fixed-Length Records," and "Form 2 Record Format."

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

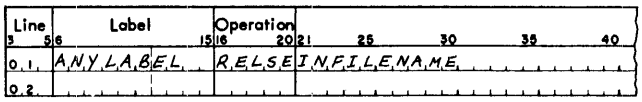

Figure 20

The operand of this macro-instruction (Figure 20) 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 macro-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 TYPELABEL entry.)

For output files, this macro-instruction 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. (No check is made to determine whether an end-of-file conditions exists, and the trailer label is not processed.)

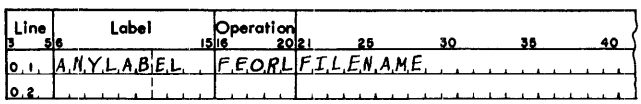

Figure 21

The operand in Figure 21 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 CHKPT macro-instruction (see Figure 22) develops all the coding required to cause the program to branch to the Checkpoint Routine. The Checkpoint Routine causes a checkpoint to be written on the tape unit specified by the DIOCS CHECKPOINT entry. A checkpoint consists of two records. The first record (i.e., the control record) contains: an identifier that notifies the IOCS that a checkpoint is to follow, the number of the checkpoint, and the four address constants required to restart the program. The second record includes the entire contents of core storage. This information allows partially run programs to be restarted at the point in the program where the checkpoint was taken.

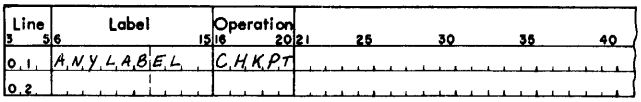

Figure 22

The CHKPT macro-instruction can be entered from the console printer at any time, if: (a) the DIOCS INQUIRY entry has been omitted, and (b) the OVERLAP and PRIORITY operands of the DIOCS FEATURES entry have been specified. The CHKPT macro-instruction is entered as follows: (a) the operator presses the Inquiry Request key, (b) enters CHKPT, and (c) presses the Inquiry Release key.

If a checkpoint record is to be taken, the last 4,400 positions of core storage cannot contain any information provided by DTF or DIOCS entries.

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

RESTARTING FROM A CHECKPOINT

To restart a program from a checkpoint, a control card that conforms to the format shown below must be placed in the card reader immediately following the restart program deck.

COLUMN	CONTENTS
1-7	**CHKPT
8-10	Three-digit checkpoint number of the checkpoint from which the program is to be restarted.
11-12	**
13-14	R% if the checkpoint from which the program is to be restarted was taken on channel 1; RX if the relevant checkpoint was taken on channel 2.
15	Number of the tape drive on which the checkpoint was written.
16	U for even parity. (Checkpoints are always written in even parity.)
17	% if the user's program utilizes channel 1 and *does not* utilize the Overlap and Priority special feature. @ if the user's program utilizes channel 1

COLUMN	CONTENTS
	and *does* utilize the Overlap and Priority special feature.
	□ if the user's program utilizes channel 2 and *does not* utilize the Overlap and Priority special feature.
	* if the user's program utilizes channel 2 and *does* utilize the Overlap and Priority special feature.
	□ if the user's program utilizes channel 1 *and* channel 2 and *does* not utilize the Overlap and Priority special feature.
	* if the user's program utilizes channel 1 *and* channel 2 and *does* utilize the Overlap and Priority special feature.

The restart program reads this control card and searches the checkpoint file for the relevant control record. When the relevant control record has been located, the restart program reads into core storage the information (i.e., the storage dump) that is written on the checkpoint file following the control record. The restart program then uses the seven address constants contained in the relevant control record to obtain the information necessary to restart the program, and proceeds to execute the restart.

MODIFICATION OF THE RESTART PROGRAM

The restart program is provided by IBM in the form of a symbolic card deck. It occupies the first 3,700 positions of core storage. Under certain conditions the programmer must modify this program. These conditions and the necessary modifications are as follows:

Condition: None of the tape files referenced by the program that is to be restarted use standard labels, but one or more use nonstandard labels.

Modification: The programmer must write a routine capable of reading the nonstandard labels, and include that routine in the restart program. The programmer must also place the address of his routine in the appropriate portion of the branch instruction at RSNONSTD+1, and change the NOPWM at RSNONSTD to NOP.

Condition: One or more of the tape files referenced by the program that is to be restarted use two or more header labels on each reel of tape.

Modification: The programmer must write a routine that is capable of reading the additional header labels and include that routine in the restart program. This routine should read the additional labels into the area labeled RSDUMY within the restart program. After each additional label is read, the user's routine should branch to RSEROR. RSEROR is the label of the tape error routine included in the restart program. The programmer must also place a word mark at PGLIN 0132, and insert the address of his routine in the appropriate portion of the branch instruction found at PGLIN 0132.

RDLIN

The programmer may use the RDLIN macro-instruction to:

1. Modify the DTF-specified information against which standard input header labels will be compared.
2. Specify changes in the DTF-specified standard output header labels.

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

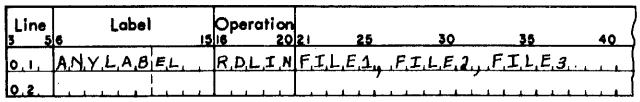

Figure 23

The operand in Figure 23 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.

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. Columns 16-20 must contain the characters RDLIN.

For 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. That is, the columns should contain:

INFORMATION	COLUMNS
RDLIN	16-20
File Serial Number	21-25
Reel Sequence Number	26-30
File Name	31-40
Creation Date	41-45
Retention Cycle	46-50

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.

For 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. All RDLIN cards affecting files opened by the first OPEN macro-instruction encountered in an object program must be placed immediately following the first Execute card (identified by the character E in column 1) in the object program deck, if; (a) the DIOCS PRIORITY entry has been omitted, and (b) the OVERLAP and PRIORITY operands of the DIOCS FEATURES entry have been specified. In all other cases, the placement of RDLIN cards is not critical.

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.

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

IOBSP

This macro-instruction corresponds to the BSP (Backspace Tape) mnemonic operation code. (For this and the following macro-instructions, see *IBM 1410 Principles of Operation*, Form A22-0526.)

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.
2. Check for error conditions.

NOTE: A tape mark is considered a tape record.

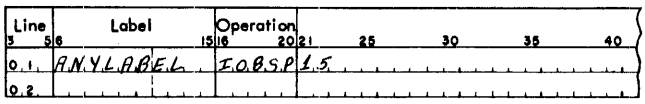

Figure 24

The operand in Figure 24 indicates that tape unit 5 of channel 1 is to be backspaced over one tape record.

IORWD

This macro-instruction corresponds to the RWD (Rewind Tape) mnemonic operation code. This macro-instruction will generate the coding required to:

1. Rewind the tape unit (specified by the operand of this macro-instruction).
2. Check for error conditions.

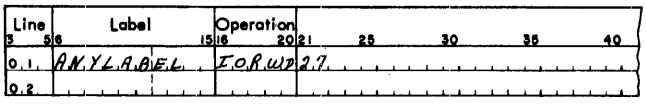

Figure 25

The operand in Figure 25 indicates that tape unit 7 of channel 2 is to be rewound.

IORWU

This macro-instruction corresponds to the RWU (Rewind and Unload) mnemonic operation code. 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.
3. Check for error conditions.

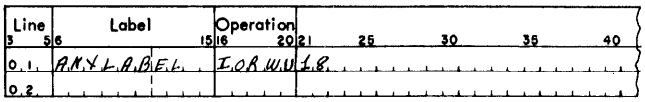

Figure 26

The operand in Figure 26 indicates that tape unit 8 of channel 1 is to be rewound and unloaded.

IOWTM

This macro-instruction corresponds to the WTM (Write Tape Mark) mnemonic operation code. This macro-instruction will generate the coding required to:

1. Write a tape mark (a single-character record in even parity) on the tape specified by the operand of this macro-instruction.
2. Check for error conditions.

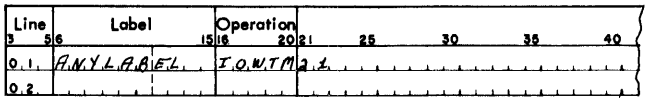

Figure 27

The operand in Figure 27 indicates that a tape mark is to be written on the tape on tape unit 1, channel 2.

PSTAC

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

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 PUT's (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 PUT's (3) and (4) will be selected into the pocket specified by the first PSTAC macro-instruction (I). Cards punched by PUT's (5) and (6) will be selected into the pocket specified by the second PSTAC macro-instruction (II), etc.

A punch output file using the PSTAC macro-instruction *must* have a DTF CARDPOC entry.

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

The *first operand* of the PSTAC macro-instruction identifies the pocket into which cards are to be selected (see Figure 28).

The *second operand* of this macro-instruction is:

x or CHANX where x is the channel to which the punch is attached, or

pp where pp is the operand of this punch file's DTF PREFIX entry, if that entry was used in defining the file.

NOTE: If the 1402 Card Read Punch is attached to channel 1, this operand may be omitted *unless* a DTF PREFIX entry was used in defining the file involved in this operation.

Line	Label	Operation	
0.1	ANYLABEL	PSTAC	8
0.2			
0.3	ANYLABEL	PSTAC	8, 2
0.4			
0.5	ANYLABEL	PSTAC	8, CHAN2
0.6			
0.7	ANYLABEL	PSTAC	8, pp
0.8			

Figure 28

The example on line 1 of Figure 28 indicates that all cards following this macro-instruction are to be selected into pocket 8 of the 1402 Card Read Punch until another PSTAC macro is encountered. The examples on lines 3 and 5 of Figure 28 indicate that all cards are to be selected into pocket 8 of the 1402 attached to channel 2. The example on line 7 indicates that all cards are to be selected into pocket 8 of the 1402 attached to the channel indicated by the pp operand of this file's DTF FILETYPE entry.

RTAPE

This macro-instruction enables the programmer to read a tape record from any tape unit (independently of the GET/PUT logic of the IOCS). Input files for which there is a DTF SCHEDULER NO entry can be read only with this macro-instruction. Also, it is not necessary (though it is permissible) to write a set of DTF entries for files addressed by the RTAPE macro-instruction.

This macro-instruction must never be used for two-area tape files.

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.
2. Check for all error conditions except the wrong-length-record indication.

In addition, if the programmer wishes, this macro-instruction can direct the IOCS to indicate the availability to input records by means of a program switch. (See fourth operand.)

By using another operand, the programmer can also direct the IOCS to store the contents of the B-, E-, or F-address register upon the completion of the read operation. (B Register for non-overlapped operations on either channel, E Register for overlapped operations on channel 1, F Register for overlapped operations on channel 2.) The object program can then use this information to determine the length of individual records from a file of variable-length records. (See sixth operand.)

The *first operand* of the RTAPE macro-instruction is:

RT if the record is to be read in even parity, in the MOVE mode, and use of the Overlap special feature has *not* been specified in the operand of the DIOCS FEATURES entry. The following code letters *must* be added to the RT code, if the conditions described below apply.

B if the record is to be read in odd parity.

G if the record read is to be terminated *only* by the inter-record gap.

W if the record is to be read in the LOAD mode.

O if use of the Overlap special feature *has* been specified in the operand of the DIOCS FEATURES entry.

NOTE: These code letters are added to the RT code in the order in which they are discussed above.

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

c indicates the channel

u indicates the tape unit.

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

The *fourth operand*, which is optional, is the label of a core-storage position that the IOCS can use as a switch to indicate the availability of input records.

(When the RTAPE is issued, the IOCS sets a word mark at this location; when the read operation has been successfully completed, the IOCS clears the word mark.) This operand is used only for overlapped read operations. An RTO operation that has this operand is both scheduled and overlapped by the IOCS.

The *fifth operand* is the label of the user's end-of-file routine for the input tape addressed by the RTAPE macro-instruction. The IOCS transfers control to this routine when an end-of-file condition results from the read operation. This operand must be included if there are no DTF entries for the file. If there *is* a set of DTF entries for the file, then the DTF EOFADDR entry supplies the label of the end-of-file routine. However, if the fifth operand is included for this macro-instruction *in addition* to a DTF EOFADDR entry, the IOCS branches to the end-of-file routine specified by the operand, and no relation between the file and the macro-instruction is established.

If the fourth and fifth operands are *both* specified, the user's end-of-file routine must store the contents of the B-Register, note that an end-of-file condition exists (e.g., set a word mark switch), and branch to the stored contents of the B-Register – without executing any I/O macro-instructions, entering priority alert, or disturbing the High-Low-Equal or Zero Balance indicators. The reason for this procedure is that the user's end-of-file routine has been entered as a result of an interrupt caused by completion of the I/O operation which caused the end-of-file condition.

The *sixth operand,* which is optional, is the label of a five-position area into which the IOCS can store the contents of the B-, E-, or F-address register upon completion of the read operation.

