File Number 7070-30
Form C28-6175-2

IBM Systems Reference Library

IBM 7070-Series Input/Output Control System for IBM 729 Magnetic Tape Units Unit Record Equipment (7070-IO-904)

The Input/Output Control System (IOCS) described in this publication was designed for users of IBM 7070-Series Data Processing Systems with IBM 729 Magnetic Tape Units and unit record equipment. Part I of this publication is devoted to a general explanation of IOCS. Part II describes the method of supplying IOCS with variables, and explains the provisions made for tape errors and SPOOL programs. Part III presents recommendations for writing and assembling programs using IOCS.

MAJOR REVISION (December, 1963)

This publication is a major revision of the reference manual IBM 7070 Series Input/Output Control System, Form C28-6175-1, which is now obsolete. The present edition should be reviewed in its entirety.

The information in the technical newsletter N28-1104 is included in this publication.

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. D91, PO Box 390, Poughkeepsie, N. Y.

PURPOSE OF THIS PUBLICATION

This publication describes the Input/Output Control System (IOCS) designed for users of IBM 7070-Series Data Processing Systems with IBM 729 Magnetic Tape Units and unit record equipment.

PURPOSE OF IOCS

IOCS provides users with routines for reading and writing card and tape records. Since input and output routines constitute nearly half of the average program, use of IOCS results in substantial savings in the cost of program writing and testing, and a major reduction in programming time and effort. In addition, IOCS produces input/output routines that are efficient, free from programming errors, and standardized.

Tape Records

IOCS macro-instructions enable the programmer to process data records that are to be written on tape or read from tape. The reading and writing of tape records is controlled by IOCS and occurs simultaneously with processing. Blocking of output data records and deblocking of input data records are handled automatically by the Input/Output Control System.

IOCS includes a tape reel control system that checks the mounting of magnetic tapes for each run of the program and aids in tape library maintenance.

Unit Records

IOCS macro-instructions enable the user to cause unit records to be read, punched, and printed on on-line equipment.

Additional Functions of the IOCS

IOCS includes checkpoint and restart routines that make it possible to interrupt a program at any point, and to continue from that point at another time.

IOCS allows the running of SPOOL (Simultaneous Peripheral Operations On-Line) programs during the main program. See IBM 7070 SPOOL System, Form J28-6047, for information on SPOOL programs.

IOCS contains error routines for tape records and unit records. The error routines correct errors automatically whenever it is possible to do so.

The functions provided by IOCS are incorporated into the user's program during assembly by Autocoder. If Four-Tape Autocoder is used to assemble a program, precautions regarding the use of certain macro-instructions must be observed (see "Use of IOCS with Four-Tape Autocoder" for details of these precautions).

PREREQUISITES

The reader should be familiar with one of the 7070-Series Autocoder programming systems. These are described in the following publications:

IBM 7070/7074 Four-Tape Autocoder, Form C28-6102

IBM 7070 Series Programming Systems, Autocoder, Form C28-6121

A knowledge of the following publication is also required:

IBM 7070-7074 Data Processing Systems, Form A22-7003

MACHINE REQUIREMENTS

Programs using IOCS are assembled by the Autocoder Processor, which requires the following machine configuration:

1. IBM 7070-Series Data Processing System
2. Six to ten IBM 729 Magnetic Tape Units
3. IBM 7500 Card Reader or IBM 7501 Console Card Reader, if any input is in punched card form
4. IBM 7550 Card Punch, if on-line punching of output is desired
5. IBM 7400 Printer, if on-line printing of output is desired

USE OF 729 IOCS AND 7340 IOCS

A 7074 Program that will run using both IBM 7340 Hypertape Drives and IBM 729 Magnetic Tape Units requires the use of both 7340 IOCS and 729 IOCS. The 7074 user who plans to incorporate both IOCSs into his program should be familiar with the publication IBM 7074 Input/Output Control System for IBM 7340 Hypertape Drives, Form C28-6315.

ORGANIZATION OF THIS PUBLICATION

This publication is divided into three parts. Part I discusses the factors governing the programmer's choice of processing and record-handling methods when using IOCS. Part II describes the method of supplying IOCS with variables, and explains the provisions made for tape errors and SPOOL programs. Part III presents recommendations for writing and assembling programs using IOCS. An appendix is included, in which the theory of scheduler operation is presented.

PART I: GENERAL PROGRAMMING METHODS

To aid the programmer in understanding and using the IBM 7070-Series Input/Output Control System for 729 tape units and unit record equipment, the first part of this publication discusses the factors governing the choice of record blocking, record areas, processing methods and record format. An understanding of these factors will enable the programmer to write an efficient program using all the features provided through IOCS.

SYSTEM ELEMENTS

IOCS is incorporated into a user's program in several different but closely integrated sections; these sections are produced during Autocoder assembly based on information supplied by the programmer.

File Specifications

Each file used in a program must be described by a set of File Specifications. This is done through the use of a DTF (Define Tape File) statement and its subsequent entries. The DTF defines a number of parameters related to the file and its records. In conjunction with the File Specifications, the programmer must furnish one or more Autocoder DA entries for the tape record area(s) and an Autocoder DC entry for tape label information.

File Scheduler

A File Scheduler that controls the reading or writing of the file is produced for each file during Autocoder assembly. The precise arrangement of the File Scheduler is based on parameters supplied by the programmer in the DTF statement (see "File Specifications").

Channel Scheduler

A Channel Scheduler is used to regulate the use of one tape channel by one or more File Schedulers. During Autocoder assembly, a Channel Scheduler is produced for each tape channel used in the program.

Subroutines

A number of subroutines that are used for the entire program are incorporated into the program during the Autocoder assembly. The most important of these subroutines are:

1. The OPEN subroutine that is used by a macro-instruction to prepare the first reel of a tape file for processing.
2. The CLOSE subroutine that is used by a macro-instruction to make a tape file unavailable for processing.
3. An end-of-reel subroutine that is used to change from one reel of a file to the next. It is also used to complete the processing of the last reel of a file and then remove it from processing.
4. An error subroutine that is used to detect and correct tape errors. This subroutine is also used to handle any of the unusual condition codes that may occur during the running of the program.
5. A checkpoint subroutine that enables the user to record the contents of core storage at a specific moment, to permit restarting the program later at the same point.
6. A restart subroutine that allows the user to interrupt a program at any time and to continue it again at another time.

Each of the subroutines mentioned above is entered, directly or indirectly, by means of a macro-instruction that causes the generation of a linkage to the subroutine.

PRINCIPAL MACRO-INSTRUCTIONS

The principal macro-instructions provided by the IOCS are OPEN, GET, PUT, and CLOSE. All of the macro-instructions are described in detail later in this publication, but a brief description of the four principal macro-instructions is given here to assist in understanding the use of IOCS.

OPEN Macro-Instruction

The OPEN macro-instruction prepares input and output files for processing. To prepare a file for use, the OPEN macro-instruction:

1. Initializes other IOCS routines for processing the file.
2. Tests to see that the first reel of the file is ready on the specified tape unit.
3. Processes the header label of the first reel of a file.

GET Macro-Instruction

The GET macro-instruction makes one data record available for processing; it may also be used to move the data record to a work area. Many GET macro-instructions may refer to the same file; each GET macro-instruction obtains the next sequential record in the file.

Deblocking of input record blocks is performed by the GET macro-instruction. It also indicates to the File Scheduler that all the records in an input area have been processed and that the area is ready for the next block of input records.

After the last record in an input file has been processed, the next GET results in a branch to an end-of-file subroutine rather than an attempt to obtain another input record.

PUT and PUTX Macro-Instructions

The PUT and PUTX macro-instructions are used to cause records to be included in an output file.

One form of the PUT macro-instruction moves one record from a processing area to the next available space in an output area, and causes the record in the preceding space in the output area to be included in a block of records waiting to be written.

A second form of the PUT macro-instruction is used when processing is to be done in an output area. It causes the record in the preceding space in the output area to be included in a block of records waiting to be written. It also changes the index words for the output file to define the next space in the output area.

The PUTX macro-instruction exchanges input area RDWs with output area RDWs so that records from input files can be included in output files without actually moving the records.

Many PUT and PUTX macro-instructions may refer to the same output file; each one causes a record to be included in the file.

A function of the PUT and PUTX macro-instructions is to form the output records into blocks. These macro-instructions also indicate to the File Scheduler that all spaces in an output area have been filled by processed records, and that the block is ready to be written on tape.

CLOSE Macro-Instruction

The CLOSE macro-instruction makes the input and output files named in its operand unavailable for processing. It is normally used when all input files in a program have reached end of file. When executed, the CLOSE macro-instruction:

1. Writes any output records that may still be in output areas.
2. Places the files in inactive status to make them unavailable for processing.

The CLOSE macro-instruction allows the user to specify which files are to be made unavailable. (The END macro-instruction can be used to perform similar operations on all active files.)

Examples of Macro-Instruction Use

The following simplified programs illustrate the use of the principal macro-instructions. These examples show only the macro-instructions; since the instructions for processing the records are irrelevant, they are represented only by dots.

Example 1. A program for reading one input file, updating each record, and writing an output file can be written as follows:

```
Sequence (Pg)(Lin)   Name (Label)   Operation Code   OPERAND                          Basic Autocoder
01                                                    INITIALIZE
02
03                                   OPEN   INPUTFILE,OUTPUTFILE
04
05
06                   INGET           GET    INPUTFILE                                 SEE NOTE A
07
08
09                                   PUT    INPUTFILE IN OUTPUTFILE                   SEE NOTE B
10
11                                   B      INGET
12                   ENDOFJOB        CLOSE  INPUTFILE,OUTPUTFILE
13
14                                   HP                                               END OF PROGRAM
15
```

NOTE A: One input record is made available for processing. If all the records in the input file have been processed, a branch to ENDOFJOB will occur.

NOTE B: The record made available by the GET macro-instruction and processed in the input area is moved to the output area.

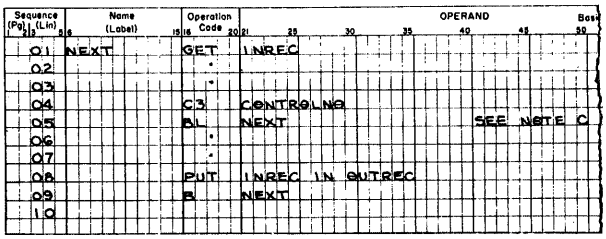
Example 2. A program for reading one input file, updating each record, and placing it in an output file by exchanging RDWs can be written as follows:

```
Sequence (Pg)(Lin)   Name (Label)   Operation Code   OPERAND
01                                   GET    INREC
02
03
04                                   PUTX   INREC IN OUTREC
05
```

Example 3. A program for reading one input file, moving each record to a work area for processing and writing an output file can be written as follows:

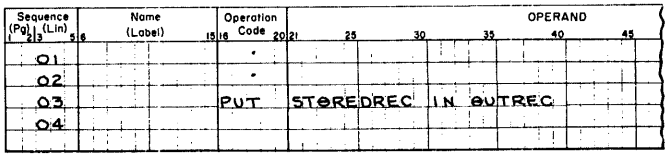
```
Sequence (Pg)(Lin)   Name (Label)   Operation Code   OPERAND
01                                   GET    INREC TO WORKAREA
02
03
04                                   PUT    WORKAREA IN OUTREC
05
```

Example 4. It may be desirable to delete a record during the reading, updating, and writing of a file. This can be accomplished by ignoring the record when executing a PUT or PUTX macro-instruction. A program involving deletions can be written as follows:



NOTE C: In this example, the record is deleted whenever the comparison (Line 04) turns on the LOW indicator. Any method may be used to determine which records to delete.

Example 5. Records that do not appear in an input file may be assembled in storage for inclusion in an output file. For example, heading lines or total lines for a report could be assembled in storage and inserted in an output file when necessary. A program for this purpose can be written as follows:



TAPE RECORD BLOCKING

Tape reading and writing time and the amount of tape used should be considered when planning a program using magnetic tape input and/or output; it is usually desirable to keep these items to a minimum. By writing records in blocks (i.e., two or more data records not separated by inter-record gaps), reading and writing time per data record and the amount of tape used can be reduced. The number of data records in one block is called the "blocking factor."

Reading and Writing Time

The manner by which blocking reduces the reading and writing time is easier to understand through a comparison of blocked and unblocked records. For comparison purposes, assume that 1.0 millisecond is required to process one data record of a file in a program that reads only one tape file from an IBM 729 IV Magnetic Tape Unit; each data record has 50 tape characters written at a density of 556 characters per inch. The various times for unblocked records and for blocks of 40 data records are as shown below:

Tape Operation	Time per Data Record in Milliseconds	
	Unblocked	40 Data Records per block
Read Record	0.8	0.8
Start/Stop	7.3	0.2*
Total	8.1	1.0

*Approximate quotient of 7.3 divided by 40

When the time to process one data record (1.0) is compared with the total time for the tape operations (8.1), it can be seen that about 87% (8.1/8.1-1.0) of the total time is spent waiting for a tape operation to be completed. Such a program is said to be "tape-limited." The table shows that by writing 40 data records in a block, the total tape operation time per data record can be reduced to 1.0 millisecond. The reduction is possible because the tape must be started and stopped only once for the block of 40 data records, rather than once for each unblocked data record.

From the example, it would seem that the highest possible blocking factor should be used, but very high blocking factors may create other problems. Extremely high blocking factors may require too many storage locations, thereby limiting the space needed by the program. Although the tape operation time per data record can be reduced to a very small value by using a very high blocking factor, the running time of the program may remain the same. In the example given above, the tape operating time could be reduced below 1.0 millisecond by using a blocking factor greater than 40, but the running time of the program would remain 1.0 millisecond per data record; when the processing time determines the running time, the program is said to be "process-limited."

The blocking factor used should be a compromise between tape operation time and storage required. A blocking factor that produces a tape record of from 2,000 to 4,000 characters is suggested as a suitable compromise. If tapes are to be prepared on, or used with, an IBM 1401 Data Processing System, a blocking factor to produce tape records of from 1,000 to 2,000 characters may be more suitable.

Amount of Tape Used

Blocking data records on tape reduces the amount of tape used by reducing the number of inter-record gaps. If each tape record contains 50 characters

written at a density of 556 characters per inch, about 33,000 records can be written on one reel of tape in unblocked form. About 254,000 of these records can be written on one reel of tape if a blocking factor of 40 is used. Blocking records reduces the number of reels of tape required for a long file. An added advantage of using fewer reels of tape for a file is that the operator has fewer mounting and unmounting operations to perform; fewer manual operations reduce the likelihood of errors in tape handling.

Processing Blocked Records

The blocking and deblocking of records is handled entirely by IOCS, and does not complicate the programmer's job. The records are made available in sequence, so the programmer may write the program as though each data record were a separate tape record. In conjunction with blocking and deblocking data records, the reading and writing of tape records are controlled by IOCS, so that tape operations occur simultaneously with processing.

INPUT/OUTPUT AREAS

In order to read or write a block of records, a storage area must be provided to contain the block after it is read or before it is written. The storage area must be large enough to contain a complete tape block.