The RTAPE macro-instruction does not generate the coding required to increment block counts, record counts, or hash totals. However, the programmer can himself increment these counts at the following points:

1. If the affected file has been defined (by means of DTF entries) the block count can be incremented at IOCSCUTBC where u is the tape unit the file is using and c is the channel to which that unit is attached. If the program is restarted from a checkpoint, the relevant tapes cannot be properly repositioned unless the affected block counts are up to date.
2. If the first operand (i.e., RECORD) of both the DIOCS COUNTS entry and the DTF TOTALS entry has been specified, the record count of the affected file can be incremented at IOCSCUTRC where c and u have the same values noted above.
3. If the second operand of the DIOCS COUNTS entry (i.e., HASH) and the second operand of the DTF TOTALS entry (i.e., an appropriate integer) have been specified, the hash total of the affected file can be incremented at IOCSCUTHT where c and u have the same values noted above.

The fields at IOCSCUTBC, IOCSCUTRC and IOCSCUTHT are set to blanks by the IOCS each time an end-of-reel condition occurs on the affected file.

The contents of the fields at IOCSCUTBC, IOCSCUTRC and IOCSCUTHT are included by the IOCS in the trailer labels of reels belonging to the affected file if standard labels have been specified for that file.

Line	Label	Operation	OPERAND
0 1	ANYLABEL	RTAPE	R,T,B,0,,2,3,,AREA1,,INSW,,STORADREG
0 2			

Figure 29

The operand field in Figure 29 specifies that a record is to be read from tape unit 3, which is on channel 2, into AREA1. The record is to be read in odd parity, in the MOVE mode, and the read operation is to be overlapped with processing. The entry also directs the IOCS to set a word mark at INSW when this macro-instruction is issued by the program. When the read operation has been successfully completed, the IOCS will clear the word mark at INSW and store the contents of the F-address register (channel 2 operation) into STORADREG.

NOTE: Assume that the file on tape 23 has been defined by a set of DTF entries. Because it is an input file, the DTF EOFADDR entry was required. Therefore, the fifth operand of this macro-instruction need not be included, but must be indicated by a comma. (The comma enables the Autocoder Processor to recognize STORADREG as the sixth operand, rather than the fifth.)

WTAPE

This macro-instruction enables the programmer to write a record on any tape unit (independently of the GET/PUT logic of the IOCS). Output files for which there is a DTF SCHEDULER NO entry can be written only with this macro-instruction. Also, it is not necessary (though it is permissible) to write a set of DTF entries for files addressed by the WTAPE macro-instruction.

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.
2. Check for all error conditions.

In addition, if the programmer wishes, this macro-instruction can direct the IOCS to indicate the availability of output areas by means of a program switch. (See fourth operand.)

The *first operand* of the WTAPE macro-instruction is:

WT if the record is to be written in even parity, in the MOVE mode, and use of the Overlap

special feature has *not* been specified in the operand of the DIOCS FEATURES entry.

The following code letters *must* be added to the WT code, if the conditions described below apply.

B if the record is to be written in odd parity.

E if the write operation is not to be terminated until the end of core storage has been reached.

W if the record is to be written in the Load mode.

O if use of the Overlap special feature *has* been specified in the operand of the DIOCS FEATURES entry.

NOTE: These code letters are added to the WT code in the order in which they are discussed above.

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

c indicates the channel

u indicates the tape unit.

The *third operand* is the label of the output area from which the record is to be written.

The *fourth operand,* which is optional, is the label of a core-storage position that the IOCS can use as a switch to indicate the availability of the output area. (When the WTAPE is issued, the IOCS sets a word mark at this location; when the write operation has been successfully completed, the IOCS clears the word mark.) This operand is used only for overlapped write operations. A WTO operation that has this operand is both scheduled and overlapped by the IOCS.

The *fifth operand* is the label of the user's end-of-reel routine for the output tape addressed by the WTAPE macro-instruction. The IOCS transfers control to this routine when an end-of-reel condition results from the write operation. This operand must be included if there are no DTF entries for the output file. (If there is a set of DTF entries for the file, and this operand is not used, the IOCS automatically handles the end-of-reel condition in accordance with the specifications of the DTF entries.)

If the fourth and fifth operands are *both* specified, the user's end-of-file routine must store the contents of the B-Register, note that an end-of-file condition exists, and branch to the stored contents of the B-Register – without executing any I/O macro-instructions, entering priority alert, or disturbing the Hi-Low-Equal or Zero Balance indicators. The reason for this procedure is that the user's end-of-file routine has been entered as a result of an interrupt caused by completion of the I/O operation that caused the end-of-file condition.

The WTAPE macro-instruction does not generate the coding required to increment block counts, record counts, or hash totals. However, the programmer can himself increment these counts at the following points:

1. If the affected file has been defined (by means of DTF entries) the block count can be incremented at IOCSCUTBC where c is the tape unit the file is using and u is the channel to which the unit is attached. If the program is restarted from a checkpoint, the relevant tapes cannot be repositioned properly unless the affected block counts are up to date.
2. If the first operand (i.e., RECORD) of both the DIOCS COUNTS entry and the DTF TOTALS entry has been specified, the record count of the affected file can be incremented at IOCSCUTRC where c and u have the values noted above.
3. If the second operand of the DIOCS COUNTS entry (i.e., HASH) and the second operand of the DTF TOTALS entry (i.e., the appropriate integer) have been specified, the hash total of the affected file can be incremented at IOCSCUTHT where c and u have the same values noted above.

The fields at IOCSCUTBC, IOCSCUTRC, and IOCSCUTHT are set to blanks by the IOCS each time an end-of-reel condition occurs on the affected file.

The contents of the fields at IOCSCUTBC, IOCSCUTRC, and IOCSCUTHT are included by the IOCS in the trailer labels of reels belonging to the affected file if standard labels have been specified for that file.

The operand field in Figure 30 specifies that a record is to be written from AREA2 onto tape unit 3, which is on channel 2. The record is to be written in odd parity, in the MOVE mode, and the write operation is to be overlapped with processing. The entry also directs the IOCS to set a word mark at OUTSW when this macro-instruction is issued by the program, and to clear that word mark when the write operation has been successfully completed. If an end-of-reel condition occurs, the IOCS will transfer control to the routine labeled EORRTN.

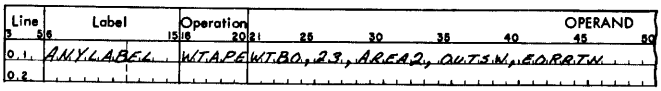

Figure 30

IOSYS

This macro-instruction allows the programmer to issue I/O instructions (other than IOCS I/O macro-instructions) to I/O devices on either channel, during execution of any object program utilizing the IOCS with the Overlap and Priority special features. This macro-instruction has two formats.

FORMAT A

The operand on the first line of Figure 31 specifies that all channels defined in the DIOCS CHANX entry, or entries, are to be cleared. The operand on the second line specifies that channel 1 is to be cleared. The operand on the third line specifies that channel 2 is to be cleared. The operand on the fourth line specifies that both channel 1 and channel 2 are to be cleared.

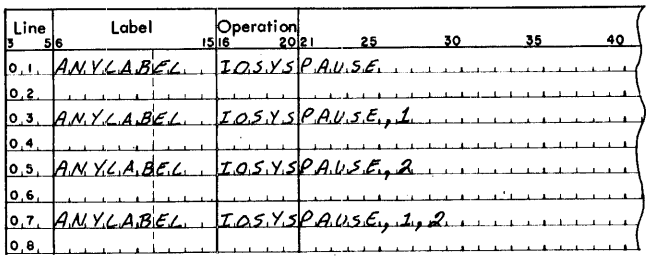

Figure 31

This format of the macro-instruction will establish an uninterruptible waiting loop in the Channel Scheduler(s) specified. Then, it will terminate the waiting loop when the currently overlapping I/O operation in process on the channel(s) specified (if any) is completed and errors resulting from the I/O operation (if any) are corrected. The next sequential instruction in the user's program is then executed, but the IOCS does not initiate any I/O operations that are pending on the channel(s).

FORMAT B

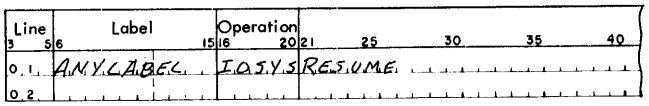

Figure 32

This format of the macro-instruction will return control of the program to the IOCS and allow the IOCS to initate any I/O operations that are pending on the channel or channels cleared by IOSYS PAUSE. This form is used to negate the effect of IOSYS PAUSE (Figure 32).

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.

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 CHANX entries are mandatory. The remaining entries are not always required.

List of DIOCS Entries

This section describes the purpose of each of the following entries:

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

The DIOCS Header Line

The first DIOCS entry (Figure 33) 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 such as AUTOCODER RUN, PST, JOB, COMMENTS and CTL) to enter the system during Autocoder assembly.

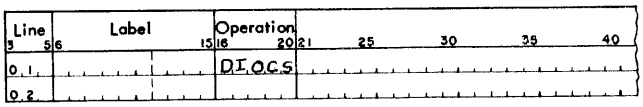

Figure 33

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.

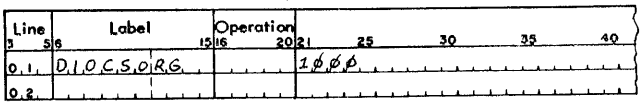

Figure 34

The entry in Figure 34 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 397.

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:

OVERLAP if the program uses the Overlap special feature. (The present IOCS does not provide overlapping without the Priority 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.

Line	Label	Operation	
01	FEATURES		OVERLAP,PRIORITY
02			

Figure 35

The entry in Figure 35 indicates that the object program uses the channel 2 and the Overlap and Priority special features.

CHANx

The x in CHANx indicates the channel to which the input/output devices (specified in the operand field) are attached. Therefore, if all devices that are to be controlled by the IOCS are on one channel, one CHANX entry is included in the source deck. If the input/output devices are on two channels, then two CHANX entries are included in the source deck: one entry specifying all the devices attached to channel 1 and the other specifying all the devices attached to channel 2.

CHAN1 or CHAN2 is written in the label field and one or more of the following devices can be specified in the operand field:

OPERAND	DEVICE
TAPE	Magnetic Tape Unit
READER	Card Reader
PUNCH	Card Punch
PRINTER	132-character Printer (Model 2)
1405	1405 Disk Storage
1301	1301 Disk Storage

Line	Label	Operation	OPERAND
01	CHAN1		TAPE,READER,PUNCH,PRINTER
02	CHAN2		1301,TAPE
03			

Figure 36

The first entry in Figure 36 indicates to the IOCS that the object program requires coding for the following channel 1 operations: tape, card (both input and output), and printer. The second entry indicates that the program also requires coding for tape and 1301 Disk operations on channel 2.

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 nonstandard tape labels and none of the files use standard labels.

MIXED if some files have standard labels and some files have either nonstandard 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. 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. A tape mark between the header label and the first block of records on each reel of tape is recommended, but not required. (If a tape mark follows a nonstandard header label the IOCS will assume that end of file has been reached. However, the programmer may circumvent this condition by taking a label exit and issuing an RTLBL macro-instruction.)

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

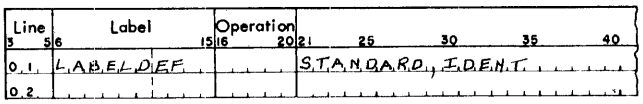

Figure 37

The entry in Figure 37 indicates that:

1. The program uses only standard IBM tape labels.
2. The program contains one or more files which require only a check of the header label's identification field.
3. None of the header labels are to be completely checked.

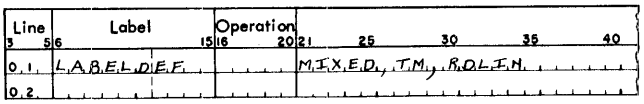

Figure 38

The entry in Figure 38 indicates that some files of the program have standard labels and that some files have either nonstandard 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.

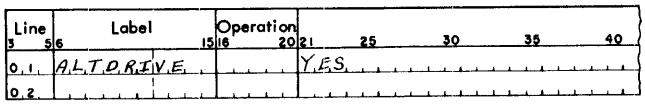

Figure 39

The entry in Figure 39 indicates that one or more tape files used by the program are to be provided with alternate tape units.

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 or the FEORL macro is executed, 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. (See the section "The Eight IOCS Exits.")

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.

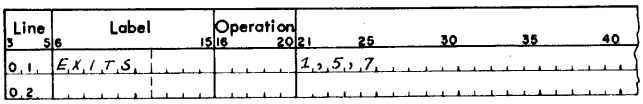

Figure 40

The entry in Figure 40 indicates that the program uses exits 1, 5, and 7.

COUNTS

This entry is required if any of the files used by the program require record counts or hash totals. The possible operands of the COUNTS entry are:

RECORD if any files require record counts.

HASH if any files require hash totals.

If both operands are supplied they must be separated by commas, and may be listed in any order.

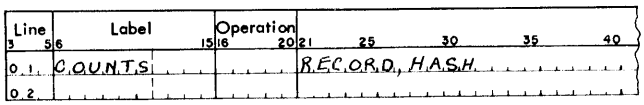

Figure 41

The entry in Figure 41 indicates that a record count and a hash total are required by at least one file of the program. (Block counts are taken and inserted into the trailer label automatically by the IOCS.)

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 closed, and when end-of-reel and end-of-file conditions are encountered.

The operand of the RWDOPTION entry can be *either* NORWD *or* UNLOAD, as shown in Figures 42 and 43.

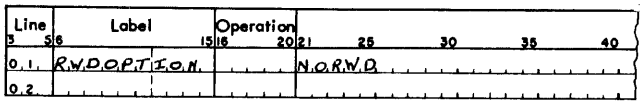

Figure 42

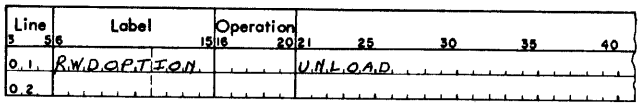

Figure 43

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.

The NORWD operand provides the coding required to eliminate the otherwise automatic rewind that is given when a file is opened and closed, and when end-of-file and end-of-reel conditions occur. (The coding provided by this operand is *part* of the coding provided by the UNLOAD operand.)

Note that the DIOCS RWDOPTION entry is related to the DTF REWIND entry (described later in this publication). That DTF entry can be used to specify whether a *particular* file is to make use of the coding provided by one of the operands of this DIOCS entry.

READERROR

This entry is needed if the object program uses tape files. The operands of the READERROR entry are shown in Figure 44. If only the SCAN operand is specified, the IOCS will generate a *SCAN routine. This routine examines the affected input block (Form 2 and Form 4 data records) or the affected input record (Form 1 and Form 3), high order to low order, and types out the location of asterisks. (Invalid characters entering core storage during read operations are converted into asterisks if the ASTERISK INSERT switch is set to ON.) (See *IBM 1410 Principles of Operation*, Form A22-0526.)

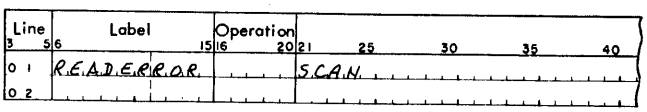

Figure 44

If a read error occurs, a message is printed on the console typewriter by the IOCS and a waiting loop is established. The operator then has four options. He may enter the code word *SCAN for an asterisk scan of the affected block or record. He may enter the code word PROC to process the block or record. He may enter the code word RETRY to retry the read operation, or he may enter the code word SKIP to skip the block or record.

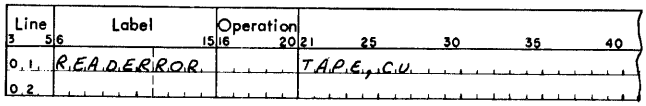

Figure 45

If only the TAPE,cu operand is specified (Figure 45), the IOCS will generate a DUMP routine. This routine writes the affected block on the tape specified by cu, where c is the channel and u is the tape unit. No *SCAN routine is generated.

If a read error occurs, the DUMP routine is automatically executed and a message is printed on the console typewriter by the IOCS. No options are available unless the user has modified IOCSEROPTN.

The character c of the cu operand may be 2 for channel 2 only if all tapes to be read or written by the object program are to be read or written on channel 2. Otherwise, this entry must be 1 for channel 1.

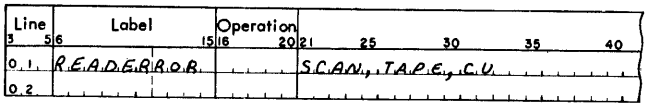

Figure 46

If both SCAN and TAPE,cu operands are specified (Figure 46), a *SCAN and a DUMP routine are generated by the IOCS. If a read error occurs, a message is printed on the console typewriter and a waiting loop is established. The machine operator then has five options. He may enter any of the following code words: DUMP, *SCAN, PROC, RETRY, and SKIP.

SCAN must precede the TAPE,cu operand. The SCAN and TAPE,cu operands must be separated by a comma.

The tape file specified in the operand need not be defined by a set of DTF entries. A set, however, may be written. This would be especially valuable if the programmer wished to check tape labels.

If the *SCAN or the DUMP option is executed, the IOCS again enters a waiting loop and writes the same message, thereby allowing the machine operator to continue processing by selecting another option.

If the READERROR entry is omitted, *SCAN and DUMP routines are not generated by the IOCS. If a read error occurs, a message is printed on the console typewriter by the IOCS, and a waiting loop is established. The ma-

chine operator then has three options. He may enter any of the following code words: PROC, RETRY, and SKIP.

THE READ-ERROR OPTION CONTROL FIELD

This field is generated during the Autocoder assembly of the object program, but may be modified by the user at any time. It is a six-character field labeled IOCSEROPTN. The label addresses the low-order position of the field. This field, depending on the READERROR operand or operands specified, will have one of the formats shown below:

1. SCAN operand only, @b*SRPb@.
2. TAPE,cu operand only, @DbSRPD@.
3. SCAN and TAPE,cu operands both specified, @D*SRPb@.
4. If the READERROR entry is omitted, this field will have the following format: @bbSRPb@.

In the formats shown, a low-order D indicates an automatic dump will occur. P stands for the code word PROC. R stands for the code word RETRY. S stands for the code word SKIP, * stands for the code word *SCAN, and a high-order D stands for the code word DUMP.

PRIORITY

The PRIORITY entry indicates to the IOCS how the user wishes to use the IOCS Priority Assignment Routine. (The Priority Assignment Routine controls the order in which the file schedulers are arranged. It is this order of arrangement that in turn determines the priority of input/output requests for the files.) This entry may be omitted if the object program meets *both* of the following conditions:

1. All two-area tape files are opened by the first OPEN macro-instruction (one-area files may be opened at any time).
2. No post-assembly modifications are made to any file's channel or priority assignments after the first OPEN macro-instruction. (If the PRIORITY entry is omitted, changes to drive assignments may be made at any time.)

If the PRIORITY entry is omitted, the IOCS overlays the Priority Assignment Routine after the first OPEN macro-instruction is encountered in the object program.

Any *one* of three entries is used in the operand field of this entry.

1. NONOVERLAY

If this operand is used, the IOCS does not overlay the Priority Assignment Routine. All tape files may be opened at any time during the running of the object program, and post-assembly modification of channel, drive and priority assignments can be made at any time.

2. NONOVERLAY, origin

The "origin" is the five-digit address of a location where the programmer wishes the IOCS to place the Priority Assignment Routine.

If this operand is used, the IOCS generates the Priority Assignment Routine starting at the specified origin address and, of course, does not overlay the routine.

This operand allows the programmer to overlay the Priority Assignment Routine at a time most convenient to his program. Two-area tape files cannot be opened after this routine is overlaid; neither the channel, nor the priority, nor the drive of *any* tape files can be changed after the routine is overlaid.

3. ASSEMBLE

If this operand is used, the IOCS does not generate a Priority Assignment Routine for the object program. The IOCS assigns file priority according to the order in which the sets of DTF entries are arranged in the source deck. (This effects a saving of approximately 500 core-storage positions.) Although all tape files may be opened at any time, neither the channel nor the priority assignments can be changed after assembly. (Drive assignments, however, can be changed at any time.)

The programmer need not include PRIORITY ASSEMBLE in his DIOCS entries to suppress generation of the priority assignment routine if no DTF entries are included in the source program. The omission of DTF entries in the source program will suppress generation of this routine.

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. However, checkpoint records may *not* be written on reels of a tape file whose DTF entries include the ALTTAPE entry.

If any input tape contains checkpoint records, the IOCS will recognize and ignore them *only if* the DTF FILETYPE entry for that particular tape input file contains the operand CHKPT as well as the two other operands TAPE and INPUT.

If any end-of-reel condition occurs during the writing of a checkpoint record, the tape is backspaced over the record that was just written, and no CHKPT is written as a result of this macro-instruction.

The tape unit specified by the operand can be one for which no file was defined by a set of DTF entries. If such a unit is specified, the user must insure that the tape is rewound to load point before the object program is executed, and that an end-of-reel condition will not occur. (Since there are no DTF entries for the tape unit,

the IOCS has no specifications for rewinding the tape or for end-of-reel procedure for that tape.)

A set of DTF entries, however, may be written if the programmer wishes to check the labels of the tape on which the checkpoint records are to be written. If a DTF is written, the file should be opened, closed, etc., as if it were a normal file.

Checkpoint records may not be written on odd parity data files.

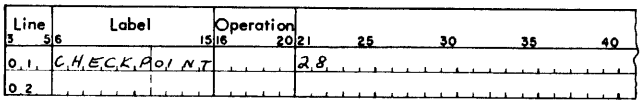

Figure 47

The entry in Figure 47 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 48.

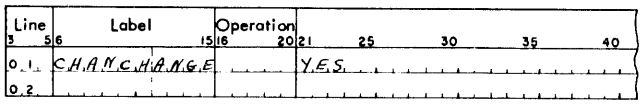

Figure 48

If the DIOCS CHANCHANGE entry is used with programs using the Overlap and Priority special features, approximately 200 additional storage locations are required.

The additional storage requirement for programs that do not use the Overlap and Priority special features is negligible.

The CHANCHANGE entry cannot be used if the DIOCS PRIORITY ASSEMBLE entry is used.

INQUIRY
UR1REQUEST
UR2REQUEST

These entries are provided to facilitate interrupt programming. The operand of the INQUIRY entry (Figure 49). is the label of a routine the user has written to service interrupts caused by depressing the Inquiry Request Key on the IBM 1415 Console.

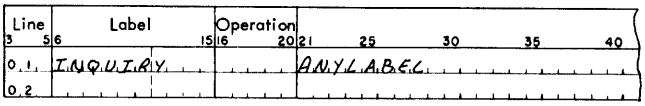

Figure 49

The operand of the UR1REQUEST entry (Figure 50) is the label of a routine the user has written to service interrupts from unit record equipment attached to channel 1.

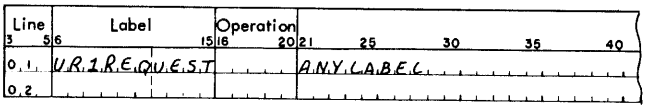

Figure 50

The operand of the UR2REQUEST entry (Figure 51) is the label of a routine the user has written to service interrupts from unit-record equipment attached to channel 2.

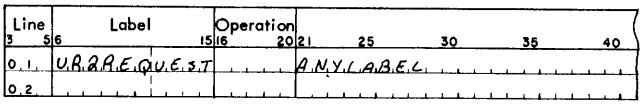

Figure 51

The interrupts these routines service occur upon: (a) the completion of the transfer of a group of data from an IBM 1402 Card Read Punch or an IBM 1011 Paper Tape Reader to the appropriate buffer in the IBM 1414 Input/Output Synchronizer, or (b) the completion of the transfer of a group of data from the buffer in the IBM 1414 Input/Output Synchronizer to an IBM 1402 Card Read Punch or an IBM 1403 Printer.

To obtain the unit-record equipment interrupts discussed above, the Priority Select Switch on the IBM 1415 Console must be turned to CARD READER, CARD PUNCH, PRINTER, or PAPER TAPE READER (whichever is appropriate), and the appropriate Priority Select On-Off Key must be depressed.

ACTION OF THE IOCS

When a console inquiry or unit-record interrupt occurs, the IOCS determines whether the relevant user's routine is servicing a previous interrupt. If the user's routine is not servicing a previous interrupt the IOCS: (a) saves the status of the High-Low-Equal and Zero Balance Indicators, the contents of index registers 13, 14 and 15, and associates this information with the user's routine; (b) saves the return address to the interrupted instruction and associates it with the user's routine; (c) leaves clear the appropriate channel by insuring that the next input/output operation scheduled for execu-

tion on that channel is not initiated; and (d) causes a branch that removes the program from Priority Alert to be executed to the user's routine. If the user's routine is servicing a previous interrupt, the IOCS simply causes a branch to be executed in the noninterruptible mode to the user's routine.

Re-entry into the user's unit record interrupt routine may occur if a GET or PUT is issued against a unit-record file within the user's unit-record interrupt routine. Re-entry into the user's console inquiry interrupt routine may occur if any IOCS macro-instruction except IOSYS is issued within the user's console inquiry interrupt routine.

ACTION OF THE USER'S ROUTINE

User-written routines that service console inquiry and unit-record interrupts must conform to the programming scheme illustrated in Figure 52 and discussed below. The steps of the suggested programming scheme for user-written interrupt routines is as follows:

1. A switch, hereafter referred to as the Nest Switch, is tested.

2. If the Nest Switch is ON (indicating that the routine is servicing a previous interrupt) the fact that another interrupt has occurred (to be serviced by the routine) is noted, and a branch that removes the program from Priority Alert (BXPA) is executed to an address labeled $CS1ENT.