IOCS allows the user to specify one, two, or three storage areas for each tape file. This means that one, two, or three tape blocks from the corresponding file may be contained in storage at the same time. The number of areas used for each file can determine the efficiency of the program. When selecting the number of areas to be used for a file, the programmer must consider:

1. The amount of storage that will be taken up by one, two, and three tape blocks. The storage locations needed for other portions of the program, i.e., instructions, constants, etc., may limit the number of areas that can be specified.
2. The number of tape files in the program. If there are many files used in the program, the number of areas specified for some files may have to be limited to conserve storage locations.
3. The activity of the file. When the storage locations available for areas are limited, the most active files should be given preference for multiple areas.
4. Whether the program will be tape-limited or process-limited. If the program is tape-limited, multiple areas may help to reduce the running time of the program.

The various numbers of areas that may be used with IOCS are described separately below, to aid the user in selecting the most suitable number for each file.

One Area

When one area is specified, the program must pause for a tape operation (read for input files, write for output files) after the processing of each block of records. The pause follows the processing of the last record of a block. Before the first record of the next block can be processed, the next input block must be read, or the output block just completed must be written. The time taken by the tape operation will be equal to the start time (10.8ms. for a 729 II or 7.3ms. for a 729 IV) plus the time required to read or write the entire block of records.

It can be seen that processing is suspended during the tape operation when only one area is used for a file; multiple areas allow processing to overlap tape operations. Therefore, one area should be specified only when storage locations are limited by other program requirements. One area may be suitable for files that have only a few records compared to other files used in the program. In this case, and for low-activity files, the occasional pauses for tape operations have little or no effect on the total running time of the program.

Two Areas

The pause for tape operations can be eliminated by using two areas for a file. After the last record in a block has been processed, the program can advance to the first record in the next block, which will be in the other record area. While processing continues in the second area, the first area can be used for a tape read or write operation. The two areas are used alternately for processing and tape operations. For input files, this action obtains the next block of records before the program calls for it; for output files, this action enters the last block of records into the file while the program is producing the next block.

Considering the advantages of using two areas rather than one, the use of two areas will probably become the normal procedure; the use of one area will be confined to programs that impose restrictions as explained under "One Area."

Three Areas for One File

Three areas may be specified so that three blocks of records from a file can be contained in storage at the same time. The use of three areas for one file is

comparable to the use of two areas as described above, except that another block of records is available. For input files, two blocks of records can be read into storage while the program is processing another block; for output files, the program may prepare two blocks of records while the other is being written on tape.

Under most circumstances, the use of three areas has no timing advantage over the use of two areas.

Three Areas for Two Files

When an input file is to be read, updated, and written in an output file, the "normal" procedure is to use two areas for each file as described above. The need for four areas for this type of processing can be reduced to three under certain conditions. If no records are to be added or deleted during the processing of the files, IOCS allows the use of only three areas for both files.

When two files are used with three areas, each area is used in rotating sequence. This means that each area functions successively as an input area, as a processing area and then as an output area. For example, when the first area is being written into the output file, processing is being done in the second area and the input file is being read into the third area; after processing in the second area is completed, it will be used for writing into the output file, and processing will then take place in the third area while the first area is used for reading from the input file.

Three-Area Rotating System

When an input file and an output file share three tape record areas, as indicated by the digit 3 in Line 16 (IOMETHOD) of the File Specifications (see "File Specifications Entries") for the two files, no PUT macro-instructions that reference the output file should be used. The writing of tape blocks on the output file is controlled entirely by the use of GET macro-instructions referencing the input file. As the GET routine obtains the first record of a new input block for processing, it also causes the previous input block (all records of which have been processed) to be written.

Processing Using Multiple Areas

The use of more than one area for a file does not complicate the programmer's job. All functions governing the use of multiple areas are performed by the IOCS and the programmer may write his program as though only one area were being used.

RECORD PROCESSING

A very common data-processing operation requires that data be read from an input file, processed, and then written in an output file. Examples of this type of operation are file maintenance (read a master tape, update the records, write a new master tape) and preparation of reports (read a tape, edit data into proper form, write a tape for printing later). The macro-instructions provided by IOCS allow two methods of record processing; one method moves the records, the second exchanges the RDWs that define the records.

Record Move

The most obvious way of placing an input record in an output file is to move the record from the input area to the output area. This can be accomplished through the use of the PUT macro-instruction, which moves data by the use of the block transmission feature of the IBM 7070. When records are moved from an input file to an output file, processing must be done in the input area before the record is moved. To facilitate processing in the input area, the GET macro-instruction, which provides the input record, places the address of the first word of the record in the indexing portion of an index word. The record may be processed by instructions that consider the first word of the record as relative to 0000, and that are indexed by that index word.

When all records contained in one input area have been moved to an output area(s), IOCS will read a new block of input records into the area. After an output area has been filled by moving records into it, IOCS will write the contents of the area on tape; upon completion of the tape write operation, the area will be availabe to receive more records.

RDW Exchange

Another, and frequently superior, method of record processing made possible by IOCS exchanges the Record Definition Words that define the records in the input and output areas. For most record forms, reading and writing are controlled by lists of RDWs; this allows a record to be "sent" from an input area to an output area by exchanging RDWs in the lists. Through the use of the PUTX macro-instruction, the RDW that defines the input record is placed in a list that will be used for writing, while the RDW that defines the next available output space is placed in a list to be used for reading. When records are "sent" from an input file to an output file by exchanging RDWs, processing must be done in the input area before the RDWs are exchanged. The record may be processed in the input area using indexed instructions as described above for records to be moved.

To illustrate the exchange of RDWs, assume that five RDWs of an input area are in List A, and those for an output area in List B. Individual RDWs will be referred to by their list letter and number in the list; e.g., the fifth RDW in List B will be called B5.

Figure 1A shows the status of each list of RDWs before any RDWs are exchanged.

The RDWs in List A were used to control the reading of the input records that are about to be processed. Previously, the RDWs in List B were used to control the writing of a block of output records; they now represent available spaces.

List A	List B
A1	B1
A2	B2
A3	B3
A4	B4
A5	B5

Figure 1A

List A	List B
B1	A1
B2	A2
A3	B3
A4	B4
A5	B5

Figure 1B

List A	List B
B1	A1
B2	A2
B3	A3
B4	A4
B5	A5

Figure 1C

After two of the records have been processed, the status of the lists would be as shown in Figure 1B. The input records defined in RDWs A1 and A2 have been processed and the RDWs have been placed in List B, while RDWs B1 and B2 have been placed in List A to represent available spaces.

Figure 1C shows the status of both lists after all records in the block have been processed. The RDWs in List B will be used for writing a block of output records defined by RDWs A1 through A5, and List A will be used to control the reading of the next block of input records. It can be seen that List A always controls reading and List B always controls writing although the actual storage locations defined by the RDWs change.

The advantage of RDW exchange over the record move method is especially apparent when large records are being processed; for although the time needed to move a record depends on its length, the exchange of RDWs is independent of record length. The way the form of record affects the time needed for the move and exchange method is explained under "Record Form Operating Time."

Figures 1A, 1B, and 1C illustrate the exchange of RDWs with a simple "balanced" operation; i.e., having one input file and one output file with identical blocking factors. These conditions were chosen to simplify the description, but the RDW exchange is not restricted to "balanced" operations. As mentioned under "System Elements", each tape file has an independent File Scheduler that allows RDWs to be exchanged for practically any type of processing operation. RDW exchange can be used to process many input and output files that may have different blocking factors. Records can be deleted from a file by not referring to them with a PUTX macro-instruction. To insert records into files that are processed by RDW exchange at some point in the program, it is only necessary to move the record into the file using a PUT macro-instruction. The records to be inserted need not be obtained from an input file, but may be created by the program in a work area.

Note that once an input record has been "sent" to an output file by RDW exchange, it can not be moved or exchanged with any other output file. The record move method allows a record to be placed in any number of output files; therefore, if a combination of the move and exchange methods are used, the exchange of RDWs must occur last. For example, if an input file is to be used to produce four duplicate files using both the record move and the RDW exchange methods, records must be placed into the first three output files using the move method; the exchange method may be used to produce the fourth output file.

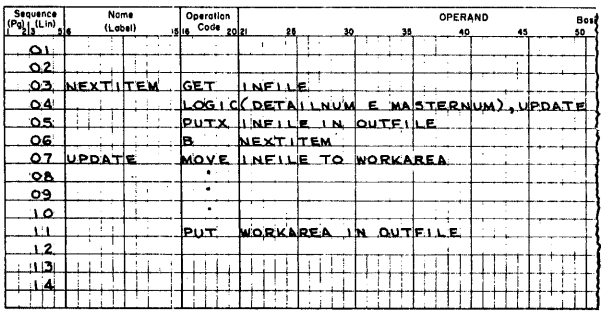
Processing in Work Areas

The record move method of record processing (see "Record Move") requires that the input records be processed in the input area. Records may be processed by moving each record to a work area, and then moving it to an output area. The record may be moved to a work area either immediately by a GET macro-instruction or, after some initial processing, by a MOVE macro-instruction. When processing in the work area is completed, the record must be moved to an output area. RDW exchange is not permitted at any stage of this double-move method.

Although processing records in work areas requires that the record be moved twice, the instructions that are used to process the data in the work area need not be indexed.

For low-activity files, a combination of processing using the MOVE and PUTX macro-instructions can eliminate the time required to move an inactive record twice, while permitting active records to be processed in the work area without indexed instructions. This method of processing requires that each record be checked for activity in the input area after being obtained by a GET macro-instruction. If the check indicates that the record is inactive, the record is entered into an output file by using the PUTX macro-instruction, thereby eliminating the time required to move the record to the work area and then to the output file. When the check indicates an active record, the record is placed into the work area, using the MOVE macro-instruction. The record is processed in the work area using non-indexed instructions, and then entered in the output file by using the PUT macro-instruction. A simplified routine for this method is shown below.

In this example, a record is considered active when the part number in a detail record (DETAILNUM) is equal to the part number in the master file (MASTERNUM).



TAPE RECORD FORMS

The factors that must be considered in selecting the record form to be used for a program are:

1. The amount of storage required for a record,
2. The amount of tape used for a record,
3. The time required to execute the macro-instructions that refer to the records,
4. The convenience of processing the data when it is in a particular form.

IOCS has been designed to process four different record forms. Programs may be written using several tape files that have different record forms provided all records in any one file are of the same form.

Of the four forms processed by IOCS, one (Form 1) refers to blocks of data records having a fixed number of words. The other three (Forms 2, 3, and 4) refer to various methods of controlling the blocking of data records having a variable number of words. Each of these forms is described separately below, together with an explanation regarding the applicability of each form to various situations. A pictorial representation of the four forms of records is shown in Figure 2.

Form 1

Records are referred to as Form 1 records when there is a fixed number of records in each block and each record has a fixed number of words. Form 1 is the simplest, most straightforward of the four forms and should be used whenever the amount of data does not vary from record to record.

Processing of Form 1 records can be done either by moving the records or by exchanging the RDWs of the records.

Form 2

Records are referred to as Form 2 records when there is a fixed number of records in each block and each record has a variable number of words (i.e., any number of words up to a maximum selected by the user). The last word in any Form 2 record which does not contain the maximum number of words must be alphameric, with a record mark in the low-order position. It can be seen that Form 2 records are very similar to Form 1 records, except that records may vary in length under control of record marks at the end of records. Form 2 is suggested for files when the variation in length between records is slight.

Processing of Form 2 records can be done either by moving the records or by exchanging the RDWs of the records.

Form 3

Records are referred to as Form 3 when each block has a variable number of variable-length records; the maximum number of words in a block is selected by the user. A field must be included in a fixed position in each Form 3 record to contain a count of the number of words in the record.

The number of records in a block will not be fixed, but will vary according to the length of the individual records in the block. Records will be placed in a block as long as the addition of a record does not exceed the maximum size of the block. When a record is too long to be inserted into the remaining words of one block, that block is written onto tape, and the record becomes the first record in the next block. For example, if the maximum block size is 100 words, and 96 words have been filled, a 7-word record could not be inserted into this block. A tape block of 96 words would be written, and the 7-word record would become the first record of the next block.

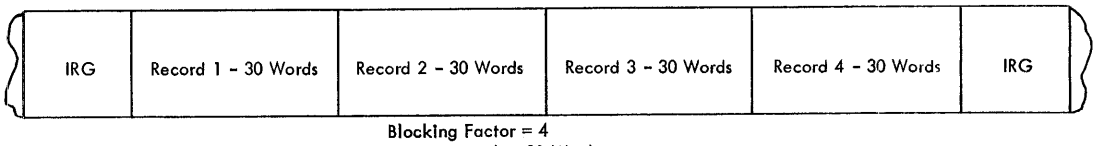
Form 3 is suggested for files consisting of a large number of records that vary in length between wide limits. Although the lengths of Form 3 records may be very different, the block size will tend to be constant as compared with Form 2 records; the block size for Form 2 records can vary considerably if the record lengths have wide variation.

Processing of Form 3 records must be done by moving the records. Form 3 records can not be processed by exchanging the RDWs because this form of record is not read or written under control of individual RDWs.

Form 4

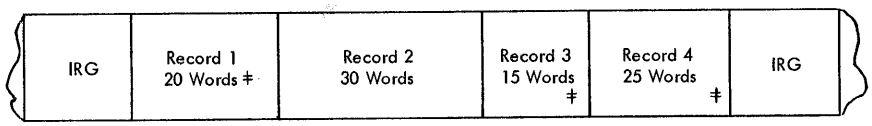
Records are referred to as Form 4 when each block has a fixed number of records, each of which is divided into a fixed number of sections of variable length. Each section of a record may have a different maximum size; the maximum number of words in each section is selected by the user. Whenever

Form 1 Records



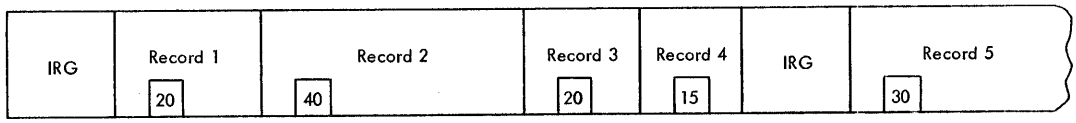
Blocking Factor = 4
Record Length = 30 Words

Form 2 Records



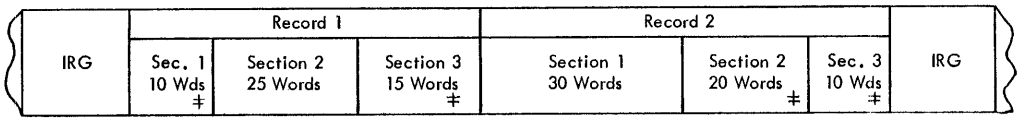
Blocking Factor = 4
Maximum Record Length = 30 Words

Form 3 Records



Maximum Block Size = 120 Words

Form 4 Records



Blocking Factor = 2
Subrecord Blocking Factor = 3
Maximum Length of Section 1 = 30 Words
Maximum Length of Section 2 = 25 Words
Maximum Length of Section 3 = 20 Words

Figure 2

a section contains less than the maximum number of words, the last word in the section must be alphameric, with a record mark in the low-order position.