If the current interrupt originated from the console printer, the printer must be read into an alternate area (i.e., an area other than the area into which the routine normally reads the console printer) by means of the CONSL macro-instruction before the branch is executed to $CS1ENT.

If the current interrupt originated from the console printer, the user's interrupt routine must not disturb the contents at index registers 13-15.

3. If the Nest Switch is OFF (indicating that the routine is free to service the current interrupt), routines that service console inquiry interrupts must read the console printer by means of the CONSL macro-instruction, and routines that service unit-record interrupts should issue a GET macro-instruction against the file which refers to the unit-record device from which the interrupt originated.

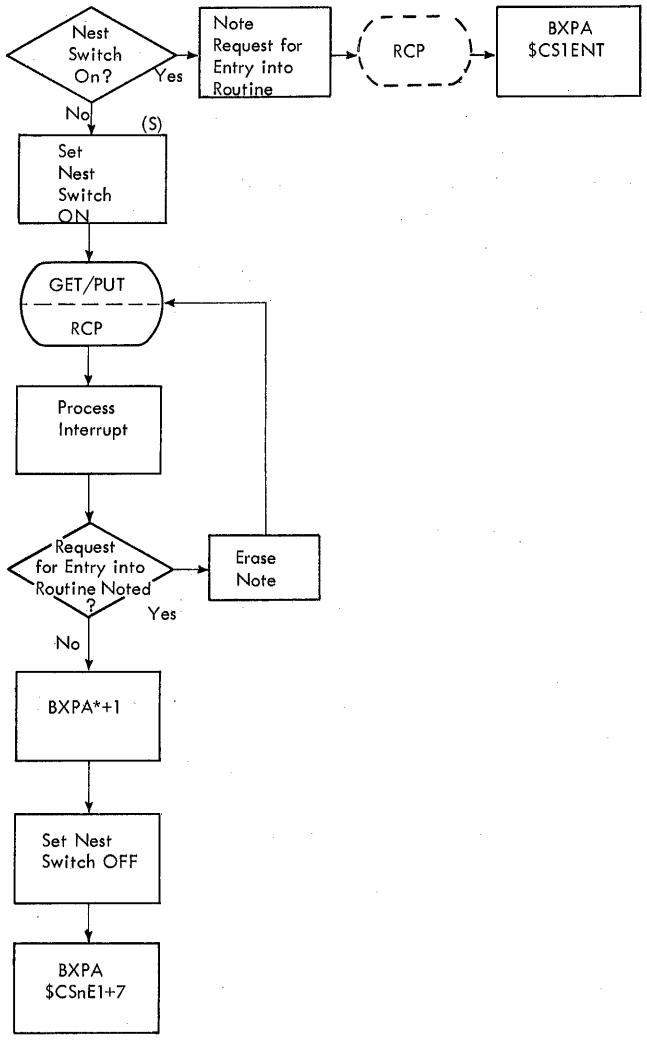

Figure 52

If two or more GET macro-instructions are issued against the file, the sequence of instructions shown in Figure 53 must precede all but the first GET macro-instruction.

4. Once Step 3 has been executed, the routine can process the information obtained by the appropriate macro-instruction as the programmer sees fit (i.e., service the interrupt).

5. When this information has been processed, a check is made to determine if another interrupt (to be serviced by the routine) has been noted.

Line	Label	Operation	OPERAND
0,1		IOSYS	PAUSE,n (This instruction clears the channel represented by "n")
0,2		NRO	0,0 (These instructions cause the IOCS to pause until the relevant
0,3		BAL	1,URERR unit record device is free)
0,4		AUPR1	*+1 (This instruction suppresses unit record interrupts)
0,5			

Figure 53

6. If another interrupt has been noted, the note is erased and a branch is executed to the macro-instruction in Step 3.

7. If another interrupt has not been noted, the program is taken out of Priority Alert (the macro-instruction issued in Step 3 caused the program to be placed in Priority Alert) by means of the instruction BXPA *+1; the Nest Switch is set OFF, and a branch (BXPA) is executed to the address labeled $CSNE1+7. (The n in the label $CSNE1+7 represents the channel from which the interrupt originated.)

The DTF Entries

In addition to the DIOCS entries, the programmer who wishes to use the Input/Output Control System should 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 three 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.

NOTE: If the programmer does not include any DTF entries in his source program, a minimal IOCS will be generated. The OPEN, CLOSE, GET, and PUT macro-instructions cannot be used if a minimal IOCS is generated. All other macro-instructions can be used.

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	TOTALS
FILETYPE	TYPELABEL
PREFIX	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

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 (Figure 54).

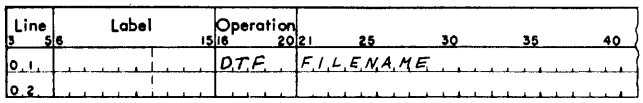

Figure 54

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.

CHAN2 if the file described by the DTF is read or written by the 1402 Card Read Punch (or the 1403 Printer) attached to channel 2.

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.

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

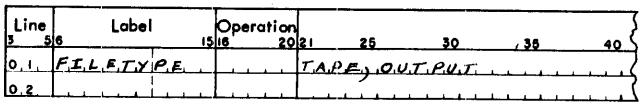

Figure 55

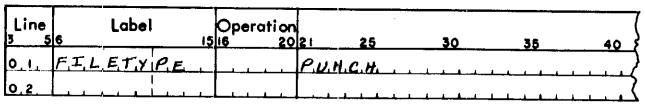

Figure 56

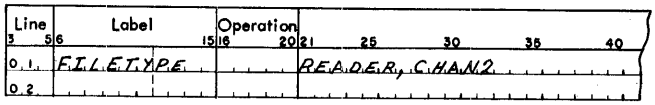

Figure 57

The entry in Figure 55 indicates that the DTF describes a tape output file. The entry in Figure 56 indicates that the DTF describes a punch (and hence an output) file. The entry in Figure 57 indicates that the file described by this DTF is a card input file that will be read by the 1402 Card Read Punch attached to channel 2.

PREFIX

This entry is required if two files reference the same tape unit, 1402 Card Read Punch, or 1403 Printer. The operand is xx, where xx can be any two letters that are not used by other PREFIX operands.

This entry insures that the File Schedulers associated with these files will not have the same label.

Figure 58 illustrates the use of the PREFIX entry to define two files that refer to the same I/O unit. Lines 01-09x define two files that refer to tape unit 3 on channel 1. Lines 10-16x define two files that refer to the 1403 Printer on channel 2. File Scheduler labels for INWKFL will have the format IOCS13LABEL. The labels for OUTWKFL will have the format IOCSAALABEL. Similarly, the File Scheduler labels for the channel 2 printer files will be IOCS2LLABEL and IOCSZZLABEL. (Autocoder designation for the 1403 Printer is the letter L.)

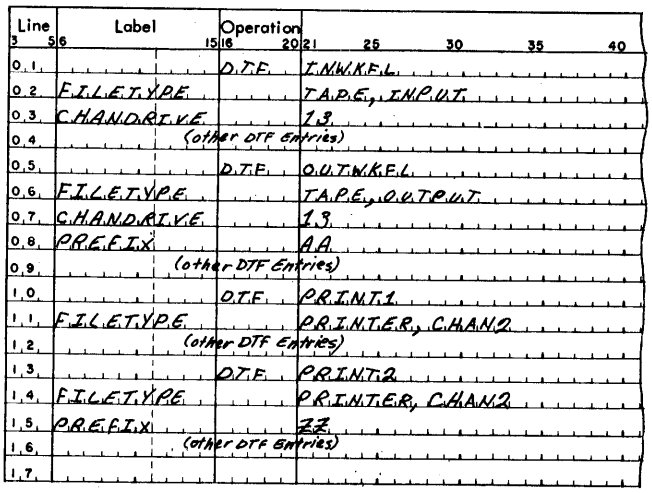

Figure 58

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. (See comments regarding the MOVE and LOAD modes and parity considerations that follow.)

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.

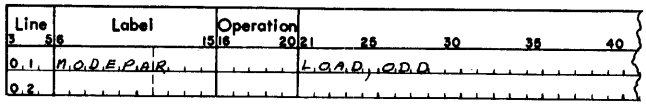

Figure 59

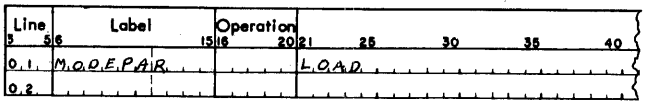

Figure 60

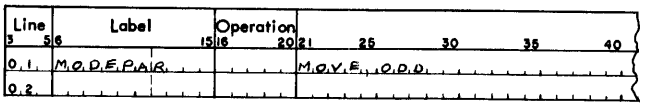

Figure 61

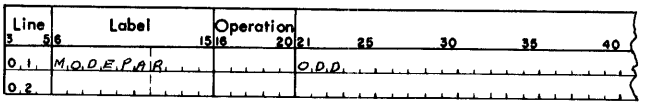

Figure 62

The entry in Figure 59 indicates that the tape file described by this DTF is to be read or written in the LOAD mode and in odd parity. The entry in Figure 60

indicates that the file is to be read or written in the LOAD mode, using even parity. The entry in Figure 61 indicates that the file is to be read or written in the MOVE mode and in odd parity. The entry in Figure 62 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.

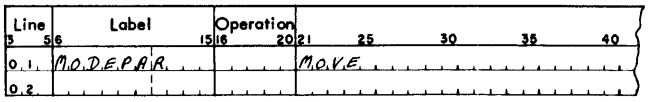

Figure 63

The entry in Figure 63 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 are read or written. Word marks in storage are undisturbed, but they are not written out as word separator characters during write operations. 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.

If two or more sets of DTF entries use the same CHANDRIVE operand, a DTF PREFIX entry must be written for all but one of them.

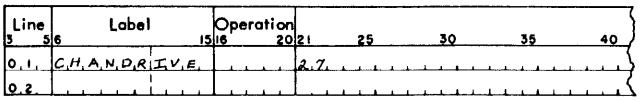

Figure 64

The entry in Figure 64 indicates that the initial tape unit described by this DTF entry is unit 7 of channel 2.

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.

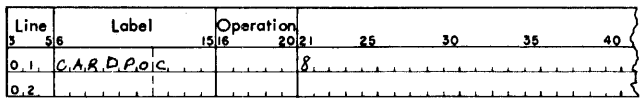

Figure 65

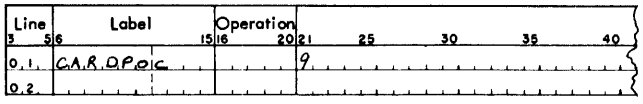

Figure 66

The entry in Figure 65 indicates that all cards from this file are to be selected into pocket 8 of the card punch. The entry in Figure 66 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.

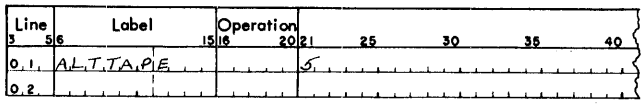

Figure 67

The entry in Figure 67 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. The operands FIXED and UNBLOCKED, referring to fixed-length and unblocked records, respectively, may be used but are not required.

RECORD FORMATS HANDLED BY THE 1410 IOCS

Records made available by the GET macro-instruction must always have a record mark or a group mark with word mark in the low-order position if the IOCS is to move them from one location in core storage to another.

Form 1 Records: These are fixed-length, unblocked records, as shown in Figure 68.

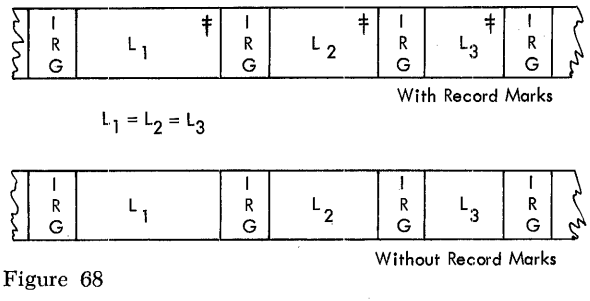

Figure 68

Form 2 Records: These are fixed-length, blocked records with padding of short-length blocks, as shown in Figure 69. If necessary, blocks consisting of fixed-length records are padded – either with the character specified in the DTF PADDING entry or with blanks if the PADDING entry was omitted. The user should insure that padding records are not mistakenly processed as data records. Padding records may be bypassed by issuing consecutive GET macro-instructions until an end-of-file indication is obtained.

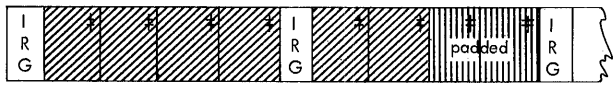

Figure 69

Form 3 Records: These are variable-length, unblocked records, as shown in Figure 70.

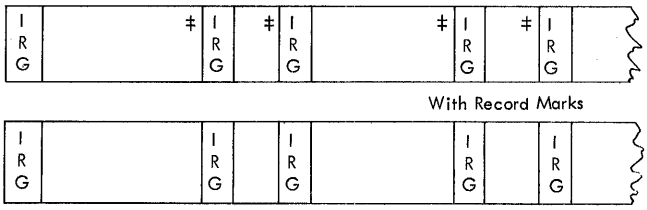

Figure 70

Form 4 Records: These records are variable-length, blocked, with a block character count field at the beginning of each block, and a record character count field in each record (see Figure 71). 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. (Terminal group marks with word marks are not included in this count.) 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.

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 (if appropriate).

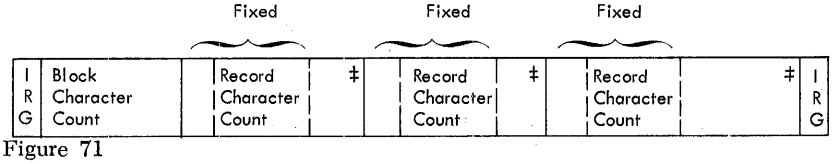

Figure 71

	Unblocked	Blocked	BLOCK		RECORD		May use indexing registers	May use work Areas
			Fixed-Length	Variable-Length	Fixed-Length	Variable-Length		
Form 1	X		--	--	X		Needed only if two I/O areas are used	
Form 2		X	X		X		Yes	Yes
Form 3	X		--	--		X	Needed only if two I/O areas are used	
Form 4		X		X		X	Yes	Yes

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

Figure 72. Summary of Record Formats

The high-order position of this field must have a word mark. The maximum size of blocks permissible is 9,999 positions. See Figure 72.

EXAMPLES OF RECFORM ENTRIES

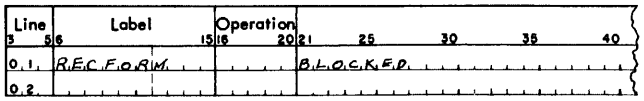

Figure 73

The entry in Figure 73 indicates that this file consists of Form 2 (i.e., fixed-length, blocked) records. The same DTF statement could have been written as shown in Figure 74.

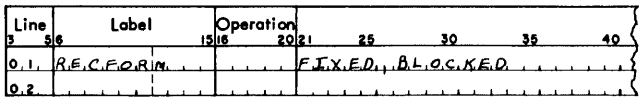

Figure 74

As indicated above, records are assumed to be 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. 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.

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.

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.

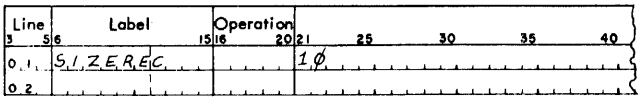

Figure 75

The entry in Figure 75 indicates that the low-order position of the character count field in each record of this file is the tenth position of the record. See also Figure 76.

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 80-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.

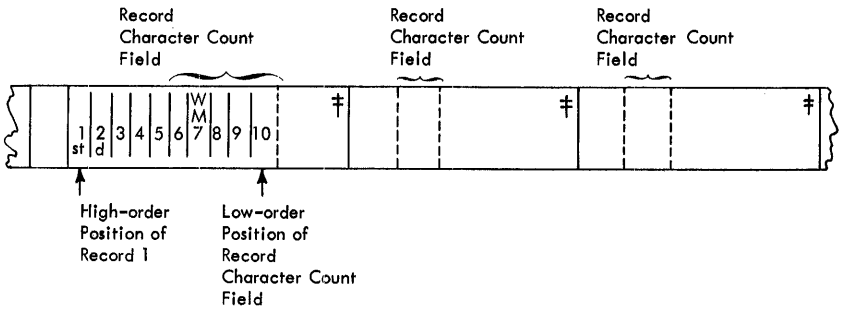

Figure 76. The Record Character Count Field

The following characters may *not* be used for padding: asterisk, tape mark, word separator character, record mark, cent sign and group mark.

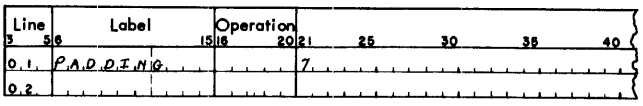

Figure 77

The entry in Figure 77 indicates that partially filled blocks are to be padded with the digit 7.

Padded records are included in the record count and padding is added to the hash total.

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

BLOCKSIZE

This entry is required for all files except those which consist of unblocked, variable-length records if checkpoint records are not to be bypassed. If checkpoint records are to be bypassed, the DTF BLOCKSIZE entry is not required for files which consist of unblocked fixed-length records. 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.

The group mark with word mark terminating the I/O area is *not* included in this count.

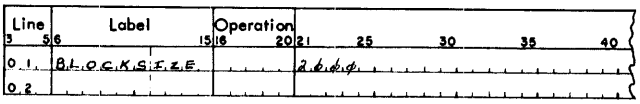

Figure 78

The entry in Figure 78 indicates that the file defined by this DTF has 2,000-character input or output area(s). (If two input or output areas are used, each has 2,000 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.

If only one input or output area is specified, the label of that area may be indexed, provided the records to be processed in this area are not blocked.

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). The 1410 IOCS does not permit unit record files to have two input/output areas.

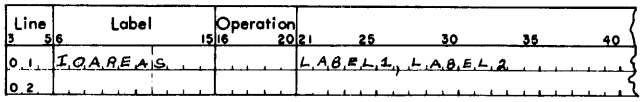

Figure 79

The entry in Figure 79 indicates that the input (or output) areas of the file described by this DTF are those defined by DA's labeled LABEL1 and LABEL2, respectively.

ADVANTAGES OF TWO I/O AREAS

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.

1. 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.

2. 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 (Figure 80). *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.

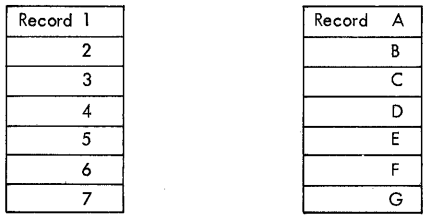

Figure 80

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.

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.

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, nor is it used if the DIOCS PRIORITY ASSEMBLE entry is used with the program.

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

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

Files for which the DTF PRIORITY entry is omitted will be assigned lowest (i.e., 9) priority. (If the DIOCS PRIORITY ASSEMBLE entry is used, the IOCS assigns file priority according to the order in which the sets of DTF entries are arranged in the source deck.)

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. If the file is a unit-record file, the GET following the GET that caused the last card of the file to be read will sense the fact that an end-of-file condition has occurred and cause a branch to be taken to the user's end-of-file routine. If the file is a tape file that does not use labels, the sensing of a tape mark will cause the IOCS to branch to the user's end-of-file routine. If the file is a tape file that uses labels, 1EOFb in a trailer label will cause the IOCS to exit to the user's end-of-file routine.

WLRADDR

This entry is not needed for output files and for variable-length, unblocked input files. The operand of the WLRADDR entry is the label of the wrong-length-record routine written by the programmer.

If this entry is omitted, a wrong-length-record check will not be made. Wrong-length-record checks will not be made for variable-length, unblocked files. In his own routine, the programmer may give any I/O commands except GET macro-instructions to the affected file.

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.

If a wrong-length-record condition occurs, the IOCS will always branch to the address represented by WLRADDR. If there are two areas used by the file, the IOCS will use a location labeled IOCSCUIOB to indicate in

which area the error condition occurred. If the error was in the first area (the area defined by the first operand of the DTF IOAREAS entry), the IOCS will set a word mark at IOCSCUIOB. If the error occurred in the second area, the IOCS will clear the word mark at IOCSCUIOB. (The cu is the channel and unit number of the tape file, unless a DTF PREFIX entry is used for the file, in which case the two letters specified by that entry replace the cu.)

If the file consists of Form 4 records, the length of the wrong-length record will be stored in an area labeled IOCSCUBLKL (where cu is the channel and unit number of the tape file). If the file uses a DTF PREFIX entry, cu is replaced by the two letters specified by that entry.

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.

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 81 and 82.

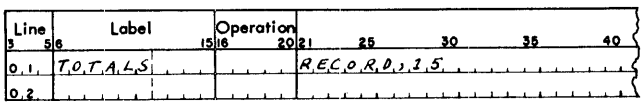

Figure 81

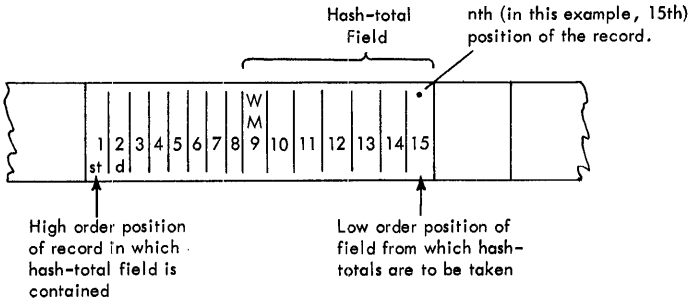

Figure 82. Hash Total Field

Record counts cannot be requested for tape files that consist of unblocked records.

The entry in Figure 81 indicates that hash total counts are to be taken from the field shown in Figure 82.

Hash total fields may contain any alphameric information, but must not be larger than ten characters. A word mark must define the high-order position of the field. Hash totals cannot be requested for unit record files.

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 nonstandard 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. See Figure 83.

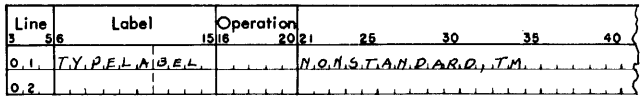

Figure 83

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.

Tape Labels

This section describes the IBM 1410 tape labels under four topics:

The purpose of tape labels
Recommended tape labeling practices
Standard IBM tape labels
Use of tape labels by the IOCS

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 IOCS can read and check tape header labels to insure that the object program uses the

proper tapes. (If no check of header labels is to be made, output header labels are not read.)

Before writing a new output file, the IOCS checks for the existence of a header flag, 1HDRb, then 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 90.

RECOMMENDED TAPE LABELING PRACTICES

Temporary Labeling: 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 84.

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 84. 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. 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. The physical label should not be removed until the tape is retired from the system.

IBM STANDARD TAPE LABELS

Standard IBM tape labels have 80 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 (Figure 85) consists of eight fields containing:

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

1HDRb	Tape Serial Number	File Serial Number	Reel Sequence Number	File Name	Creation Date (YYDDD)	Retention Cycle (-000b)	
1	6	11	16	21	31	36	41

Figure 85. The Standard Header Label

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

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 one 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, columns 1-5.
2. A five-character block count, columns 6-10.
3. A ten-character record count, (optional), columns 11-20.
4. A ten-character hash total (optional), columns 21-30 (or columns 11-20 if no record count is taken).
5. A 50-position field that is blank if not filled in by the user, columns 31-80. This field may be used in any way desired by the programmer.

FOUR FORMATS OF THE STANDARD TRAILER LABEL

1. If both record count and hash total are specified by the DIOCS COUNTS entry, the standard trailer label contains the fields shown in Figure 86.

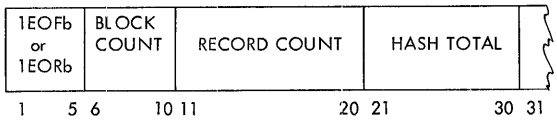

Figure 86

2. If neither record count nor hash total are specified by the DIOCS COUNTS entry, the third and fourth fields (columns 11-30) contain blanks, as shown in Figure 87.

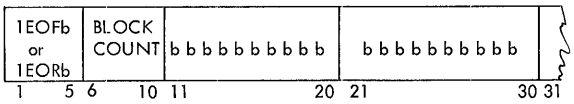

Figure 87

3. If only a hash total is specified by the DIOCS COUNTS entry, the third field (columns 11-20) contains the current hash total, beginning with all zeros. The fourth field (columns 21-30) contains blanks. See Figure 88.

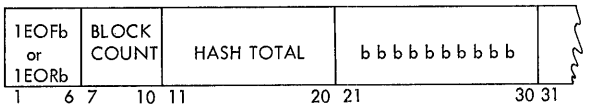

Figure 88

4. If only a record count is specified by the DIOCS COUNTS entry, the third field (columns 11-20) contains the current record count, beginning with all zeros. The fourth field (columns 21-30) contains blanks. See Figure 89.

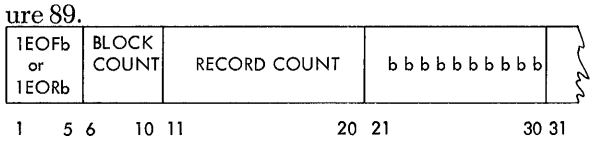

Figure 89

USE OF TAPE LABELS BY THE IOCS

The label checking and label writing functions provided by the 1410 IOCS are summarized in Figure 90.