Form 4 is suggested for files having variable length records that can be divided into a fixed number of sections of variable length. It can be seen that one section of a Form 4 record is similar to an entire Form 2 record. When records are less than the maximum length, Form 2 can result in a saving of time and tape at the end of each record while Form 4 can result in a saving at the end of each section that is not the maximum length.

Processing of Form 4 records can be performed either by moving the records or by exchanging the RDWs of the records. Exchanging RDWs is possible because there is an RDW for each section of a record.

Record Form Operating Time

A factor that may enter into the selection of a record form for a file is the time required by the macro-instructions to process each record form.

The following table shows the approximate operating times for each record form for GET, PUT, and PUTX macro-instructions.

Record Form	Operating Time in Microseconds		
	GET	PUT	PUTX
1	156	300 +24W	300
2	156	300 +24W	300
3	252	396 +24W	Not Used
4	180	360 +24W +36R	480+24R

The W in the table is equal to the number of words moved into the output area; the R is equal to the number of sections in each record or the number of RDWs for each record.

Each time a tape block is completed and IOCS begins using an alternate area, 300 microseconds are added to each of the operating times given in the table. The additional 300 microseconds are required to perform program housekeeping operations associated with the change to another record area.

TAPE LABELS

General Description

To insure the proper mounting of magnetic tapes for each machine run, and to facilitate tape library maintenance, a tape reel control system has been included as an integral part of IOCS. Tape reel control is based on the use of eighty-character alphameric "header" and "trailer" tape labels. Header labels are the first records on each reel of tape, and serve to identify the tape. Trailer labels are (except for a tape mark) the last records on each reel of tape, and indicate whether a reel is the last reel of a file or is to be followed by other reels.

The discussion of header and trailer label formats refers to the label information as it appears on magnetic tape or after being punched or printed. When read into the IBM 7070 from tape, or when constructed within the IBM 7070 preparatory to being written on tape, both header and trailer labels occupy sixteen alphameric words. All fields in the labels are in double-digit form.

Provisions have been made for adding additional label records following the first header label if desired. These additional labels may have any desired format, and need not be the same as header and trailer labels (i.e., eighty alphameric characters).

Labeling Tapes Entering the System

When new magnetic tapes are provided for use in the 7070 system, a temporary header tape label should be written on each tape. The temporary label should remain on the tape until the tape is used in some program. The format of the temporary header tape label is as follows:

Field Number	Positions	Contents	Description
1	1-5	1BLNK	Header Label Identifier
2	6-10	XXXXX	Tape Serial Number
	11-80	Blank	

Field 2, the tape serial number, is a five-digit number from 00001 to 99999 that is assigned consecutively to the tapes as they enter the 7070 system. The number 00000 may not be used.

In addition to the header label written on the tape, a physical label should be attached to each reel of tape. This label should indicate the tape serial number and the date that the tape entered the system. Any other data the user requires may be added to this label. The physical label may remain on the reel until the tape is removed from the system. If at any time the tape serial number is changed, the number on the physical label should be changed accordingly.

Labeling Data Tapes

The header label for a data tape replaces the temporary header label, and contains the following information:

Field Number	Positions	Contents	Description	Example
1	1-5	1HDR	Label Identifier	1HDRb
2	6-19	xxxxxxxxxx-xxx	Tape Serial Number, File Serial Number, and Reel Sequence Number	0012302567-003
	20	blank		
3	21-30	xxxxxxxxxx	File Identification	PAYRLMASTR
4	31-39	xxxxx-xxx	Creation Date-Retention Cycle	58107-030
	40	blank		
5	41-80	miscellaneous	These 40 positions used for any information desired for inclusion	

Explanation of the Header Label

Field 1, Label Identifier (1HDRb): The first five positions serve to identify the record as a label. In addition, the 1 causes skipping to carriage tape channel one when writing the label on a printer with its carriage under program control.

Field 2, Tape Serial Number, File Serial Number, and Sequence Number (xxxxxxxxxx-xxx): The entries within Field 2 are:

1. The five-digit tape serial number is the number initially assigned to the tape when it enters the system.
2. The five-digit file serial number is the tape serial number of the first reel of a given file. For example, if the tape serial number for the first reel of a file is 12567, the label of each succeeding reel in the file will contain 12567 in the file serial number position.
3. The three-digit reel sequence number gives the order of the reel within a given file.

Field 3, File Identification (xxxxxxxxxx): Field 3 is a ten-character name or number identifying the file.

Field 4, Creation Date and Retention Cycle (xxxxx-xxx): The entries within Field 4 are:

1. The first two digits of the five-digit number contain the units and tens position of the year (00-99) in which the file was created. The remaining three digits contain the number (001-366) of the day of the year on which the file was created.
2. The three-digit number following the hyphen indicates the number of days the file is to be retained after the creation date. The date on which a program is being run must be supplied to the program by placing it in location 0109. This date is used when checking a tape label to determine whether the reel of tape may be written upon; the date is also used when writing header labels on output tapes.

Field 5, Miscellaneous: The remaining forty columns may be used in any way desired by the programmer.

Trailer Label Format

A tape mark, a trailer label, and another tape mark form the last three records of each tape. The function of the trailer label is to indicate whether the current tape is the last tape of a file. The trailer label has the following format:

Field Number	Positions	Contents	Field Name	Explanation
1	1-5	1EORb or 1EOFb	Trailer Label Identifier and Termination Code	EOR (End of Reel) EOF (End of File)
2	6-10	xxxxx	Block Count	A count of tape blocks written

Programs written by the IBM Programming Systems Department insert record count and/or hash totals into the proper positions of the trailer label, if such figures are produced by the program. Positions 11 through 20 of the trailer label are used for the record count; positions 21 through 30 are used for the hash total.

The user may include other data in the trailer label if necessary. Additional data may be inserted into the proper positions of the trailer label through the use of program exits, described under "DC Entry for Output File Labels."

PART II: WRITING ENTRIES FOR THE IOCS

The second part of this publication describes the method of supplying the program variables to IOCS. The information that must be supplied pertains to the machine configuration, checkpoints, data areas, and label information. Both the method of writing the macro-instructions provided by IOCS and their operation are explained. Information regarding the provisions made for tape errors and SPOOL programs is also included.

SYSTEM DESCRIPTIVE ENTRY (DIOCS)

When IOCS is to be included in a program, and tape files are to be processed by the program, a DIOCS statement must be written to select the major methods of processing to be used. One form of DIOCS statement is also used to specify system index words. Only the DIOCS statements that serve the functions just described are referred to in the following discussion as "DIOCS entries," "DIOCS statements," or "DIOCS." Other statements in which DIOCS is present in the operation column are always written in this publication so that the required label or operand is coupled with DIOCS, e.g., END DIOCS or DIOCS LINKAGE. If both 729 and 7340 tape units are to be used in a program, there must be a DIOCS entry for each type of unit. No DIOCS is used in programs processing card files only.

Two DIOCS statements are described below. The first DIOCS statement is used for programs processing both card files and 729 tape files, and may be used for programs processing only 729 tape files. It is described under "DIOCS for Programs Using Only 729 Tape Units." The second DIOCS statement is used for programs processing both 7340 and 729 tape files, and may be used instead of the first statement for programs processing only 729 tape files. It is described under "DIOCS for Programs Using 7340 and 729 Units."

DIOCS for Programs Using Only 729 Tape Units

When the only tape units to be used are 729 units, only one DIOCS statement is required for each program, regardless of the number of tape or card files processed in the program. The DIOCS entry to be used in this case is written on one or two lines of the coding sheet as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	DIOCS	IOCSIXF,IOCSIXG,IOCSIXH,CHANn,
02			OPENn,EORn,CHPT,IGENn
03	ANYLABEL	DIOCS	IWNAME,,NN,CHANn,OPENn,EORn,
04			CHPT,IGENn
05	ANYLABEL	DIOCS	,,CHANn,OPENn,EORn,CHPT,IGENn
06			

The letter DIOCS must be placed in the operation column to identify the system descriptive entry.

The first item in the operand (IOCSIXF) is used to specify the first system index word for programs using tape files. An actual two-digit index word address or a symbolic name may be used if the programmer wishes to specify the index word. The symbolic name IOCSIXF will be assigned automatically if the first item in the operand is omitted; the comma following the first entry must be included to indicate the omission. When an actual index word or a symbolic address is specified, Autocoder will equate it to IOCSIXF. (The symbolic name IOCSIXF must not be used in the operand of a DIOCS entry.)

The second item in the operand (IOCSIXG) is used to specify the second system index word for programs using tape files. An actual two-digit index word address or a symbolic name may be used if the programmer wishes to specify the index word. The symbolic name IOCSIXG will be assigned automatically if the second item in the operand is omitted; the comma following the second entry must be included to indicate the omission. When an actual index word or a symbolic address is specified, Autocoder will equate it to IOCSIXG. (The symbolic name IOCSIXG must not be used in the operand of a DIOCS entry.)

The third item in the operand (IOCSIXH) is used to specify a system index word for programs using unit record files. An actual two-digit index word address or a symbolic name may be used if the programmer wishes to specify the index word. The symbolic name IOCSIXH will be assigned automatically if the third item in the operand is omitted; the comma following the third entry must be included to indicate the omission. When an actual index word or a symbolic address is specified, Autocoder will equate it to IOCSIXH. (The symbolic name IOCSIXH must not be used in the operand of a DIOCS entry.)

Under certain conditions, the programmer may wish to use these index words. A typical use of these index words is to return control to the proper point in a program after a subroutine has been completed. This use is explained in more detail under "Program Exits."

The fourth item in the operand (CHANn) is used to specify the highest-numbered tape channel in the program. The value of the final n in CHANn may be 1, 2, 3, or 4.

The fifth item in the operand (OPENn) is used to specify the method of handling the OPEN macro-instruction routine. The value of the final n in OPENn may be 1, 2, 3, 4, 5, or 6. If OPEN1 is entered in the operand, all tape files must be opened at the same time because the OPEN macro-instruction routine will not be preserved in storage after it is used; other routines will be loaded into the locations used by the OPEN routine. If OPEN2 is used, the OPEN macro-instruction routine will be retained in storage for use whenever needed. If OPEN3 is entered in the operand, the OPEN routine will be written on the tape provided for checkpoint records. It will be read into storage whenever needed; the storage locations required for the OPEN routine will be used for other routines during the time the OPEN routine is on tape. If OPEN4 is used, the OPEN macro-instruction routine will be retained in storage for use whenever needed, as described above for OPEN2, except that Form 3 and Form 4 records can not be processed nor can three input/output areas be used for one file. When all active files consist of Form 1 and/or Form 2 records and use one or two input/output areas, the OPEN4 subroutines will occupy fewer storage locations than OPEN2 subroutines.

If any files in a program use the three-area rotating system, either the OPEN5 or OPEN6 subroutine must be specified. OPEN5 is identical to OPEN1 except for the inclusion of provisions for the three-area rotating system. OPEN6 is identical to OPEN2, except for the inclusion of provisions for the three-area rotating system. OPEN1, OPEN2, OPEN3, and OPEN4 must not be specified to open any files using the three-area system.

The sixth item in the operand (EORn) is used to specify tape label processing in the end-of-reel routine. The value of n in EORn may be 1, 2, or 3. The use of EOR1 in the operand specifies that the reading or writing of tape labels is to be determined by Line 32 of the File Specifications for each input and output file (see "File Specifications Entries"). The header labels written on each reel of the output files will be typed automatically by the end-of-reel routine. If EOR2 is used in the operand, none of the input tapes may have labels, and no labels will be written on output tapes. EOR3 is the same as EOR1, except that the header labels written on each reel of output files will not be typed automatically by the end-of-reel routine. The programmer may use end-of-reel exit 4 (see "DC Entry for Output File Labels") to type any portion of an output file header label.

The seventh item in the operand (CHPT) is used to specify whether checkpoint records are to be written. If CHPT is entered in the operand, checkpoint records will be written according to the DCHPT descriptive entry (see "DCHPT Entry"). If CHPT is omitted from the operand, no checkpoint records will be written. If Line 31 of the DTF for an output file is 1 or 2 (see "CHPT Macro-Instruction") CHPT must not be omitted.

The eighth item in the operand (IGENn) is used to specify the use of SPOOL programs and illegal double-digit character checking in the tape write error routine (see "Correction of Output Tape Errors"). The value of the final n in IGENn may be 1, 2, 3, 4, 5, or 6. Entering IGEN1 in the operand indicates that one or more SPOOL programs may operate with this main program, and that the tape error routine is to check for illegal double-digit characters. Use of IGEN2 indicates that one or more SPOOL programs may operate with this main program, but that the tape error routine will not check for illegal double-digit characters. Use of IGEN3 in the operand indicates that a SPOOL program will never be run with this main program, but that the tape error routine is to check for illegal double-digit characters. If IGEN4 is used, no SPOOL program can be run with this main program and the tape error routine will not check for illegal double-digit characters. IGEN5 and IGEN6 are provided for use with programs processing binary tape files (see "File Specifications Entries: Line 09 (FILETYPE)"). IGEN5 performs the same functions as IGEN3. In addition, IGEN5 and IGEN6 treat short character-length records as normal or short length records when executing a binary tape read instruction. When Form 3 format is used, a short character-length record is treated as normal length. When Form 1 format is used, a short character-length record is treated as a normal-length record or a short-length record, depending upon the length of the record.

The operand of the DIOCS entry must contain the following items for each program: CHANn, OPENn, EORn, IGENn. Other items in the operand must be included only when required by the program.

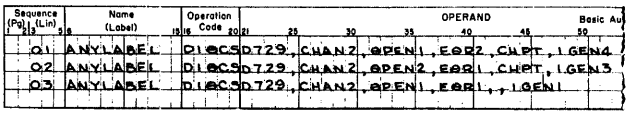
DIOCS for Programs Using 7340 and 729 Units

When 729 tape files are to be used in conjunction with 7340 files, the 729 DIOCS entry is written as described above, with the following exceptions:

1. Systems index words IOCSIXF, IOCSIXG, and IOCSIXH are not written. These values may be established, if desired, by using EQU statements.

2. The first item in the operand must be D729 to indicate that a 729 IOCS is to be generated. This form of 729 DIOCS may also be used for programs processing only 729 tape files.

The operand of the DIOCS entry for 729 tape files must contain the following items for each program: D729, CHANn, OPENn, EORn, and IGENn. The operand may contain CHPT before IGENn. If CHPT is omitted, the comma that would have followed it must be included (see the third line of the example below). If CHPT is entered in the DIOCS for 729 tapes, it must not appear in the DIOCS for 7340 tapes, since the appearance of CHPT in a DIOCS entry indicates whether the checkpoint tape is to be written on a 729 or a 7340 tape unit.



Note that for programs requiring IOCS for 729 files only, the programmer may use the DIOCS statement format described under "DIOCS for Programs Using Only 729 Tape Units," provided no other DIOCS statements of any kind are included in the program.