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 for input files and work files are:

1. The ten-character file identication name.

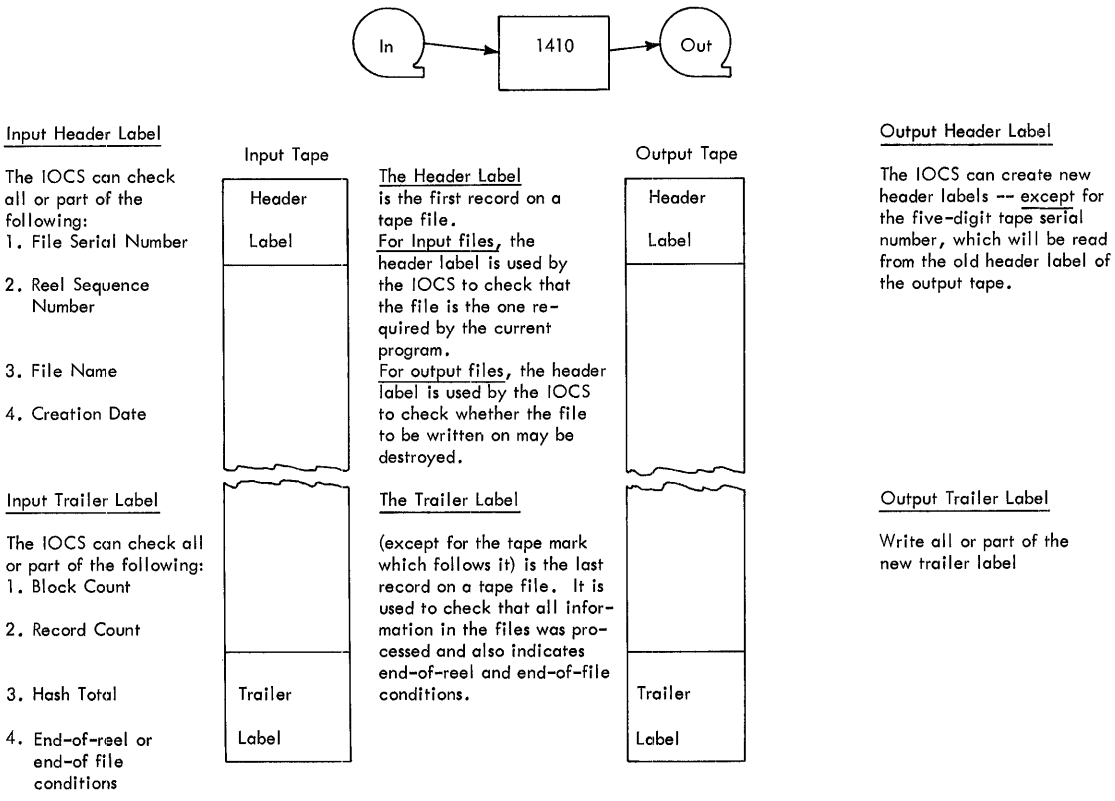

Figure 90. Use of Tape Labels by the IOCS

2. The five-digit creation date. (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. Care should be used to preserve the word mark at location 115.)

The operands of the HEADER entry for output files are:

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.

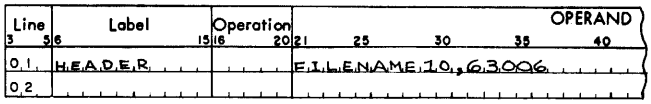

Figure 91

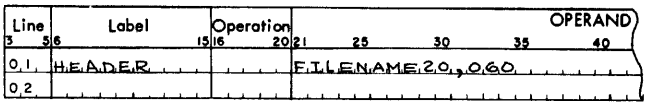

Figure 92

The entry in Figure 91 indicates that this (input) file is named FILENAME10, and was created on the sixth day of 1963. The entry in Figure 92 indicates that this (output) file is named FILENAME20, 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 (Figure 93) is

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

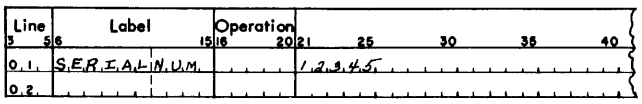

Figure 93

If this entry is present, a check is made of the file serial number if the ALL operand of the CHECKLABEL entry has been specified. If this operand has not been specified, no check of the file serial number will be made.

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.

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

REWIND

This entry is not needed if:

1. The file described by this DTF is not a tape, *or*
2. The IOCS is to rewind the tape for this file when the file is opened and closed, and when end-of-file and end-of-reel conditions occur.

The operand of this entry can be *either* UNLOAD *or* NORWD, as shown in Figures 94 and 95.

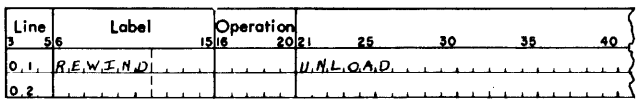

Figure 94

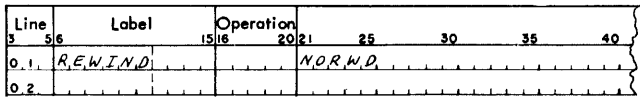

Figure 95

UNLOAD specifies that the tape for this file is to be rewound *and unloaded* when the file is closed, and when end-of-file and end-of-reel conditions occur. (This operand can be used only if the DIOCS RWDOPTION entry, with the UNLOAD operand, is included in the program.)

NORWD specifies that the tape for this file is never to be automatically rewound by the IOCS. (This operand can be used only if the DIOCS RWDOPTION entry, with *either* the UNLOAD or NORWD operand, is included in the program.)

The Eight 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.

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, 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. (For an exception, see the third paragraph under "EX6ADDR".)

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 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 whose high-order position is labeled IOCSLBAREA. (If the IOCS Independent Assembly Feature is used, IOCSLBAREA represents the low-order position of a 5-character field containing the *address* of the label area's high-order position.)

The operand of each of these DTF entries is the label of the specialized label handling routine written by the programmer. 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.

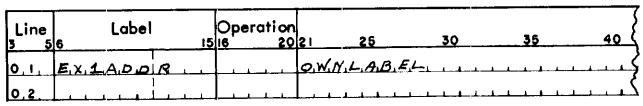

Figure 96

The entry in Figure 96 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 nonstandard 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 nonstandard 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 nonstandard input trailer labels.

This exit may be used to modify the reel sequence count and/or the DTF REWIND entry. (See Tape File Table, under "Additional Information for Programmers.")

If a tape file uses standard labels, the IOCS determines whether an end-of-reel or an end-of-file condition exists when the tape mark following the last record on a particular reel of that file is sensed. If a tape file does not use labels or uses nonstandard labels, the IOCS cannot perform this function. For this reason the programmer should: (a) specify the DTF EX6ADDR entry (for input files), (b) write a routine (whose label appears as the operand of the DTF EX6ADDR entry) which will determine whether an end-of-reel or an end-of-file condition exists, (c) write this routine so that it will cause a branch to be taken to the instruction at IOCSRENTRY if an end-of-reel condition exists, or to the first instruction of the relevant user-written end-of-file routine if an end-of-file condition exists.

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 nonstandard input header labels.

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.

EX8ADDR

This exit permits the programmer to ignore the end-of-reel reflective spot, and write additional records. In the programmer's Exit 8 routine a switch should be set and a branch executed to IOCSRENTRY. This switch should be tested in the user's main-line program following the PUT to that file. If the switch is ON, the user may include additional records on the file. The FEORL macro-instruction should then be executed and the switch reset.

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.

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

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 is the number of the channel (1 or 2), and u is the 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 100-750 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 (Figure 97).

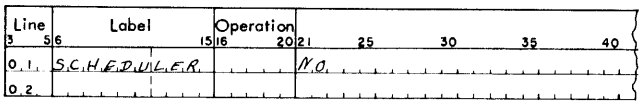

Figure 97

The DTF SCHEDULER entry may be used only for tape files.

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

The SCHEDULER entry has no effect on the way the IOCS handles end-of-reel and end-of-file conditions. They are processed just as they would be if the SCHEDULER entry was not used to define the file.

The programmer may utilize a file using the DTF SCHEDULER entry as a work file by modifying the contents of location IOCSCUTFLl, 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
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 DA's are written may be found in the bulletin, *IBM 1410 Autocoder,* Form C28-0309.) 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 DA's for all types of records which can be handled by the 1410 IOCS (both with and without indexing, whichever is applicable).

Input Areas

DA's 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 subdivisions are discussed below.

UNBLOCKED RECORDS USING ONLY ONE I/O AREA

For data read in the MOVE mode, see Figure 98. The LABEL of the DA is used to describe the area in the DTF IOAREAS entry. The 1x80 operand in Figure 98 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.

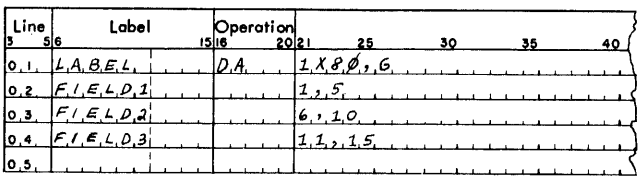

Figure 98

For data read in the LOAD mode, see Figure 99. The labels of these DA areas must not be used as operands of macro-instructions.

Line	Label	Operation	
0,1	LABEL	DA	1,X,8,Ø,,G
0,2	F,1,E,L,D,1		5
0,3	F,1,E,L,D,2		1,Ø
0,4	F,1,E,L,D,3		1,5
0,5			

Figure 99

UNBLOCKED (FIXED- OR VARIABLE-LENGTH) RECORDS USING TWO I/O AREAS

With indexing, for data read in the MOVE mode, see Figure 100.

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,X,8,Ø,,X,2,,Ø,,G
0,2	F,1,E,L,D,1		1,,5
0,3	F,1,E,L,D,2		6,,1,Ø
0,4	F,1,E,L,D,3		1,1,,1,5
0,5			
0,6	A,R,E,A,2	DA	1,X,8,Ø,,G
0,7			1,,5
0,8			6,,1,Ø
0,9			1,1,,1,5
1,0			

Figure 100

NOTE 1: The X2 operand in Figure 100 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.

With indexing, for data read in the LOAD mode, see Figure 101. The AREA1 and AREA2 labels of these DA's are the names used to describe these areas in the DTF IOAREAS entry.

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,X,8,Ø,,X,2,,Ø,,G
0,2	F,1,E,L,D,1		5
0,3	F,1,E,L,D,2		1,Ø
0,4	F,1,E,L,D,3		1,5
0,5			
0,6	A,R,E,A,2	DA	1,X,8,Ø,,G
0,7			

Figure 101

Without indexing, in either MOVE or LOAD mode (processing is done in a work area), see Figure 102.

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,X,8,Ø,,G
0,2	A,R,E,A,2	DA	1,X,8,Ø,,G
0,3			

Figure 102

BLOCKED, FIXED-LENGTH RECORDS

With indexing, for data read in the MOVE mode, see Figure 103. With indexing, for data read in the LOAD mode, see Figure 104. Without indexing, for data read in either the MOVE or LOAD mode (processing is done in a work area), see Figure 105.

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,Ø,X,8,Ø,,X,2,,Ø,,G
0,2	F,1,E,L,D,1		1,,5
0,3	F,1,E,L,D,2		6,,1,Ø
0,4	F,1,E,L,D,3		1,1,,1,5
0,5			
0,6	A,R,E,A,2	DA	1,Ø,X,8,Ø,,G
0,7			1,,5
0,8			6,,1,Ø
0,9			1,1,,1,5
1,0			

Figure 103

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,Ø,X,8,Ø,,X,2,,Ø,,G
0,2	F,1,E,L,D,1		5
0,3	F,1,E,L,D,2		1,Ø
0,4	F,1,E,L,D,3		1,5
0,5			
0,6	A,R,E,A,2	DA	1,Ø,X,8,Ø,,G
0,7			

Figure 104

Line	Label	Operation	
0,1	A,R,E,A,1	DA	1,Ø,X,8,Ø,,G
0,2	A,R,E,A,2	DA	1,Ø,X,8,Ø,,G
0,3			

Figure 105

BLOCKED, VARIABLE-LENGTH RECORDS.

With indexing, for data read in the LOAD mode, see Figure 106. (Indexing is not available for MOVE mode.) Without indexing, for data read in the MOVE mode (processing is done in a work area), see Figure 107. NOTE: The 1, 1 entries are needed by the IOCS.

Without indexing, for data read in the LOAD mode, see Figure 108.

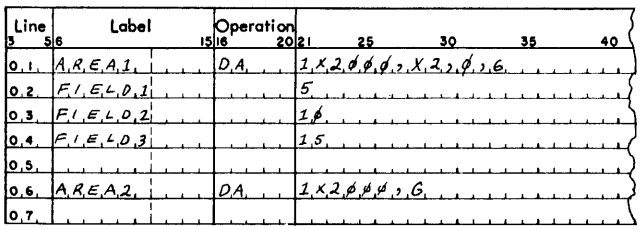

Figure 106

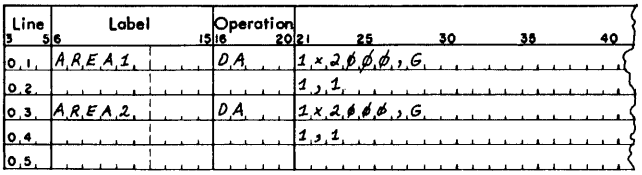

Figure 107

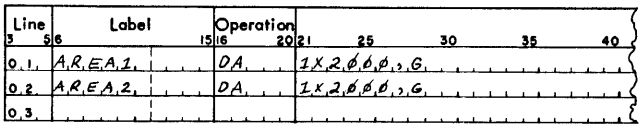

Figure 108

Ouput Areas

DA's 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 subdivisions are discussed below.

UNBLOCKED RECORDS USING ONLY ONE INPUT/OUTPUT AREA

For data written in either the MOVE or LOAD mode (the programmer can construct records in the output area), see Figure 109. The fields indicated in the DA are needed only if the programmer refers to records in the output area. Word marks and labels are needed only if the programmer does processing (or builds records) in the output area.

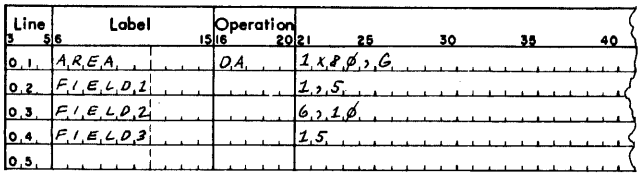

Figure 109

UNBLOCKED (FIXED- OR VARIABLE-LENGTH) RECORDS USING TWO I/O AREAS

With indexing, for data written in either the MOVE or LOAD mode (the programmer can build records in the output area), see Figure 110. The fields indicated in the DA are needed only if the programmer refers to these records in the output area. Word marks and labels are needed only if the programmer does processing in the output area or builds records in the output area.

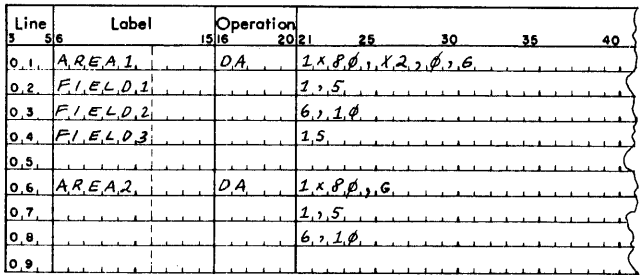

Figure 110

Without indexing, for data written in either the MOVE or LOAD mode (the programmer cannot construct records in the output area), see Figure 111. The programmer may not use any fields in the output area.

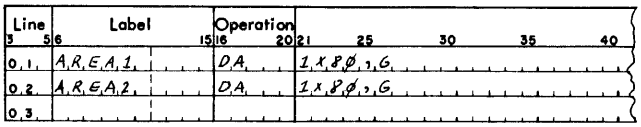

Figure 111

BLOCKED, FIXED-LENGTH RECORDS

With indexing, for data written in either the MOVE or LOAD mode (the programmer can build records in the output area), see Figure 112.

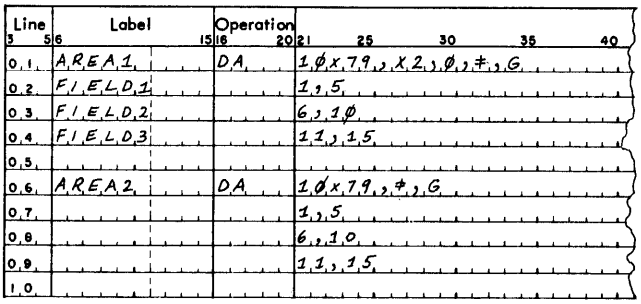

Figure 112

Without indexing, for data written in either the MOVE or LOAD mode (the programmer cannot build records in the output area), see Figure 113.

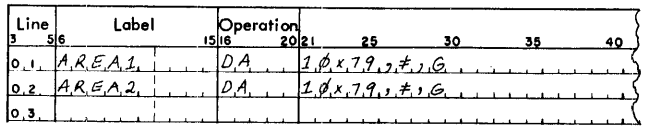

Figure 113

The records of Figures 112 and 113 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 DA's for this type of file.

BLOCKED, VARIABLE-LENGTH RECORDS, WITHOUT INDEXING

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), see Figure 114. The 1, 1 entries are needed by the IOCS.

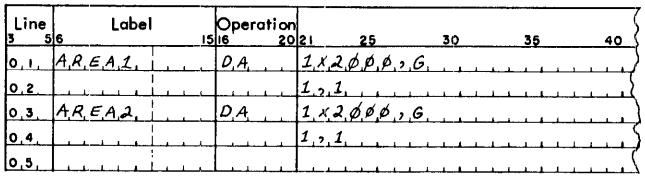

Figure 114

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. See Figure 115. The 1, 1 entry is needed by the IOCS. 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.

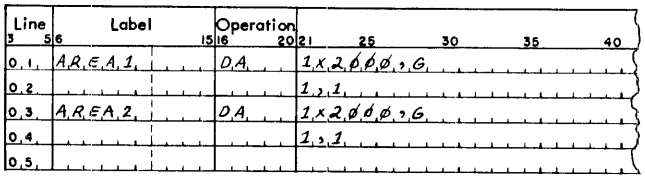

Figure 115

With the DTF VARBUILD entry, where the VARBUILD entry designates an index register: for data written in either the MOVE or LOAD mode, see Figure 116.

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. The 1, 1 entry is needed by the IOCS. The X2 entry is used as the DTF VARBUILD entry. The file does not use a DTF INDEXREG entry.

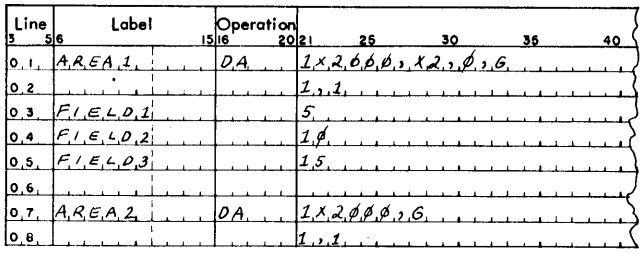

Figure 116

Work Areas

For data moved in the LOAD mode, see Figure 117. For data moved in the MOVE mode, see Figure 118. The labels of the work areas are used in the DTF WORK AREA entries and as operands of the GET TO WORK AREA macro-instructions.

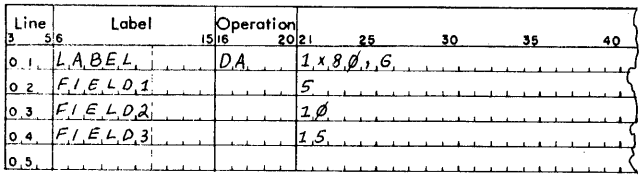

Figure 117

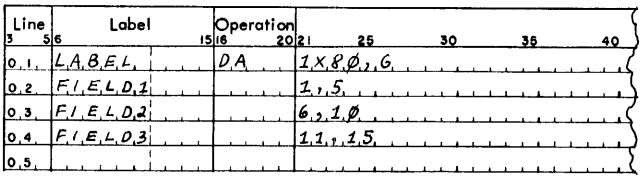

Figure 118

Additional Information for Programmers

This section 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, time required by the GET and PUT macro-instructions, and time required by the different phases of the Priority Routine.

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 the IOCS to enter a waiting loop to enable the operator either to ignore the error or to take corrective action. Read and write commands are retried 20 times on DATA CHECK indications before an error message is typed out. Read commands are retried nine times on wrong-length-record indications before an error message is typed out.

Invalid Tapes

User-written routines entered from tape label Exits 3 or 7 can be used to reject invalid tapes as follows: (a) the current header label is tested; (b) if the current tape is valid, a branch is taken to IOCSRENTRY; (c) if the current tape is not valid it is unloaded (IORWU) and a message is printed on the console typewriter (CONSL) that informs the operator an invalid tape has been mounted; (d) the address referred to by the name of the relevant file is placed in index register 15 by a zero and add instruction (ZA+FILENAME,X15) and a branch is taken to IOCSENTAB. The branch to IOCSENTAB causes the IOCS to read the header label of the tape the operator has substituted for the invalid tape.

When control is returned to the main line of the using program, index register 15 is reset (i.e., the contents of X15 at the time the branch was executed to the user's label routine are restored to X15).

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.

Size of the IOCS Routines

The amount of storage required by the IOCS routines varies considerably with the number and type of DIOCS and DTF entries written for the object program. The information below, however, can guide the user in estimating the IOCS core-storage requirements for various object programs.

WITH PROCESSING OVERLAP AND PRIORITY SPECIAL FEATURES

1. The Priority Assignment Routine: 500 positions of core storage. See the DIOCS PRIORITY entry.
2. Routines Based on the DIOCS entries:
   Maximum: 5000
   Minimum: 500

Normal requirement with standard labels: 4000
Normal requirement with nonstandard labels: 2800

3. Routines Based on the DTF entries (File Schedulers):
   Maximum: 600
   Minimum: 50

WITHOUT PROCESSING OVERLAP AND PRIORITY SPECIAL FEATURES

1. Routines Based on the DIOCS entries:
   Maximum: 4000
   Minimum: 300

Normal requirement with standard labels: 3200
Normal requirement with nonstandard labels: 2000

2. Routines Based on DTF entries (File Schedulers):
   Maximum: 250
   Minimum: 50

Operating Times for GET and PUT Macro-Instructions

The approximate operating times of GET and PUT macro-instructions for different record formats are shown in Figures 119 and 120. 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 Time in Microseconds	
Fixed-length or variable-length, blocked records	Maximum	578
Logical records remain in input area; indexing is used	Minimum	245
Fixed-length or variable length, blocked records.	Maximum	711+11.3 x L**
Logical records are moved to work area; indexing is not used	Minimum	378+11.3 x L**
Fixed-length, blocked records	Maximum	596+11.3 x L**
Logical records are moved to work area; indexing is used	Minimum	263+11.3 x L**
Variable-length, blocked records	Maximum	637+11.3 x L**
Logical records are moved to work area; indexing is used	Minimum	340+11.3 x L**

Figure 119. Operating Time in Microseconds for GET Macro-Instructions

Record Type and Handling	PUT Time in Microseconds	
Fixed-length, blocked records	Maximum	578
Records are built in output area; indexing is used	Minimum	245
Fixed-length, blocked records	Maximum	596+11.3 x L**
Records are moved to output area; indexing is used	Minimum	263+11.3 x L**
Fixed-length, blocked records	Maximum	711+11.3 x L**
Records are moved to output area; indexing is not used	Minimum	378+11.3 x L **
Variable-length, blocked records	Maximum	882
Records are built in output area; indexing is used	Minimum	549
Variable-length, blocked records	Maximum	790+11.3 x L **
Records are moved to output area; indexing is used	Minimum	457+11.3 x L **
Variable-length, blocked records	Maximum	905+11.3 x L **
Records are moved to output area; indexing is not used	Minimum	572+11.3 x L **

Figure 120. Time of PUT Macro-Instructions in Microseconds

The major (and sometimes the only) factor determining the difference between maximum and minimum times shown in Figure 120 is the time required to take the trailer record and hash total counts. The L shown in Figure 120 represents the number of characters in a logical record.

Priority Interrupt Routine Time

The approximate time in microseconds required by different phases of the Priority Interrupt Routine are indicated in Figure 121. 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 121 refer to the corresponding letters in Figure 122.

Tape File Tables

The format of an input tape file table is determined by the DIOCS entries supplied by the programmer. For example, the DIOCS COUNTS entry causes Fields 21 and 22 (see Figure 123) to be included in the table.

The file table provides a set of directions which the various IOCS subroutines interrogate. These directions control beginning-of-reel and end-of-reel procedures. Count fields and label information fields are also included. The file table may be modified during execution of the object program. However, the IOCS interrogates the table only when it is processing the OPEN, CLOSE, FEORL, and RDLIN macro-instructions or handling beginning-of-reel and end-of-reel conditions. For this reason user modications will have no effect unless an OPEN macro-instruction is again given to the file in question, or an-end-of-reel condition is in effect at the time of the modification on the file in question.

The file table shown contains the maximum possible number of fields. The minimum possible table would contain Fields 2, 3, 4, 5, 6, 7, 15 and 20.

The $ shown in Figure 123 represents IOCS; c represents channel; u represents unit.

Field 1: This field contains the operand of the DTF PRIORITY entry, or the number 9 if the entry was omitted. This field will not appear unless the DIOCS FEATURES entry containing the OVERLAP and PRIORITY operands was supplied.

Field 2: This field contains the number 1 if one I/O area was specified in the operand of the DTF IOAREAS entry, or the number 2 if two areas were specified.

Field 3: This field contains @0b@ if the file is opened; it contains @bb@ if the file is closed. This field is used by the IOCS to determine if the tape must be repositioned if the program is restarted from a checkpoint.