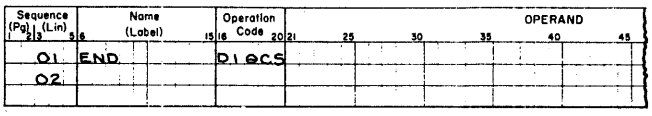
END DIOCS Statement

The END DIOCS statement must be used to indicate to the processor that:

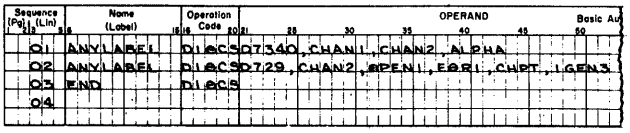
1. No more DIOCS statements follow.
2. IOCS routines should now be generated.
3. Precompiled IOCS routines should now be generated in one block (see "Precompiled IOCS Package").

Note that the END DIOCS statement is not used after the DIOCS statement described under "DIOCS for Programs Using Only 729 Tape Units."

The END DIOCS statement is written as follows:



The following example shows how DIOCS statements are written for 7340 and 729 tape units to be used in the same program:



Note that the END DIOCS statement is used to indicate that no more DIOCS statements follow. Note also that CHPT is entered in the operand of one DIOCS. Because it appears in the DIOCS for 729 IOCS, it indicates that checkpoint records are to be written on a 729 tape unit.

GENERATION OF ROUTINES

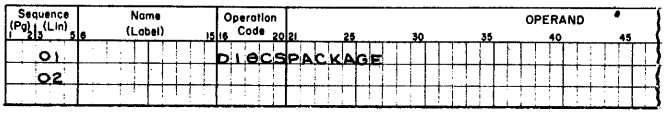
A number of routines are used by more than one IOCS. For instance, the routine that checks for illegal double-digit characters is used for 729 and 7340 files. Other routines, however, are peculiar to the IOCS for one input/output device. Both shared and unique routines are compiled after the END DIOCS statement is encountered. Only those routines are compiled that have been called for, either explicitly or implicitly, in one or more DIOCS statements.

PRECOMPILED IOCS PACKAGE

Obtaining Package

Autocoder can produce a precompiled IOCS package at assembly time if the user so desires, provided that the DIOCS used is the one described under "DIOCS for Programs Using 7340 and 729 Units." If the DIOCS used is the one described under "DIOCS for Programs Using Only 729 Tape Units," see "Precompiled IOCS Subroutine Deck.

To obtain the precompiled package, place the following card before the first DIOCS:



Write the DIOCS statement(s) as explained above. Then, after the last DIOCS, place an END DIOCS card. This card indicates that:

1. No more DIOCS statements follow.
2. A branch list (see below) must be generated immediately preceding the rest of the IOCS package.

3. All IOCS routines are now to be generated in one block.

The following items must be included with the DIOCS statements when compiling the IOCS package:

1. An ORIGIN CNTRL statement, to establish the location for the start of generation of the Precompiled IOCS package.

2. An EQU statement to establish the location of the first word of the first DTF entry. This location must be common to all programs using the precompiled IOCS package. The label of the EQU entry must be IOC.FTBL01.

3. EQU statements to establish fixed locations for the IOCS index words IOCSIXF and IOCSIXG. IOCSIXF and IOCSIXG must also be assigned the same fixed locations in programs using the precompiled IOCS package.

4. An EQU statement to establish the location of the DCHPT entry, if checkpoint routines are included in the precompiled IOCS package.

After compilation, the final execute card and load program should be removed from the condensed deck.

Branch List

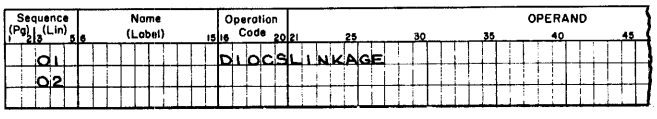
The branch list makes the addresses of certain symbolic labels within each routine in the package available to the source program. This list occupies a number of fixed locations and consists wholly of branch instructions. The operand of each branch instruction contains a symbolic label appearing in an IOCS routine.

The compilation that uses the DIOCS PACKAGE statement produces the precompiled package preceded by the branch list. The compilation that uses the DIOCS LINKAGE statement (see below) produces an assembled object program with linkages to the branch list of the precompiled package.

The use of the branch list makes it possible to make changes in IOCS routines without reassembling source programs.

Using Precompiled IOCS Package

To use a precompiled IOCS package in a program, place the following card before the first DIOCS statement in the symbolic program:



The DIOCS LINKAGE card causes the generation of linkages to the precompiled package branch list.

After the last DIOCS, place an END DIOCS card with its origin at the same location as the END DIOCS card used in compiling the IOCS package. This location may be obtained from a listing of the precompiled package.

When this method is used, no routines are generated when the END DIOCS statement is encountered. Instead, the END DIOCS statement causes the generation of EQU statements that equate symbolic labels of IOCS routines with branch list locations. The user must include his own EQU statements to insure common assignment of IOCS index words IOCSIXF and IOCSIXG in both his program and the IOCS routines.

It is recommended that the DIOCS statements be positioned so that they follow the user's program. However, if a segment of the user's program is to be compiled after the DIOCS statements, an ORIGIN CNTRL statement should be used to position this segment at a point following the last location used by the precompiled IOCS package.

To insure that the coding generated out of line by the Autocoder Processor will not be assigned the same locations as the precompiled IOCS package, a LITORIGIN CNTRL statement should precede the DIOCS statements.

If the programmer wishes to use the FEORN macro-instruction, the following labels must be equated to their respective locations:

IOC.EOR	IOC.SEQILB
IOC.EOFEX	IOC.TEF
IOC.RETEOR	IOC.OPNSW2

If the programmer wishes to use the FEOR macro-instruction, the following labels must be equated to their respective locations:

IOC.ICHECK	IOC.CELOOP
IOC.OPNSW2	IOC.RETRN
IOC.IPSLO	IOC.CEBACK

A program cannot use both precompiled IOCS and IOCS compiled in the normal manner.

FILE SPECIFICATIONS FOR TAPE RECORDS

To be able to handle the large variety of input and output tape files required by various programs, a File Specifications Table is assembled from the File Specifications supplied by the programmer. The File Specifications describe the file, the location of subroutines supplied by the user, and the location of tape label information. A File Specifications Table is generated from each set of File Specifications and

the name of the file is treated as the name of the table. Each time the name of a file is used in a macro-instruction, the Autocoder assembly program selects a sequence of instructions based on information contained in the corresponding File Specifications. The File Specifications Table is used during the running of the program; therefore it must not be moved or erased from storage.

It may be desirable to change items in a File Specifications Table between successive runs of a program. A method of addressing the items in the table is incorporated into the File Specifications to allow changes in, or use of, the data in the table.

The File Specifications are punched into Autocoder cards from the coding sheet described below. These cards are entered with the source program when the program is assembled. The cards containing the File Specifications must be entered in the same sequence as shown on the coding sheet. The File Specifications for all files must be entered in consecutive order when the program is assembled.

There are 36 entries in the File Specifications for each tape file. Refer to Form X28-1366, 7070 File Specifications Coding Sheet (Figure 3). Each line is described separately below; line numbers are used for reference purposes. Note that the line numbers for File Specification entries do not have to be as shown on the coding sheet, provided all entries are made on consecutive lines in the order given.

The label of each entry on the 7070 File Specifications Coding Sheet is printed with recommended characters as the first eight characters of the label. When using these recommended labels, the user must add one or two alphameric characters so that each label will be unique. By using the labels as recommended, the user need not write or keep a record of labels used in the File Specifications. If desired, the user may substitute other labels for those printed on the coding sheet. The label of each entry need not be punched into cards unless the entry will be referred to by the program; the labels of entries not referred to may be blank.

Programmed Entries

The ability to modify data in the File Specifications Table is provided for two purposes. One purpose is to allow the programmer to set the initial value of the operand without having to punch the value into a card when the File Specifications are punched. The other purpose is to allow operands to be changed during the running of a program. Each purpose is described separately below. The description of each entry under "File Specifications Entries" indicates whether the operand may be inserted and/or changed through programming. Unless otherwise stated, all entries that may be inserted through programming may also be changed. The labels of entries to be inserted through programming must be punched into the cards.

Initial Entries

Operands that are not punched into cards when the File Specifications are punched must be entered before the File Specifications Table is used. All operands must be entered except in those entries that specifically allow blank operands. The OPEN macro-instruction initializes the tape files and the routines based on the data in the File Specifications Table; data to be entered must be inserted before the first OPEN macro-instruction is executed. To illustrate one method of programming an entry, assume that a programmer wants to write both labels and records of file 7B at 556 characters per inch (see "File Specifications Entries: Line 21 (TDENSITY)") for a certain run of the program. Using the recommended labels, the label of Line 21 would be TDENSITY7B. A 2, which specifies high density for labels and records, can be inserted by the following steps:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01		ZA3	+2
02		STD3	TDENSITY7B
03			

The first step places a +2 in accumulator 3 and the second step stores the +2 in the tape density entry of the File Specifications Table for file 7B.

Changing Entries: It may be desirable to change data in one or several entries in the File Specifications Table during the running of a program. A typical situation that requires a change in the File Specifications Table arises when a program produces an output file that must be used as input later in the program. For this type of situation, the program is usually considered to be divided into "phases." In this example, the production of the output file would be regarded as one phase and the use of that file as input would be another phase.

Unless stated otherwise under "File Specification Entries," changes in the tables should be made between phases of the program because they require that an OPEN macro-instruction be executed in order to incorporate the changes into the tape files and routines. When data in the File Specification Table of a file is to be changed, the file must be named in a CLOSE macro-instruction before the change is made. After all changes have been made, an OPEN macro-instruction naming all files that have been changed must be given.

The programming of changes is the same as the programming of initial entries. The method described under "Initial Entries" can be used between the phases of a program as well as at the beginning of the program.

File Specifications Entries

Line 01 (TAPEFILE): The label of the DTF entry may be as shown or may be any other symbolic label the user desires. When using the recommended labels, the user must fill in the last two characters of the label with two characters identifying the tape file. This insures that there will be no duplicate labels among the tape files used in the program. When using other than the recommended labels, the user must establish a procedure to avoid duplicate labels.

The operand of the DTF entry contains the name of the tape file described by the File Specifications. This name will be used in the operands of macro-instructions that refer to this tape file. The operand of the DTF header line must not be blank.

Line 02 (FCHANNEL) -- File Channel: The operand may be a one-digit number to indicate the channel to which the tape units specified on Lines 03, 04, and 05 are connected, or it may be blank. If the operand is blank, the channel number must be inserted by programming before the file is named in an OPEN macro-instruction.

The assignment of channel numbers to the files in a program should be made so that each channel will be in use for approximately the same time. When reading and writing times are nearly equal, it is usually advisable to use separate channels for reading and writing.

Line 03 (BASETAPE) -- First Tape in File: The operand may be a one-digit number to specify the tape unit that will contain the first tape of the file, or it may be blank. If the operand is blank, the tape unit number must be inserted by programming before the file is named in an OPEN macro-instruction.

Line 04 (ALT1TAPE) -- First Alternate Tape Unit: The operand may be a one-digit number to specify the first alternate tape unit to be used in conjunction with the base unit specified on Line 03, or it may be blank. This operand must be different from the operand of Line 03. The tape unit number may be inserted by programming before the file is named in an OPEN macro-instruction. If no alternate tape unit is to be used, the operand must be blank.

Line 05 (ALT2TAPE) -- Second Alternate Tape Unit: The operand may be a one-digit number to specify the second alternate tape unit to be used in conjunction with the units specified on Lines 03 and 04, or it may be blank. This operand must be different from those used for Lines 03 and 04. The tape unit number may be inserted by programming before the file is named in an OPEN macro-instruction. If a second alternate tape unit will not be used, this operand must be blank.

Line 06 (ACTIVITY): The operand of this line must be blank. The macro-instructions OPEN, CLOSE, and END will automatically insert a 0 or a 1 into this entry for all files named in their operands. An OPEN macro-instruction will insert a 1 to indicate that the tape file is to be active (used). A CLOSE or END macro-instruction will insert a 0 to indicate that the file is to be inactive (not used).

Line 07 (BLOCKCNT) -- Block Count: The operand of this line must be blank. IOCS will insert a count of the number of tape record blocks read from, or written into, the file.

Line 08 (FILEFORM): The operand is a one-digit number from 1 to 4 to indicate the form of records in the tape file. Form 1, 2, 3, and 4 records are described under "Tape Record Forms." The operand of this line must never be blank and must never be changed during the running of a program. If the operand is changed, reassembly may be necessary.

Line 09 (FILETYPE): The operand is a one-digit number from 0 to 6 to indicate the type of operation to be executed with the tape file. The type of operation specified by each number is as follows:

0 -- a read binary operation
1 -- a read operation
2 -- a read operation under record mark control
3 -- a write operation
4 -- a write operation under record mark control
5 -- a write operation with zero elimination
6 -- a write operation with zero elimination under record mark control

The operand of this line must never be blank. A change in this operand necessitates reassembly if a change in record form is involved. Note that even in that case, if a change from Form 1 to Form 2 records or from Form 2 to Form 1 records is involved, reassembly is not necessary. When reassembly is not required, changes in this operand during the running of a program are possible only at the end of one phase and before the start of the next phase; i.e., after the CLOSE macro-instruction for the first phase is executed, but before the OPEN macro-instruction for the second phase is executed.

Line 10 (RECLNGTH) -- Record Length: The operand must be a four-digit number specifying the record length. If the operand is blank, the record length must be inserted through programming. The

record length will be either the number of words in Form 1 records or the maximum number of words in Form 2, 3, or 4 records. In order to change this operand, reassembly is necessary.

Line 11 (BLOCKING): The operand must be a four-digit number specifying the tape record blocking. If the operand is blank, the number must be inserted through programming. The record blocking will be either the number of records in a tape block for Form 1, 2, or 4 records, or the maximum number of words in a tape block for Form 3 records. In order to change this operand, reassembly is necessary.

Line 12 (OPENPROC) -- Open Procedure: The operand may be 0 or 2 to specify the rewind procedure to be used when an OPEN macro-instruction is executed, or it may be blank. The digit may be inserted through programming. A 0 indicates that the tape is not to be rewound. A 2 indicates that the tape is to be rewound when the OPEN macro-instruction is executed. Unless a 0 or 2 is inserted, the blank will be regarded as a 0. This operand may be changed at any time.

Line 13 (CLSEPROC) -- Close Procedure: The operand may be 0, 2, or 3 to specify the rewind procedure to be used when a CLOSE macro-instruction is executed, or it may be blank. The digit may be inserted through programming. A 0 indicates that the tape is not to be rewound, a 2 indicates that the tape is to be rewound, and a 3 indicates that the tape is to be rewound and unloaded when the CLOSE macro-instruction is executed. Unless a 0, 2, or 3 is inserted, the blank will be regarded as a 0. This operand may be changed at any time.

Line 14 (TPERROPT) -- Tape Error Option: The operand may be 00, 10-50, or 60, to specify the procedure to be used if uncorrected tape errors occur when reading an input file. The operand can also be blank. This operand may be inserted through programming. It specifies the error correction procedure to be followed whenever a tape reading error for this file occurs. An explanation of each error correction procedure that may be selected by this entry is given under "Correction of Input Tape Errors." Note that this operand must be blank for output files.