Field 4: This field contains the Op Code and X-Control field of the instruction required to read the base tape. If @ or * appear in the hundreds position of the X-Control field, the DIOCS FEATURES entry containing the OVERLAP and PRIORITY operands must be supplied; if % or □ appears, these operands must be omitted. It also contains the Op Code of the Branch If Any I/O Channel

Status Indicator On instruction used to check for errors after execution of the I/O instruction used to read the base tape.

Field 5: This field contains the same information as Field 4, except that the information in this field is for the *alternate* tape.

Field 6: This field contains the address of the file initialization sequence.

Field 7: This field contains the programmer's end-of-file routine address for input files, the padding routine address for fixed, blocked output files, or the address of a location within the IOCS end-of-reel routine for all other output files.

Field 8: This field is missing unless the DIOCS LABELDEF operand was specified with either the MIXED or the STANDARD operand.

Phase of Priority Interrupt Routine					Time (Without Read or Write Errors)	
No.	FROM	TO	Conditions	Line of Coding in Fig. 122	Maximum	Minimum
1	Interrupt of Main Routine (A)	Return to Main Routine (F)	5 Files (See NOTE). No input/output operation pending	A-B-D-D'-D-E-F	1050	820
2	Interrupt of Main Routine (A)	Start of execution of Input/Output Command (C')	5 Files; 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

NOTE: Add or subtract 50 microseconds for each file difference from five. Applies to Phase 1 only.

Figure 121. Approximate Time in Microseconds for Different Phases of the Priority Interrupt Routine

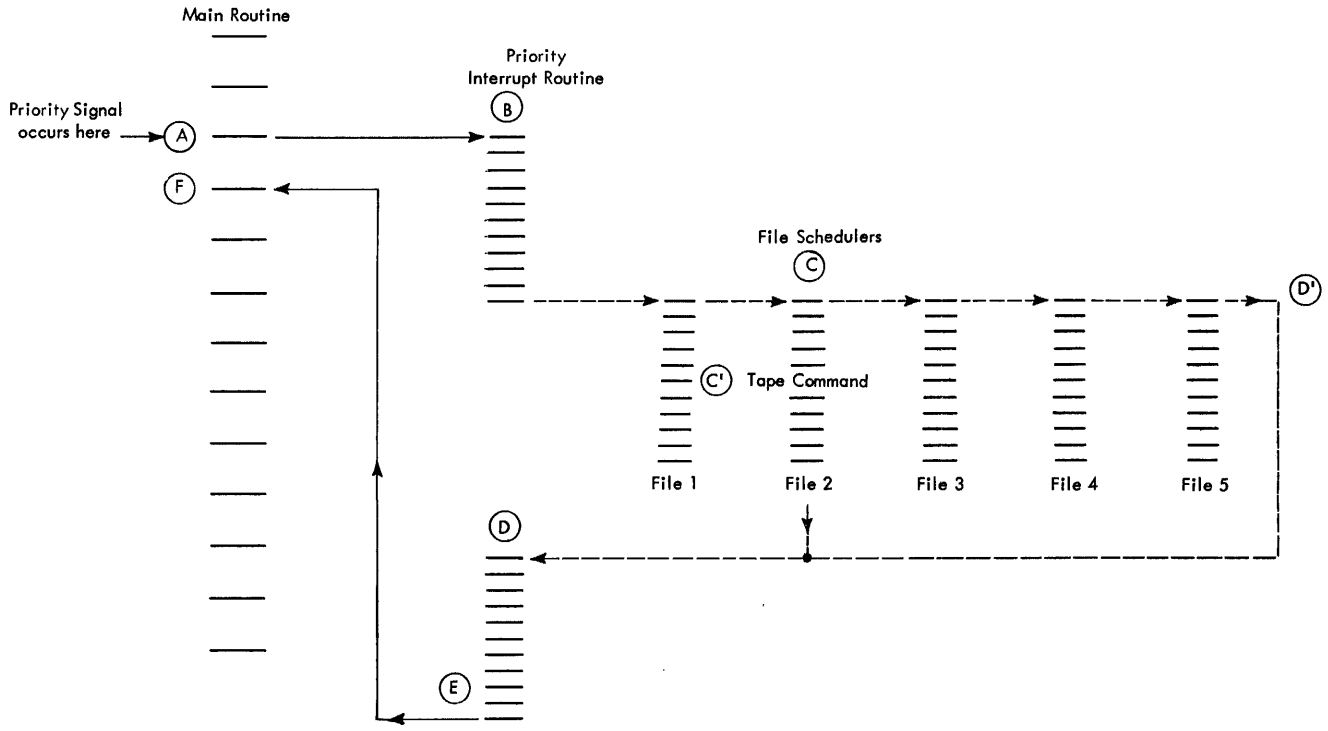

Figure 122. Approximate Time Required by Different Phases of the Priority Interrupt Routine

Field	Label	Mnemonic	Operand
1		DCW	@9@
2	$cuACT	DCW	@2@
3		DCW	@bb@
4	$cuBASE	DCW	@M@U3R@
5		DCW	@M@U4R@
6		DCW	$cuINIT
7	$cuEOF	DCW	FILEOFADDR
8	$cuFSCK	DCW	@0@
9	$cuHFS	DCW	@12345-@
10		DCW	@016b@
11		DC	@ALPHAXNAME@
12		DC	@63365@
13		DC	@-030b@
14	$cuTFLB	DCW	@0@
15	$cuTFL1	DCW	@1@
16	$cuTFL2	DCW	@1@
17	$cuTFL3	DCW	@2@
18	$cuTFL4	DCW	@0@
19	$cuTFL5	DCW	@0@
20	$cuTBC	DCW	@bbbbb@
21	$cuTRC	DCW	@bbbbbbbbbb@
22	$cuTHT	DCW	@bbbbbbbbbbbbbbbb@
(INPUT)			(OUTPUT)
23	$cuD6	DCW	@1@ $cuD1 DCW @1@
24		DCW	EXIT6ADDR DCW EXIT1ADDR
25	$cuD7	DCW	@1@ $cuD2 DCW @1@
26		DCW	EXIT7ADDR DCW EXIT2ADDR
27			$cuD3 DCW @1@
28			DCW EXIT3ADDR
29			$cuD4 DCW @1@
30			DCW EXIT4ADDR
31			$cuD5 DCW @1@
32			DCW EXIT5ADDR
33			$cuD8 DCW @1@
34			DCW EXIT8ADDR

● Figure 123. Input Tape File, File Table

For input files this field contains 0 if the operand of the DTF CHECKLABEL entry supplied was ALL. This will cause the input header label routine to compare the header file serial number with the contents of field 9. This field contains 1 if the ALL operand of the DTF CHECKLABEL entry was omitted. In this case the input header routine will not compare the header file serial number with the contents of field 9.

For output files this field contains 1 if the DTF SERIALNUM entry was omitted. This will cause the input header routine to place the tape serial number in field 9 when the file is opened, but before the new header label is written. This field contains 0 if the DTF SERIALNUM entry was supplied. In this case the reel sequence number is not placed in Field 9.

Fields 9 through 13: These fields are missing if field 8 is missing. These fields contain the label information shown in Figure 85, positions 11 through 40. The IOCS uses these fields to prepare output header labels and to check input header labels. The DTF HEADER and SERIALNUM entries supply this information. If they are omitted, these fields contain blanks.

Field 14: This field is missing if field 8 is missing. This field contains 0 if the STANDARD operand of the DTF TYPELABEL entry was supplied, 1 if this entry was omitted, or 2 if the NONSTANDARD operand was supplied.

Field 15: This field contains 0 if the DTF FILETYPE entry specified an output file; 1, if an input file.

Field 16: This field does not appear if the DIOCS ALTDRIVE entry has been omitted. It contains 1 if the DTF ALTTAPE entry has been specified. It contains 0 if the DTF ALTTAPE entry has been omitted.

Field 17: This field is missing if field 8 is missing. Furthermore, this field will not appear unless either the DTF HEADER and SERIALNUM entries or the DIOCS LABELDEF entry with the IDENT operand was supplied. This field contains 0 if the DTF CHECKLABEL entry with the ALL operand was supplied; 1, if this entry was omitted; 2, if the IDENT operand was supplied.

Field 18: This field is missing if the TM operand of the DIOCS LABELDEF entry is omitted. This field contains 1 if the TM operand of the DTF TYPELABEL entry was supplied, or 0 if the TM operand was omitted.

Field 19: This field does not appear unless the DIOCS RWDOPTION entry was made. This field contains 0 if the NORWD operand of the DTF REWIND entry was supplied, 1 if the DTF REWIND entry was omitted; 2 if the UNLOAD operand was supplied.

Field 20: This field contains the number of blocks made available to the user by means of the GET macro-instruction. Input blocks bypassed by the user's WLR routine are not included in this count.

Field 21: This field contains the number of records made available by the GET macro-instruction or processed by the PUT macro-instruction. This field appears if the RECORD operand of the DIOCS COUNTS entry was supplied.

Field 22: The hash total is accumulated in this field. This field appears if the HASH operand of the DIOCS COUNTS entry was supplied.

Fields 23, 25: For input files, these fields contain 1 if the DTF EXIT6ADDR and EXIT7ADDR entries were supplied, or 0 if they were omitted. These fields appear if operands 6 and 7 of the DIOCS EXITS entry were supplied.

For output files, these fields contain 1 if the DTF EXIT1ADDR *and* EXIT2ADDR entries were supplied; 0 if they were omitted. These fields appear if operands 1 and 2 of the DIOCS EXITS entry were supplied.

Fields 24, 26: For input files, these fields appear if operands 6 and 7 of the DIOCS EXITS entry were supplied.

For output files, these fields appear if operands 1 and 2 of the DIOCS EXITS entry were supplied.

Fields 27, 29, 31, 33: These fields contain 1 if the DTF EXIT3ADDR, EXIT4ADDR, EXIT5ADDR, and EXIT8ADDR entries were supplied, or 0 if they were omitted. These

fields appear if operands 3, 4, 5, and 8 of the DIOCS EXITS entry were supplied.

Fields 28, 30, 32, 34: These fields appear if operands 3, 4, 5, and 8 of the DIOCS EXITS entry were supplied.

IOCS Diagnostic Messages

During assembly, the IOCS may generate diagnostic messages for errors in the use of the IOCS coding entries. These messages are printed on the assembly listing and on the console typewriter. (Thus, installations employing an assemble-and-execute sequence of operation can save machine time by not attempting to execute immediately an object program that will not run.)

The format of the IOCS diagnostic messages is as follows:

nnn.MACRO.xxx

nnn is the identification number of the message; MACRO is the name of the macro-instruction in which the coding error was made; and xxx are code letters that indicate the type of error, as follows:

W (Warning) The program *may* run as assembled.

E (Error) The Program *cannot* run as assembled.

C (Comment) The program *can* run as assembled. (The IOCS writes an explanatory comment when this code letter is given.)

One of these letters is always given as the first letter of xxx. Other code letters of xxx further define the mistake that was made in coding the macro-instruction, as follows:

M Operand or specification missing.

U Unrecognizable or undefined operand.

A Assumed. (The IOCS has assumed, on the basis of the DIOCS and DTF entries, the intent of the programmer, and the assembly has proceeded on that assumption.)

D Deleted. (The IOCS has not generated any routines or linkages for the macro-instruction.)

P The object deck can probably be "patched" (with the instruction listed by the IOCS whenever it gives a message containing this code letter).

O Overcall. (The IOCS correlates the DIOCS, DTF, and macro-instruction entries. The overcall code letter is given whenever one of these entries is either redundant or unnecessary. For example, if a DIOCS LABELDEF entry is written for the program but there are no DTF entries for tape files, the IOCS assumes that LABELDEF is an overcall.)

User-Coded 1402 and 1403 Input/Output Instructions

The user may code 1402 Card Read Punch and 1403 Printer input/output instructions in Autocoder, provided the conventions shown in Figure 124 are observed.

Line	Label	Operation	
0.1		IOSYS	PAUSE, 1
0.2		R	0, READINAREA
0.3		BAL	IOCSUREERR
0.4		BEF 1	EOFADDR
0.5		IOSYS	RESUME
0.6			

Figure 124.

Any card read punch or printer I/O instruction coded in Autocoder may be substituted for the instruction shown on line 2.

BEF is only used following card read operations; however, it may be omitted. If it is omitted, and if an end-of-file condition is sensed by the unit-record error routine, the IOCS prints the message "10100 NR (I/O OP)", issues a read-a-card instruction, and enters a waiting loop. The waiting loop is terminated by a successful card read operation.

BEF need not be addressed to EOFADDR; it may, for example, be addressed to the next sequential instruction.

IOSYS PAUSE and RESUME are not necessary unless read/write and processing operations are overlapped on channel 1.

Index

Printed in U.S.A. C28-0334-1

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, New York