Line 15 (IORDWLST) -- Input/Output RDW List: The operand may be an actual, symbolic, or blank address. If blank, the address must be inserted through programming. The operand must contain the address of the first of the RDWs for the tape record area. The RDWs may be generated by either a DA or a DRDW operation as described under "Tape Record Areas."

If this file is to share input/output areas with another file (see Line 16), the address that is used in this operand must also be used as the operand of the IORDWLST entry of the File Specifications for the other file. The same RDWs must be used for both the input and the output file. The RDWs may be generated either by a DA entry or by a DRDW operation (see "Tape Record Areas").

Line 16 (IOMETHOD) -- Input/Output Method: The operand is either 3 or blank. A 3 indicates that an input file and an output file will share the same three areas for both input and output in rotating sequence. The operand of this line must never be changed during the running of a program.

If the operand is changed, reassembly may be necessary.

Whenever a 3 is placed in the operand, the following restrictions must be observed:

1. Records may not be added to, or deleted from the file.
2. Input and output record form must be identical.
3. Record length and blocking factor must be the same for both files.
4. The number of areas used must be 3 (see Line 17).
5. Both files must use the same RDWs (see Line 15).

Line 17 (TIOAREAS) -- Tape Input/Output Areas: The operand is a one-digit number from 1 to 3 to specify the number of areas to be used by the corresponding file. The selection of the number of areas to be used for a file is described under "Input/Output Areas." If the method of handling tape records uses record areas in rotating sequence (see Line 16), a 3 must be placed in the operand of this line. The operand of this line must never be changed during the running of a program. If the operand is changed, reassembly may be necessary.

Line 18 (PRIORITY): The operand is either a one-digit number from 1 to 9 to specify the relative priority of the tape file, or it is blank. The use of "priority" in this entry does not refer to the Priority Processing feature of the IBM 7070; as used here, "priority" refers to the relative importance of the preferred order of the tape files. A 1 indicates the file with the highest priority and a 9 the lowest priority. The number may be inserted through programming. A priority number may be assigned to a maximum of nine tape files. This number will be used to determine which tape record area is to be read into or written from whenever two or more areas become available at the same time. If the user does not wish to specify a priority number or if numbers 1 through 9 have already been assigned, the operand must be blank. Note that no two files should be assigned the same values (1-9) at the same time.

X28-1366

IBM

Program

Programmed by

Date

7070 AUTOCODER CODING SHEET
7070 INPUT/OUTPUT CONTROL SYSTEM
TAPE FILE SPECIFICATIONS

Identification 76 80

Page No. 1 2 of

Line	Label	Operation	Operand
0,1	TAPEFILE	DTF	
0,2	FCHANNEL		
0,3	BASETAPE		
0,4	ALT1TAPE		
0,5	ALT2TAPE		
0,6	ACTIVITY		
0,7	BLOCKCNT		
0,8	FILEFORM		
0,9	FILETYPE		
1,0	RECLNGTH		
1,1	BLOCKING		
1,2	OPENPROC		
1,3	CLSEPROC		
1,4	TPERROPT		
1,5	IOERDWLST		
1,6	IOMETHOD		
1,7	TIOAREAS		
1,8	PRIOPRTY		
1,9	INDXWRDA		
2,0	INDXWRDB		
2,1	TDENSITY		
2,2	SLRPROCD		
2,3	LLRPROCD		
2,4	SCLPROCD		
2,5	TPERRPLD		
2,6	TPSKPLD		
2,7	EOSPROCD		
2,8	EORPROCD		
2,9	EOFPROCD		
3,0	RWDPROCD		
3,1	CHECKPNT		
3,2	LABELINF		
3,3	SRBFOPM4		
3,4	RLIFOPM3		
3,5	SPARELNF		
3,6	SCHEDLNF		

Figure 3

Line 19 (INDXWRDA) -- Index Word A: The operand, which specifies the index word to be used as Index Word A (XA) for the file, may be actual or symbolic. If actual, the operand must be a two-digit index word address. When using an actual address, the selection of the index word may be affected by other programs as mentioned under "Summary of Storage, Index Word, and Electronic Switch Utilization." The operand should not be changed. The contents of XA are described under "Use of Tape Record Index Words."

Line 20 (INDXWRDB) -- Index Word B: The operand, which specifies the index word to be used as Index Word B (XB) for the file, may be actual or symbolic. If actual, the operand must be a two-digit index word address. When using an actual address, the selection of the index word may be affected by other programs as mentioned under "Summary of Storage, Index Word, and Electronic Switch Utilization." The operand should not be changed. The contents of XB are described under "Use of Tape Index Words."

Line 21 (TDENSITY) -- Tape Density: The operand is a one-digit number from 0 to 2 to specify the tape character density of the tape file, or it is blank. A 0 specifies that both the labels and the records on a tape are written, or are to be written, at a low character density (normally 200 characters per inch). A 1 specifies that the labels use a low character density (normally 200 characters per inch) while the records in the file use a high character density (normally 556 characters per inch). A 2 specifies that both the labels and the records use a high character density (normally 556 characters per inch). This entry will not be effective for tape files on IBM 7330 Magnetic Tape Units. When model 729 V or 729 VI Magnetic Tape Units are used, the Tape Adapter Unit associated with a 729 V or 729 VI can be manually set to any one of the following combinations of high and low density: 800cpi/556cpi, 800cpi/200cpi, or 556cpi/200cpi. When the switch is set to 800cpi/556cpi, a 1 inserted in the TDENSITY entry will specify that the labels use a character density of 556 characters per inch while the records in the file use a character density of 800 characters per inch. The setting of the Tape Adapter Unit may not be changed during the program.

Line 22 (SLRPROCD) -- Short Length Record Procedure: The operand is an address that may be

actual, symbolic, blank, or 9999. The address should specify the location of a Short Length Record routine for an input file; the routine must end with a Branch to 0+IOCSIXG if the user wishes to return to IOCS. The address may be inserted through programming. If no address is inserted, a short length record will cause a message to be typed and then halt the machine to allow the operator to decide what action must be taken. If the user expects to return to IOCS from the Short Length Record exit, he may give IOCS macro-instructions for other files, provided he saves the contents of IOCSIXG before giving these instructions and then restores the contents after the execution of the instructions.

No IOCS macro-instruction should be given for this file if the user expects to return from the exit to IOCS. If the user does not wish to return to IOCS, any macro-instruction may be given. If 9999 is inserted in the operand, a short length record will be regarded as a correct length record, and processing will continue. The operand may be changed at any time. This entry will not be effective for output files or input files having Form 3 records.

Line 23 (LLRPROCD) -- Long Length Record Procedure: The operand is an address that may be actual, symbolic, blank, or 9999. The address should specify the location of a Long Length Record routine for an input file; the routine must end with a Branch to 0+IOCSIXG if the user wishes to return to IOCS. The address may be inserted through programming. If no address is inserted, a long length record will cause a message to be typed and then halt the machine to allow the operator to decide what action must be taken. If 9999 is inserted into the operand, a long length record will be regarded as a correct length record and processing will continue. The operand may be changed at any time. This entry will not be effective for output files.

For this exit, IOCS macro-instructions for other files may be given provided the user saves the contents of IOCSIXG before giving these instructions and then restores the contents after the execution of the instructions.

No IOCS macro-instruction should be given for this file if the user expects to return from the exit to IOCS. If the user does not wish to return to IOCS, he may give any macro-instruction with the exception that if this file contains Form 3 records, a GET macro-instruction must not be given.

Line 24 (SCLPROCD) -- Short Character Length Procedure: The operand is an address that may be actual, symbolic, blank, or 9999. The address should specify the location of a Short Character Length Record routine for an input file; the routine must end with a Branch to 0+IOCSIXG instruction if the user wishes to return to IOCS. The address may be inserted through programming. If no address is inserted, a short character-length record will cause a message to be typed and then halt the machine to allow the operator to decide what action must be taken. If 9999 is inserted in the operand, a short character-length record will be regarded as a correct length record, and processing will continue. The operand may be changed at any time. This entry will not be effective for output files.

For this exit, IOCS macro-instructions for other files may be given provided the user saves the contents of IOCSIXG before giving these instructions and then restores the contents after the execution of the instructions.

No IOCS macro-instruction should be given for this file if the user expects to return from the exit to IOCS. If the user does not wish to return to IOCS, he may give any macro-instruction with the exception of the case that if this file contains Form 3 records, a GET macro-instruction must not be given.

Line 25 (TPERRFLD) -- Tape Error Field: The operand of this line must be blank. IOCS will insert a count of the input tape errors that were not corrected in nine attempts made in the error correction routine.

Line 26 (TPSKPFLD) -- Tape Skip Field: The operand of this line must be blank. IOCS will insert a count of the number of times that the Tape Skip (TSK) instruction was executed during the writing of an output file.

Line 27 (EOSPROCD) -- End-of-Segment Procedure: The operand is an address that may be actual, symbolic, blank, or 9999. The address should specify the location of an end-of-segment routine for an input file; the routine must end with a Branch to 0+IOCSIXG instruction if the user wishes to return to IOCS. The address may be inserted through programming. If no address is inserted, an end-of-segment indication will cause a message to be typed and then halt the machine to allow the operator to decide what action must be taken. If 9999 is inserted in the operand, an end-of-segment indication will be ignored and processing will continue. The operand may be changed at any time. This entry will not be effective for output files.

No IOCS macro-instructions should be given on this file if the user expects to return from the exit to IOCS. If the user does not wish to return to IOCS, any macro-instruction may be given.

Line 28 (EORPROCD) -- End-of-Reel Procedure: The operand is an address that may be actual, symbolic, or blank. The address may specify the starting location of additional instructions that are to be executed when an end-of-reel condition occurs. These instructions will be executed in addition to the

routine provided for an end-of-reel condition by the IOCS. The return from this exit must be a Branch to 0+IOCSIXF. The address may be inserted through programming. If no additional instructions are to be executed, the operand must be blank. The operand may be changed at any time.

Line 29 (EOFPROCD) -- End-of-File Procedure: The operand is an address that may be actual, symbolic, or blank. The address must specify the location of the user's end-of-file routine. If blank, the address must be inserted through programming. When the trailer record of an input file indicates that the end of the file has been reached, the end-of-file routine will be entered. The end-of-file condition will be detected when the trailer record is read and checked during the end-of-reel routine provided by the IOCS. The operand may be changed at any time.

An end-of-file routine must be provided by the programmer. The operations to be performed during this routine are determined by the programmer. Typical operations for an end-of-file routine would be checks to determine:

1. If the end-of-file has been reached for all input files of a multifile program.
2. If the end of a phase has been reached.
3. If the end of the job has been reached.

Depending on the results of the checks and the requirements of the program, the end-of-file routine can perform the required function. For example, if a check indicated an end-of-job condition, the programmer would want to execute at least a CLOSE or an END macro-instruction.

No IOCS macro-instructions may be given in the EOFPROCD routine if the user intends to return to IOCS. IOCS macro-instructions may be given if the user does not wish to return to IOCS.

If the input file does not have tape labels, the end-of-file routine specified by this address will be entered each time a tape mark is read. The user's end-of-file routine must determine whether the tape just read is the last reel of the file, because IOCS depends on a trailer label to differentiate between the end of a reel and the end of a file. Other checks to be made in the end-of-file routine are the same as stated above. When the user's end-of-file routine determines that the reel of tape just read is not the last reel of a file, processing is continued by branching to 0+IOCSIXG.

Line 30 (RWDPROCD) -- Rewind Procedure: The operand may be 0, 2, 3, 4, 6, 7 to specify rewind procedures to be followed after a trailer label has been read or written, or it may be blank. This operand specifies the rewind procedure for both the current reel of a file and the next reel of the file, if there is one. The rewind procedures that may be used are as follows:

Operand	Action for Current Reel	Action for Next Reel, if any
0	No Rewind	Rewind
2	Rewind	Rewind
3	Rewind and Unload	Rewind
4	No Rewind	No Rewind
6	Rewind	No Rewind
7	Rewind and Unload	No Rewind

The operand may be inserted through programming. If a digit is not inserted, the blank will be regarded as a 0. The operand may be changed at any time.

Line 31 (CHECKPNT) -- Checkpoint: The operand may be 2, 1, or blank. A 2 indicates that this file is an output file that will also be used as the checkpoint tape, and that this output file will have alternate reels.

For an output file, a 1 indicates that this file will also be used as the checkpoint tape. Alternating of tapes will not be allowed. For an input file, a 1 indicates that checkpoint records are contained between data records of this file.

If this is an output file and the operand is 1 or 2, CHPT must be included in the DIOCS statement.

For an output file, a blank signifies that this file will not be used as a shared checkpoint tape. For an input file, a blank signifies that no checkpoint records have been written on tapes of this file.

Line 32 (LABELINF) -- Label Information: The operand is an address that may be actual, symbolic, or blank. This address must specify the location of the DC entry that defines data and procedures associated with tape header and trailer labels for the file. The address may be inserted through programming. A blank operand indicates that this is either an unlabeled input file or an output file that is not to be labeled. If this operand is changed, reassembly of the program may be necessary. This entry will be ignored if the sixth item in the operand of the DIOCS entry is EOR2. When this file is being used as a shared checkpoint tape, the LABELINF entry must be restored by the user after the file is opened.

Line 33 (SRBFORM4) -- Subrecord Blocking Form 4 Records: The operand is either a four-digit number to specify the subrecord blocking for Form 4 records or blank. A four-digit entry is used for Form 4 records only. The operand must be blank for all other record forms. When Form 4 records are used, the

operand must not be blank and must never be changed during the running of a program. For Form 4 records, the operand must contain the number of sections (subrecords) that make up one record. If the operand is changed, the program must be reassembled.

Line 34 (RLIFORM3) -- Record Length Indicator Form 3 Records: This entry is used for Form 3 records only and the operand must be blank for all other record forms. When Form 3 records are used, the operand must not be blank and must never be changed during the running of a program. For Form 3 records, the operand must contain the position of the record length field within the record. The position of the record length field is specified by giving the first and last digit positions of the field as is done when defining a field under a DA entry. (Field definition is described in IBM 7070 Series Programming Systems: Autocoder, Form C28-6121.)

The record length field must be positioned so that it is contained in one word of the record, i.e., the field may not begin in one word and end in the next. If the position of the record length field is changed, reassembly will be necessary.

Line 35 (SPAREINF) -- Spare Information: This entry represents a two-digit field that is available to the programmer for any desired purpose. This field may not be used for programs compiled in the additional storage mode.

Line 36 (SCHEDINF) -- Schedule Information: The operand of this entry must be blank. This entry reserves an area for use by IOCS.

RETURNS FROM IOCS EXITS

The following list indicates the returns from the IOCS exits.

IOCS Exit	Return
All Label Exits (No macro-instructions may be given in these routines)	0+IOCSIXF
End-of-File Exit (For an unlabeled file)	0+IOCSIXG
End-of-Reel Exit	0+IOCSIXF
End-of-Segment Exit	0+IOCSIXG
Long Length Record Exit	0+IOCSIXG
Short Character Length Record Exit	0+IOCSIXG
Short Length Record Exit	0+IOCSIXG

TAPE RECORD AREAS

An area must be defined for each tape input and output file used in the program. Each area must be defined to indicate the blocking and the arrangement of the fields within a tape record. The area is defined by a DA entry as described in the Autocoder reference manual. The IOCS requires that RDWs be created for each of the tape record areas. The user has the option of creating RDWs by either a DA or a DRDW operation; both methods are described below. The various tape record forms require slightly different methods of defining the tape record areas; each method of definition will be described separately below.

Defining Areas for Form 1 and Form 2 Input Records

If the RDWs are to be generated using the DA operation, the DA entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DA	N,RDW,0+INDXWRDA
02	FIELD1		00,09
03	FIELD2		10,16
04	FIELD3		17,34
05	FIELD4		35,42
06	FIELD5		43,59
07			

If the RDWs are to be generated using the DRDW operation, the DA entry and the DRDW entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	DANAME	DA	N,,0+INDXWRDA
02	FIELD1		00,09
03	FIELD2		10,16
04	FIELD3		17,34
05	FIELD4		35,42
06	FIELD5		43,59
07			

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	DANAME
02			

The use of the label of the DA entry will depend on the method used for generating the RDWs. If the RDWs are to be generated by the DA entry, the label must be entered on Line 15 in the File Specifications for the corresponding input tape file. If the RDWs are to be generated by a DRDW operation, the label of the DA entry must be used in the operand of the DRDW entry, and the label of the DRDW entry must be entered on Line 15 in the File Specifications.

The value of N in the operand is equal to the number of records in a tape block multiplied by the number of areas to be used for the file.

The next item in the operand depends on the method to be used for generating the RDWs; see the example above. IOCS will assign plus and minus signs to the RDWs as required by the tape file, based on data supplied in the File Specifications.

The next item in the operand is a zero so that fields within the record will be defined relative to zero and can be referred to using index word XA for the file.

The last item in the operand is +INDXWRDA, which automatically assigns indexing to instructions that refer to fields defined in the subsequent entries. The symbolic name must be the same as the operand used for index word XA of the File Specifications (see Line 19). Use of an index word in a DA entry is described in the Autocoder reference manual. If the program is to be assembled with Four-Tape Autocoder, the index word cannot be included in the operand of the DA entry; to refer to fields defined relative to zero, the user must specify the index word in each instruction that refers to those fields.

Subsequent entries under the DA are used to describe one record in the block. These entries name the fields, specify the location of the fields within the record, and specify the form of data in the field. A detailed description of the subsequent entries may be found in the reference manual IBM 7070 Series Programming Systems: Autocoder, Form C28-6121.

Defining Areas for Form 1 and Form 2 Output Records

If the RDWs are to be generated using the DA operation, the DA entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DA	N,RDW
02			00,59
03			

If the RDWs are to be generated using the DRDW operation, the DA entry and the DRDW entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	DANAME	DA	N
02			00,59
03			

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	DANAME
02			

The use of the label of the DA entry will depend on the method used for generating the RDWs. If the RDWs are to be generated by the DA entry, the label must be entered on Line 15 in the File Specifications for the corresponding input tape file. If a DRDW operation is used, the label of the DA entry must appear in the operand of the DRDW entry and must be the same as the label of the DRDW entry already entered on Line 15 in the File Specifications.

The value of N in the operand is equal to the number of records in a tape block multiplied by the number of areas to be used for the file.

The operand of the DA entry may include the letters RDW if the DA operation is to generate the necessary RDWs. If the letters RDW are omitted, a DRDW operation must be included in the program; see the examples above. IOCS will assign plus and minus signs to the RDWs as required by the tape file, based on data supplied in the File Specifications.

One subsequent entry under the DA is necessary to specify the area required for one record.

Defining Areas for Form 3 Input Records

If the RDWs are to be generated using the DA operation, the DA entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DA	N,RDW,0+INDXWRDA
02	FIELD1		00,12
03	FIELD2		13,18
04	FIELD3		19,29
05	FIELD4		30,37
06	FIELD5		38,45
07			46,499
08			

If the RDWs are to be generated using the DRDW operation, the DA entry and the DRDW entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	DANAME	DA	N,,0+INDXWRDA
02	FIELD1		00,12
03	FIELD2		13,18
04	FIELD3		19,29
05	FIELD4		30,37
06	FIELD5		38,45
07			46,499
08			

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	DANAME
02			

The use of the label of the DA entry will depend on the method used for generating the RDWs. If the RDWs are to be generated by the DA entry, the label must be entered on Line 15 in the File Specifications for the corresponding input tape file. If a DRDW operation is used, the label of the DA entry must be used as the operand of the DRDW entry and must be the same as the label of the DRDW entry already entered on Line 15 in the File Specifications.

The value of N in the operand is 1, 2, or 3 to specify the number of areas to be used for the file.

The next item in the operand depends on the method to be used for generating the RDWs; see the examples above. One RDW will be generated for each area. Only one RDW is used for each area because the number of records in each block will be different; this prevents the generation of an RDW for each record in advance as for Form 1 and 2 records.

The remainder of the operand of the DA entry is the same as described above for Form 1 and 2 input records.

Subsequent entries under the DA are the same as described for Form 1 and 2 input records with one addition. One entry under the DA must specify the maximum permissible number of words in a block. This additional entry is illustrated by the last entry in the examples above; the entry may have a label if the user desires.

Defining Areas for Form 3 Output Records

If the RDWs are to be generated using the DA operation, the DA entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DA	N,RDW
02			00,499
03			

If the RDWs are to be generated using the DRDW operation, the DA entry and the DRDW entry may be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	DANAME	DA	N
02			00,499
03			

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	DANAME
02			

The use of the label of the DA entry will depend on the method used for generating the RDWs. If the RDWs are to be generated by the DA entry, the label must be entered on Line 15 in the File Specifications for the corresponding output tape file. If a DRDW operation is used, the label of the DA entry must be used as the operand of the DRDW entry and must be the same as the label of the DRDW entry already entered on Line 15 in the File Specifications.

The value of N in the operand is 1, 2, or 3 to specify the number of areas to be used for the file.

The operand of the DA entry may include the letters RDW if the DA operation is to generate the necessary RDWs. If the letters RDW are omitted, a DRDW operation must be included in the program; see the examples above.

One subsequent entry under the DA is necessary to specify the maximum number of words in one block of output records.

Defining Areas for Form 4 Input Records

Form 4 records require that a DA entry be written for each part (section) of each input record in each record area. To illustrate the writing of these DA entries, the following example assumes input records divided into three parts with two records contained in each block; two tape record areas are used. The label of each DA identifies the area, record, and part. For example, AR2RD1PT3 is the label of part 3 of record 1 in area 2.

Each DA entry for a Form 4 record must have a unique label, which will be used when creating the RDWs to handle these records.

The first DA entry for each part of a record has an operand in the form 1,, ADDR (see Lines 01, 06, and 08). The 1 indicates that one area is to be established for each part of the record. The two successive commas specify that no RDW is to be created by the DA entry. The RDWs for Form 4 records are specified as described under "RDWs for Form 4 Input Records." The ADDR specifies the relative address of the first position of each part of the record. In the example, part 1 is relative to location 0; part 2 is relative to 19, i.e., the first word following the end of part 1. Part 3 is specified relative to word 24, which is the first word following the end of part 2. The DA entries can be written as shown below.

The fields within each part of the record are defined once by subsequent entries under the first DA entry for each part. In the example, part 1 has fields 1 through 4, part 2 has field 5, and part 3 has field 6.

When using Form 4, the last field in each part usually will be variable in length and under record mark control; the maximum field length of such variable fields must be used when defining them.

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	AR1RD1PT1	DA	1,,0
02	FIELD1		00,15
03	FIELD2		16,35
04	FIELD3		36,79
05	FIELD4		80,189
06	AR1RD1PT2	DA	1,,19
07	FIELD5		00,49
08	AR1RD1PT3	DA	1,,24
09	FIELD6		00,89
10	AR1RD2PT1	DA	1
11			00,189
12	AR1RD2PT2	DA	1
13			00,49
14	AR1RD2PT3	DA	1
15			00,89
16	AR2RD1PT1	DA	1
17			00,189
18	AR2RD1PT2	DA	1
19			00,49
20	AR2RD1PT3	DA	1
21			00,89
22	AR2RD2PT1	DA	1
23			00,189
24	AR2RD2PT2	DA	1
25			00,49
26	AR2RD2PT3	DA	1
27			00,89

The DA entries for other records in the block and in other tape record areas contain a 1 in the operand. There will be one subsequent entry under each DA entry to specify the area required by the corresponding part of the record. Lines 10 through 27 of the coding sheet show the DA entries and the subsequent entries required by the example.

RDWs for Form 4 Input Records

IOCS requires that the RDWs for the tape record areas be in consecutive storage locations. When RDWs are created by a DA entry, the RDWs are located immediately preceding the fields defined by the subsequent entries. The requirement of one DA entry for each part of a Form 4 record would result in RDWs separated by the fields they define. To provide the RDWs in the order required by IOCS, Form 4 RDWs are created by using the Define Record Definition Word (DRDW) operation as described in the Autocoder reference manual. The DRDW entries required by the example given for Form 4 input records would be written as shown below.

The label of the first DRDW must be the same as the operand of Line 15 in the File Specifications for the corresponding input tape file.

The first character of the operand of each DRDW entry must be either a plus or minus sign. All operands will be plus except the one for the last DRDW in each tape record area. Lines 06 and 12 show the last DRDW entries for the two areas of the example. The remaining portion of the operands must be the same as the labels of the DA entries for each part of the Form 4 input records.

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	+AR1RD1PT1
02		DRDW	+AR1RD1PT2
03		DRDW	+AR1RD1PT3
04		DRDW	+AR1RD2PT1
05		DRDW	+AR1RD2PT2
06		DRDW	-AR1RD2PT3
07		DRDW	+AR2RD1PT1
08		DRDW	+AR2RD1PT2
09		DRDW	+AR2RD1PT3
10		DRDW	+AR2RD2PT1
11		DRDW	+AR2RD2PT2
12		DRDW	-AR2RD2PT3
13			

Defining Areas for Form 4 Output Records

The DA entries required for Form 4 output records are similar to those required for input records. The major difference is that the limits of the fields within each part of the record need not be entered.

To show the similarity between input and output DA entries, the example used to illustrate DA entries for output records will assume the same arrangement of records (three parts per period, two records per block, and two areas). The DA entries for this example would be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	AR1RD1PTA	DA	1
02			00,189
03	AR1RD1PTB	DA	1
04			00,49
05	AR1RD1PTC	DA	1
06			00,89
07	AR1RD2PTA	DA	1
08			00,189
09	AR1RD2PTB	DA	1
10			00,49
11	AR1RD2PTC	DA	1
12			00,89
13	AR2RD1PTA	DA	1
14			00,189
15	AR2RD1PTB	DA	1
16			00,49
17	AR2RD1PTC	DA	1
18			00,89
19	AR2RD2PTA	DA	1
20			00,189
21	AR2RD2PTB	DA	1
22			00,49
23	AR2RD2PTC	DA	1
24			00,89
25			

Each DA entry for a Form 4 record must have a unique label. This label will be used when creating the RDWs to handle these records.

The operand of each DA entry contains a 1 to establish one area for each part of the output record. There will be one subsequent entry under each DA to specify the area required by the corresponding part of the record.

RDWs for Form 4 Output Records

The RDWs for output records are specified in the same manner as those described for input records. The DRDW entries required by the example given for Form 4 output records would be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	LISTNAME	DRDW	+AR1RD1PTA
02		DRDW	+AR1RD1PTB
03		DRDW	+AR1RD1PTC
04		DRDW	+AR1RD2PTA
05		DRDW	+AR1RD2PTB
06		DRDW	-AR1RD2PTC
07		DRDW	+AR2RD1PTA
08		DRDW	+AR2RD1PTB
09		DRDW	+AR2RD1PTC
10		DRDW	+AR2RD2PTA
11		DRDW	+AR2RD2PTB
12		DRDW	-AR2RD2PTC
13			

The label of the first DRDW must be the same as the operand of Line 15 in the File Specifications for the corresponding output tape file.

The first character of each operand must be either a plus or minus sign. All operands will be plus except the one for the last DRDW in each tape record area. The remaining portion of the operands must be the same as the labels of the DA entries for each part of the Form 4 output records.

INPUT/OUTPUT MACRO-INSTRUCTIONS FOR TAPE FILES

The input/output macro-instructions are written as Autocoder instructions with the operand containing the symbolic location(s) of the information to be processed. The symbolic location may be either the name of a tape file defined by File Specifications or the name of an area in storage defined by a DA or DC entry. The operation portion of the macro-instruction is the mnemonic representation of one of the functions to be performed by the IOCS. Each of these functions is described separately below.

Each macro-instruction causes a sequence of instructions to be selected; the number of instructions required depends on the function and on the information given in the File Specifications. These instructions are selected and inserted into the object program during Autocoder assembly.

OPEN Macro-Instruction

Before beginning to use an input or output file, the file must be initialized for processing by use of the macro-instruction OPEN. The OPEN macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	OPEN	FILE1,FILE2,FILE3
02			

The items entered in the operand depend on the OPENn item in the operand of the DIOCS entry as follows:

1. If the operand of the DIOCS macro-instruction contains OPEN1 or OPEN5, the operand of the OPEN macro-instruction must name all tape files used by the program. When OPEN1 or OPEN5 is used, the programmer must use the OPEN macro-instruction in accordance with the procedures described under "System Descriptive Entry (DIOCS)."

2. If OPEN2, OPEN3, OPEN4, or OPEN6 is placed in the operand of the DIOCS macro-instruction, the operand of the OPEN macro-instruction must contain the names of all new (not already opened) tape files to be used in the portion of the program which follows the OPEN. The names in the operand must be the same as the names used as the operand of the DTF entry of the File Specifications for those files.

An OPEN macro-instruction is always used near the beginning of a program, to initialize the tape files prior to processing data. The OPEN macro-instruction is also used when data in the File Specifications Table must be changed during a program. (The use of OPEN for this purpose is described under "Programmed Entries.")

When an OPEN macro-instruction is encountered during the execution of the object program, the following operations will occur for the first reel of each file named in the operand:

1. Variables pertaining to the file will be inserted into the file scheduler routines that were selected during Autocoder assembly. The variables are taken from the File Specifications Table of the tape file. For example, tape unit and channel numbers will be inserted into tape instructions that are included in the selected routines.

2. A 1 will be inserted into the ACTIVITY entry of the File Specifications Table (see Line 06 under "File Specification Entries").

3. A check will be made to determine if a reel of tape and the tape unit specified in the File Specifications Table are available to the program.

4. The tape will be rewound according to the digit inserted on Line 12 of the File Specifications.

5. The tape label will be processed as indicated below.

Label Processing, Input Files: The OPEN macro-instruction will read and check the header tape label;

the items checked are under control of a label mask (see "The DC Entry for Input File Labels").

Label Processing, Output Files: The OPEN macro-instruction will check the retention code of the mounted reel and write a header tape label if the retention code indicates that the tape may be used for writing. The checking of the retention code and the writing of the label are under control of a label mask (see "The DC Entry for Output File Labels").

Other Label Processing: During the running of a program, similar operations are performed for subsequent reels of a multi-reel file after an end-of-reel condition has been detected. An end-of-reel condition is indicated by reading an end-of-reel trailer label for input files or sensing a reflective spot on tape for output files. The operations that occur when an end-of-reel condition is detected depend on whether the file is an input or an output file.

For input files, the operations are as follows:

1. The tape will be rewound according to the digit inserted on Line 30 of the File Specifications. (Rewind does not occur after an end-of-file trailer label is read.)
2. A check will be made to determine if the next reel of the file and its tape unit are available to the program.
3. The tape label of the next reel will be processed as described above for the first reel of an input file.

For output files, the operations that occur when an end-of-reel condition is detected are as follows:

1. A tape mark will be written following the last record on the output tape.
2. The end-of-reel trailer label will be written on the output tape. A tape mark will be written following the trailer label.
3. The tape will be rewound according to the digit inserted on Line 30 of the File Specifications.
4. The tape label of the next reel will be processed as described above for the first reel of an output file.

CLOSE Macro-Instruction

Upon completion of an output file, the tape is removed from use by giving the macro-instruction CLOSE. The CLOSE macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	CLOSE	FILEA, FILEB, FILEC
02			

The operand contains the name of the output file that is no longer to be used; more than one file may be named in the operand provided the file names are separated by commas. These names must be the same as the names used as the operand of the DTF entry of the File Specifications for those files.

When a CLOSE macro-instruction is encountered during the execution of the object program, certain operations will occur for the last reel of each file named in the operand. The operations depend on whether the file is an input file or an output file.

Input Files

1. The tape will be rewound according to the digit inserted on Line 13 of the File Specifications.
2. A 0 will be inserted into the ACTIVITY entry of the File Specifications Table (see Line 06 under "File Specifications Entries").

Output Files

1. Any records which remain in the output areas (i.e., a partly filled block) will be written on the output tape.
2. A tape mark will be written following the last output record.
3. The end-of-file trailer label will be written on the output tape. A tape mark will be written following the trailer tape label.
4. The tape will be rewound according to the digit inserted on Line 13 of the File Specifications.
5. A 0 will be inserted into the ACTIVITY entry of the File Specifications Table (see Line 06 under "File Specifications Entries").

The CLOSE macro-instruction may be used at the end of a program and also whenever data within the File Specifications Table must be changed during the running of a program; the use of the CLOSE macro-instruction when changes are made is described under "Programmed Entries."

END Macro-Instruction

Upon completion of a job, all tapes are removed from use by giving the macro-instruction END. The END macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	END	BADDRESS
02			

The operand may contain an actual or symbolic address or it may be blank. If an address is placed in the operand, the program will branch to that address after the END macro-instruction has been executed. If the operand is blank, the program will enter a loop, so that SPOOL programs (if any) can continue to operate after the END macro-instruction is executed.

When an END macro-instruction is encountered during the execution of the object program, it will perform the same operations as the CLOSE macro-instruction for all files that are still OPEN and will cause an end-of-job message to be typed. The function of the END macro-instruction, after the message has been typed, depends on what is entered in its operand and whether SPOOL programs are to operate in conjunction with the program. One of the following procedures will be followed after the end-of-job message is typed:

1. If an address has been entered in the operand of the END macro-instruction, the program will branch to the instruction located at that address.
2. If the operand of the END macro-instruction is blank and either IGEN3 or IGEN4 is specified in the DIOCS line, the program will type an end-of-job message and come to a programmed halt.
3. If the operand of the END macro-instruction is blank and either IGEN1 or IGEN2 is specified in the DIOCS line, the program will enter a loop to permit SPOOL programs to continue. Within the program loop, a test of a specific word in the SPOOL routine will be made to determine if a new main program is to be loaded from tape. As long as the specific word indicates that no new main program is to be loaded, execution of the program loop will continue to allow operation of the SPOOL program(s). If the word in the SPOOL routine indicates that a new main program is to be loaded from tape, a branch to the load program will occur to load the new main program. The loading of SPOOL programs and main programs is described in the publication IBM 7070 SPOOL System, Form J28-6047-1.

By using the END macro-instruction in place of the final CLOSE, programming is simplified because the programmer need not name the files to be closed. The ability to branch to a specified address or enter a loop provides a means for either initiating the next program or continuing SPOOL programs.

GET Macro-Instruction

To locate a record to be processed, the macro-instruction GET is used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	GET	INPUTFILE
02	ANYLABEL	GET	INPUTFILE TO WORKAREA
03			

The operand indicates the name of the input file containing the desired record; this name must be the name used as the operand of the DTF entry of the File Specifications for that file. In the second form, the portion of the operand following the word TO is the name of a work area for the input file named in the first item of the operand. WORKAREA must be defined by a DA or RDW or DRDW entry. The word TO must be preceded and followed by a single blank character.

The first form of the GET macro-instruction will place the address of the first word of the record into the indexing portion of index word XA specified on Line 19 of the File Specifications for the file named in the operand. Processing of the record may be done using instructions that are indexed by index word XA and that refer to fields within the record as defined by a DA entry relative to 0000. The contents of index words are described under "Use of Tape Index Words." If indexed instructions are to be used and the program is to be assembled with 7070 Autocoder, index word XA may be assigned to the instructions through the DA entry that defines the tape record area (see "Tape Record Areas").

Processing, when using non-indexed instructions, may be done by moving all or part of the record to a work area through the use of a MOVE macro-instruction. The MOVE macro-instruction is described in the Autocoder reference manual. If the program is to be assembled using Four-Tape Autocoder, restrictions stated under "Using IOCS with Four-Tape Autocoder" must be observed.

The second form of the GET macro-instruction will place the address of the first word of the record into the indexing portion of index word XA and will also move the entire record into the area named in the operand (WORKAREA). Processing in the work area may be executed by using non-indexed instructions.

It is imperative that the contents of the index word not be changed, so that subsequent MOVE and PUT operations will operate properly.

An automatic function of the GET macro-instruction is to read a block of records from tape whenever all the records in an input area have been processed. Tape reading is a function of the information supplied in the File Specifications and will occur, as required, when GET macro-instructions are executed.

PUT Macro-Instruction

To cause a processed record to be included in an output file, the macro-instruction PUT is used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	PUT	INPUTFILE IN OUTPUTFILE
02	ANYLABEL	PUT	AREANAME IN OUTPUTFILE
03	ANYLABEL	PUT	FIELDNAME IN OUTPUTFILE
04	ANYLABEL	PUT	OUTPUTFILE
05			

The operand may name a record from another file, an area, or a field that is to be included in an output file; the operand may also indicate that an output record has been assembled in the record area of the output file. Whenever the name of an input file, a field, or an area appears in the operand, the name of the output file must be used following the word IN; the word IN must be preceded and followed by a single blank character. File names used in the operand must be the same as the names used as the operand of the DTF entry of the File Specifications for the corresponding files.

When an entire record from an input file is to be included in an output file, the name of the input file may be used preceding the word IN. Only one input file may be named in each PUT macro-instruction.

An area or a field may form an output record by using the name of the area or field preceding the word IN. The name may be the label of one DRDW that defines the area, or it may be the name of a field defined by a subsequent entry under a DA or DC entry.

The last example shows the type of PUT macro-instruction to be used when processing is to be done in an output area. The index words for the output file will be changed to define the next space in the output area. Processing of the various record forms in an output area is described under "Processing in Output Areas."

The PUT macro-instruction causes data in an input area or a storage area to be moved to the next available space in an output area. When using this macro-instruction, the programmer must be sure that the data moved into the output area will result in the same record form as specified in the File Specifications (Line 08) for the output file.

An automatic function of the PUT macro-instruction is to write a block of records onto tape whenever enough records have been processed to make up an output block. Tape writing is a function of the information supplied in the File Specifications and will occur as required.

PUTX Macro-Instruction

When records are to be placed in an output file by exchanging RDWs rather than by moving the record, the PUTX macro-instruction must be used. The operation code PUTX is derived from the words Put and Exchange. The macro-instruction should be written as follows:

Sequence (Pg) (Lin) 1 2 3 5 6	Name (Label) 15	Operation Code 16 20	OPERAND 21 25 30 35 40 45
01	ANYLABEL	PUTX	INPUTFILE IN OUTPUTFILE
02			

The first name in the operand is the name of the input file; the second is the name of the output file. The names used in the operand must be the same as the names used as the operand of the DTF entry of the File Specifications for the corresponding files. The word IN must be preceded and followed by a single blank character.

The form of the records in the files places the following restrictions on the use of the PUTX macro-instruction:

1. Form 3 records can not be processed with a PUTX macro-instruction.

2. The combination of input record form and output record form must be one of the following:

Input File Record Form	Output File Record Form
1	1
1	2
2	1
2	2
4	4

3. The length of fixed-length records or the maximum length of variable-length records must be identical for both the input and output files.

4. For Form 4 records, the number of sections in input and output records must be the same and the maximum number of words in the corresponding sections of each input and output record must be identical.

There is no restriction on the blocking factor of the input and output files. The blocking factor (the number of records in one block) may be different for each of the files provided that the other restrictions listed above are observed.

After RDWs have been exchanged by a PUTX macro-instruction, the input record is no longer available for processing; the programmer must be certain that all processing that requires the input data is completed before issuing the command.

When PUTX is to be used, processing should be performed in the input area using indexed instructions that refer to fields within the record as defined by a DA entry relative to 0000. If input data is moved to a work area for processing, do not use PUTX. If the PUTX macro-instruction is used, the original input data rather than the results of processing will appear in the output file.

The automatic function of writing blocks of records on tape is the same for the PUTX macro-instruction as for the PUT macro-instruction.

RLSE (Release) Macro-Instruction

To begin using a new record block, the macro-instruction RLSE can be used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	RLSE	FILENAME
02			

The operand contains the name of the tape file in which a new block of records is to be started; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

If the file named in the operand is an input file, no records will be taken from the current block after the RLSE macro-instruction is executed. The next GET macro-instruction that refers to the file will obtain the first record in the next tape block. When used with input files, RLSE allows the programmer to bypass or delete the records remaining in a block of records.

If the file named in the operand of a RLSE macro-instruction is an output file, no more records will be entered into the current block after the RLSE instruction is executed. The next PUT macro-instruction causes a record to be entered as the first record in a new block. If a block is partially filled when the RLSE is executed, the block will be included in the output file as a short-length block. By using the RLSE macro-instruction with output files, the programmer may select the record which is to be the first record in a tape block. Except in certain cases, the RLSE macro-instruction must be executed prior to executing any of the following macro-instructions:

WTM WSM BSP RWD RDSF RDSB

A RLSE instruction should not be given if either of these conditions is present:

1. No GET, PUT, or PUTX macro-instructions have been given on the file.

2. The file is an input file that has a blocking factor of 1 and uses one area.

The operand of the RLSE macro-instruction must name the same file as the macro-instruction to be executed. For example, RLSE FILEA must be executed before WSM FILEA can be executed. Several macro-instructions in the above list, which refer to the same file, may be executed following a single RLSE, provided that no other macro-instructions (not in the list) referring to that file intervene between the RLSE and the listed macro-instructions.

RDLIN (Read Label Information) Macro-Instruction

Data against which labels of input files are to be checked can be entered into storage by using the RDLIN macro-instruction. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	RDLINS	
02			

The S in the operand is a digit from 1 through 4 to specify the card reader synchronizer to be used to read the Input Label Information Cards (see "Input Label Information Card"). If an IBM 7500 Card Reader is to be used, 1, 2, or 3 must be placed in the operand to specify the unit record synchronizer to which the card reader is connected. If an IBM 7501 Console Card Reader is to be used, the operand must contain a 4.

The RDLIN macro-instruction will read data that has been punched into Input Label Information Cards and store it in the alphameric portion of the input label DC entry (see "The DC Entry for Input File Labels") of the respective input files.

Only non-zero fields in an Input Label Information Card will be stored in the DC entry; zero fields will have no effect on the corresponding portion of the DC entry. Reading of Input Label Information Cards will continue until the RDLIN macro-instruction reads a Termination Card and then execution of the next instruction in the program will occur.

Before the RDLIN macro-instruction is executed, the tape channel and tape unit number of the first reel of the input file(s) must have been established in the DTF entry of the file(s). At the time the RDLIN macro-instruction is executed, an Input Label Information Card must be the next card to be read from the card reader specified in the operand.

WTM (Write Tape Mark) Macro-Instruction

To write one or more tape marks on an output file, the macro-instruction WTM can be used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	WTM	FILENAME,N
02	ANYLABEL	WTM	FILENAME
03			

The first item in the operand is the name of the output file on which one or more tape marks are to be written; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

The second item (N) in the operand may have any value from 1 to 9999 to specify the number of tape marks that are to be written. If the operand of this macro-instruction contains only the name of an output file (i.e., if the comma and the value of N are omitted), one tape mark will be written on the output file.

The WTM macro-instruction may require that a RLSE macro-instruction that refers to the same file be written immediately preceding the WTM. Use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

WSM (Write Segment Mark) Macro-Instruction

To write one or more segment marks on an output file, the macro-instruction WSM is used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	WSM	FILENAME,N
02	ANYLABEL	WSM	FILENAME
03			

The first item in the operand is the name of the output file on which segment marks are to be written; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

The second item (N) in the operand may have any value from 1 to 9999 to specify the number of segment marks that are to be written. If the operand of this macro-instruction contains only the name of an output file (i.e., if the comma and the value of N are omitted), one segment mark will be written on the output file.

The WSM macro-instruction may require that a RLSE macro-instruction that refers to the same file be written immediately preceding the WSM; use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

BSP (Backspace) Macro-Instruction

To backspace over one or more tape records in either an input or an output file, the macro-instruction BSP can be used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	BSP	FILENAME,N
02	ANYLABEL	BSP	FILENAME
03			

The first item in the operand is the name of the tape file that is to be backspaced. This name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

The second item (N) in the operand may have any value from 1 to 9999 to specify the number of tape records to be backspaced over. If the operand of this macro-instruction contains only the name of a file (i.e., if the comma and the value of N are omitted), the tape will be backspaced over one tape record.

The BSP macro-instruction may require that a RLSE macro-instruction that refers to the same file be written immediately preceding the BSP; use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

RWD (Rewind) Macro-Instruction

To rewind either an input or an output file, the macro-instruction RWD is used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	RWD	FILENAME
02			

The first item in the operand is the name of the tape file that is to be rewound; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for the file.

The RWD macro-instruction may require that a RLSE macro-instruction that refers to the same file be written immediately preceding the RWD; use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

RDSF (Read Segment Marks Forward) Macro-Instruction

To space an input tape forward over a specific number of segment marks, the macro-instruction RDSF is used. The macro-instruction should be written as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	RDSF	FILENAME,N
02	ANYLABEL	RDSF	FILENAME
03			

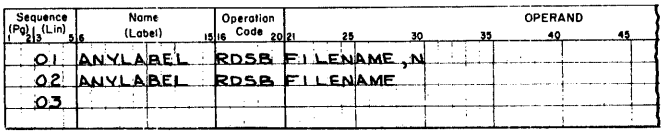
The first item in the operand is the name of the input file that is to be spaced forward; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for the file.

The second item (N) in the operand may have any value from 1 to 9999 to specify the number of segment marks to be spaced over. If the operand of this macro-instruction contains only the name of an input file (i.e., if the comma and the value of N are omitted), the tape will be spaced forward to the next segment mark. The RDSF macro-instruction will not change the block count.

The RDSF macro-instruction may require that a RLSE macro-instruction which refers to the same file be written immediately preceding the RDSF; use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

RDSB (Read Segment Marks Backward) Macro-Instruction

To space an input tape backward over a specific number of segment marks, the macro-instruction RDSB can be used. The macro-instruction should be written as follows:



The first item in the operand is the name of the input file that is to be spaced backward; this name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

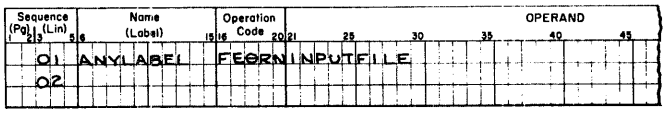
The second item (N) in the operand may have any value from 1 to 9999 to specify the number of segment marks to be spaced over. If the operand of this macro-instruction contains only the name of an input file (i.e., if the comma and the value of N are omitted), the tape will be spaced backward to the next segment mark.

The RDSB macro-instruction will not change the current block count. The RDSB macro-instruction may require that a RLSE macro-instruction that refers to the same file be written immediately preceding the RDSB; use of the RLSE macro-instruction is explained under "RLSE (Release) Macro-Instruction."

FEORN (Force End-of-Reel on an Input Tape) Macro-Instruction

To force end of reel on an input tape, the macro-instruction FEORN is used. This statement is useful if it is desired to read only a part of a reel of an input file and then to begin reading the next reel of the file.

The macro-instruction should be written as follows:

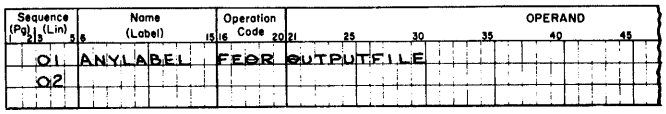


The operand contains the name of the input file which must be the same as the name used as the operand of the DTF entry of the File Specifications for that file. When the FEORN macro-instruction is given, the end-of-reel routine is entered for the input file. There is no trailer label checking and the user's end-of-file exit and trailer label exits are bypassed. The user's end-of-reel exit and header label exits will be entered if specified. The tape is rewound according to the rewind procedure specified. If an alternate tape unit is specified, the program flips to the new unit for the next reel of the file and the flip-flop message is typed. If labels are specified, the header label is read and checked according to specifications. It is necessary, prior to the use of this macro-instruction, to give a RLSE macro-instruction in which the operand contains the name of the input file.

FEOR (Force End-of-Reel for an Output File) Macro-Instruction

To force end of reel for an output file, the macro-instruction FEOR is used. This statement is especially useful when it is desired to write on only part of a reel and then continue writing on the next reel.

The macro-instruction should be written as follows:

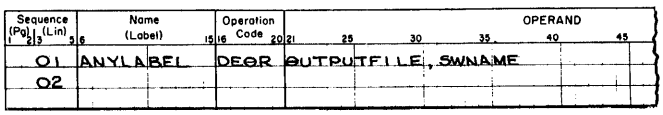


The operand contains the name of an output file. This name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file. The macro-instruction causes the end-of-reel routine to be entered for this output file. A trailer label (if labels are specified) and a tape mark are written starting at the point in the tape at which end of reel is forced. The current output reel and the new reel (if any) are rewound according to the rewind procedure specifications. If an alternate tape unit is specified, the program flips to the new unit for the next reel of the file and the flip-flop message is typed. If labels are specified, the header label is read and checked according to specifications.

A RLSE OUTPUTFILE instruction should not be given after this instruction until another PUT or PUTX statement has been executed. A RLSE OUTPUTFILE should not be given before a FEOR statement.

DEOR (Delay End-of-Reel on an Output File) Macro-Instruction

The DEOR macro-instruction is used to delay end of reel on an output file when the reflective spot has been reached, and to continue writing on the output tape. The DEOR macro-instruction should be written as follows:



The first item in the operand is the name of the output file. This name must be the same as the name used as the operand of the DTF entry of the File Specifications for that file.

The second item in the operand is the name of an electronic switch and can be actual or symbolic. This macro-instruction can be used in conjunction with the FEOR macro-instruction so that once the desired information has been written after the reflective spot, the end-of-reel condition can be forced on the output tape. This macro-instruction should be given in the initialization or housekeeping routines.

When the macro-instruction is given, the electronic switch specified in the instruction is turned off. Then the file scheduler is adjusted so that the end-of-reel routine will not be entered when the reflective spot is reached. Instead, a routine will be entered that will treat the unusual condition as a normal condition and turn on the electronic switch. At the point where the programmer wishes to stop writing on the tape, he can force an end of reel on the tape. This is done through the use of a BSF instruction, as illustrated below:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01		FEOR	OUTPUTFILE, SWNAME
02		•	
03		•	
04		•	
05		BSF	SWNAME, FORCE
06		•	
07		•	
08	FORCE	FEOR	OUTPUTFILE
09			

The user must be certain that he does not attempt to write too much after the reflective spot. An earlier positioning of the reflective spot is suggested to correct any difficulty that may arise in this connection.

PROCESSING OF LABELS BY IOCS

The reading, checking, preparing, and writing of the standard header and trailer labels described under "Tape Labels" are performed by IOCS using information contained in the File Specifications and in the DC entry (see below) for the file. These operations are performed for each reel of a multireel file.

Discrepancies encountered while checking labels will be indicated by typewriter messages that show what label data was expected and what label data was read. After a label error message has been typed, a halt will occur to allow the operator to decide what is to be done. The operator may elect to use the reel of tape with the discrepancies or to replace the reel with another reel and repeat the label checking.

Before writing on a reel of tape, a check of the header label can be made to determine whether a specified number of days (the retention cycle) has elapsed since the date the header label was written (the creation date). To perform this check, the date on which the program is being run (the current date) must be available to IOCS. The current date must be placed in storage location 0109 in the form +YYDDD00000 where YY is the tens and units position of the year and DDD is the number of the day within the year. Any method of storing the current date into location 0109 may be used. The two methods that will probably be used most frequently are (1) manually storing the date from the console and (2) punching the date into a load card. If location 0109 is not used by programs, the current date can be stored once at the beginning of a day and left there for use by all programs.

The current date is also used as the creation date in header labels written on output files. Therefore, when tape labels are written, location 0109 must contain the current date even though the user chooses not to check the retention cycle of input file labels.

Input/Output Label Area

IOCS establishes one label area that is used for reading and writing all tape labels. This area, called the Input/Output Label Area, occupies a total of seventeen storage locations. The first word of the area is the RDW for the Input/Output Label Area. The symbolic location of the RDW is IOCSLBAREA.

To refer to fields within a label record, the programmer may use address arithmetic. For example, the last word in the Input/Output Label Area can be referred to as IOCSLBAREA+16. The location of label fields in the Input/Output Label Area is listed below for the standard header and trailer records. Additional header or trailer records may be arranged as required by the programmer.

Header Label: When in the Input/Output Label Area, the label fields will be located in the following words. The description of these fields is given under "Explanation of the Header Label."

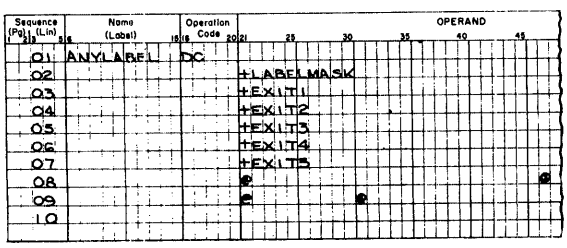
Word Number	Contents	Data Format
1	The RDW for the Input/Output Label Area	
2	Field 1, Label Identifier	1HDRb
3	Field 2, Tape Serial Number	xxxxx
4	Field 2, File Serial Number	xxxxx
5	Field 2, Reel Sequence Number	-xxxb
6 and 7	Field 3, File Identification	AAAAAAAAAA
8	Field 4, Creation Date	xxxxx
9	Field 4, Retention Cycle	-xxxb
10 through 17	These words are not used for standard label information and may be used by the programmer.	

Trailer Label: When in the Input/Output Label Area, the label fields will be located in the following words. The description of these fields is given under "Format for the Trailer Label."

Word Number	Contents	Data Format
1	The RDW for the Input/Output Label Area	
2	Field 1, Label Identifier	1 EORb or 1EOFb
3	Field 2, Block Count	xxxxx
4 through 17	These words are not used for standard label information and may be used by the programmer. Some programs written by the IBM Programming Systems Department use words 4-5 and/or 6-7 for record count and hash total, respectively.	

DC Entry for Output File Labels

Each output file to be produced requires a DC entry to specify the label data and optional program exits to be used. The form of the DC entry is as follows:



Each line is described separately below using the line numbers shown for reference purposes. Note that the line numbers for the entries do not have to be as shown on the coding sheet, provided entries are made on consecutive lines in the order given. Not all of the lines shown may be required for each DC entry. For example, if only exit 3 is to be used, the other exits may be omitted and the DC entry written on five consecutive lines.

Line 01: The label of the DC entry may be any label that the programmer desires. This label is to be entered into the operand of Line 32 of the File Specifications for the corresponding tape file.

The operand of the DC entry must be blank.

Line 02: The operand consists of a plus sign followed by a ten-digit label mask that specifies the label items to be written and the label routine program exits to be used. The significance of each digit of the label mask is as follows:

Digit 0. A 0 in this position indicates that (1) header labels, if any, on reels to be used for output are not to be read and (2) labels will be written using data in this DC entry (see Line 08). Digits 1 to 4 of the label mask will be ignored when this digit is a zero. A 1 in this position indicates that writing and checking of tape labels is to be determined by digits 1 to 4 of the label mask.

Digit 1. A 1 in this position indicates that (1) the header labels on reels to be used for output will be read (2) the tape serial number (word 3) will be stored in the tape serial number portion of this DC entry (see Line 08) and (3) the creation DATE and retention cycle will be checked to see if the tape may be used for output. A 0 in this position indicates that none of these actions are to be performed. If digit 0 is a 0, digit 1 is ignored.

Digit 2. The function of a 1 in this position depends on the value of digit 1 as follows:

If digit 1 is a 1, the file serial number of the output file will be determined by the tape serial number of the header label of the reel to be used for output.

If digit 1 is a 0, the file serial number of the output file will be taken from the tape serial number portion of this DC entry (see Line 08).

A 0 in this position indicates that the file serial number placed in the file serial number portion of this DC entry will not be affected by the header label read. If digit 0 is a 0, digit 2 is ignored.

Digit 3. A 1 in this position indicates that reel sequence numbers for reels of the output file will be 001, 002, 003, etc. A 0 indicates that the reel sequence number for all reels of the output file will be taken from the reel sequence number portion of this DC entry (see Line 08). If digit 0 is a 0, digit 3 is ignored.

Digit 4. A 1 in this position indicates that the creation date for the output file is to be taken from storage location 0109 and stored in the creation date portion of this DC entry (see Line 08). A 0 indicates that the creation date in this DC entry is not to be changed. If digit 0 is a 0, digit 4 is ignored.

Digit 5. A 1 in this position indicates that label routine program exit 5 (see Line 07 below) can be used during the program. A 0 indicates that exit 5 will not be used.

Digit 6. A 1 indicates that label routine program exit 4 (see Line 06 below) can be used during the program. A 0 indicates that exit 4 will not be used.

Digit 7. A 1 indicates that label routine program exit 3 (see Line 05 below) can be used during the program. A 0 indicates that exit 3 will not be used.

Digit 8. A 1 indicates that label routine program exit 2 (see Line 04 below) can be used during the program. A 0 in this position indicates that exit 2 will not be used.

Digit 9. A 1 indicates that label routine program exit 1 (see Line 03 below) can be used during the program. A 0 indicates that exit 1 will not be used.

Line 03: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to be entered after IOCS has processed the standard trailer label of the output tape. This exit occurs after the end-of-reel indicator and the block count (Fields 1 and 2) have been placed in the Input/Output Label Area. The user's routine may insert additional data into the unused fields of the standard trailer record by placing the data in the Input/Output Label Area.

If a symbolic address is used in the operand, the symbol must be the same as the label of the first instruction in the routine that performs the additional processing. If digit 9 in the label mask is a 0, this line must be omitted.

Line 04: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to be entered to process any additional trailer records that the user intends to write on the output tapes. This exit occurs after the standard trailer record has been written. The user's routine may be used to assemble and write additional trailer records.

If a symbolic address is used, the symbol must be the same as the label of the first instruction in the routine for processing the additional trailer records. If digit 8 in the label mask is a 0, this line must be omitted.

Line 05: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to be entered after IOCS has read the standard header label written on an output tape. This header label would have been written previously (e.g., by another program). The exit occurs after the header label has been read into the Input/Output Label Area. The user's routine may obtain data from that area to do any processing that is required.

If the location of the routine is specified by a symbolic address, the symbol must be the same as the label of the first instruction in the routine that performs the additional processing. If digit 7 in the label mask is a 0, this line may be omitted,

Line 06: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to be entered to perform additional processing on a standard header label that is to be written. This exit occurs after the retention code has been checked, the tape has been rewound and the standard header label information has been placed in the Input/Output Label Area. The user's routine may insert additional data into unused fields of the standard header label by placing the data in the Input/Output Label Area. When EOR3 has been specified in the DIOCS operand, the user's routine may also type any portion of the output file header label.

If a symbolic address is used, the symbol must be the same as the label of the first instruction in the routine that performs the additional processing.

If digit 6 in the label mask is 0, this line must be omitted.

Line 07: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to be entered to process any additional header records that the user intends to write on the output tapes. This exit occurs after the standard header label has been written. The user's routine may be used to assemble and write additional header labels.

If a symbolic address is used, the symbol must be the same as the label of the first instruction in the routine that processes the additional header records. If digit 5 in the label mask is a 0, this line must be omitted.

Line 08: Beginning with this line, from 35 through 75 alphameric positions must be defined. Although the coding sheet on page 35 shows the minimum number of positions defined on two lines, the number of positions per line and the number of lines used is irrelevant. The positions that must be defined correspond to words 3 through 9 of a header label as it appears in the Input/Output Label Area; optional positions (words 10 through 17) need be defined only when they are to be included in the header label. Data entered on these lines must conform to the format listed under "Header Label." For example, the retention cycle in the form -xxxb would be entered in word 9. Using the coding sheet on page 35, this would appear in columns 27 through 31 of Line 09.

The last word of the DC entry to be included in a header label must be followed by a numeric word because IOCS forms an output header label by moving words into the Input/Output Label Area. Movement of data begins with the first alpha word and continues until a numeric word occurs; the unused words in the header label will be set to blanks automatically. The need for a numeric word after the data for the tape label may be satisfied in any manner the programmer chooses. For example, the next item in the program may be:

1. The DC entry for label data of another file.
2. A numerical constant.
3. A DA entry which generates RDWs.
4. An instruction.

Additional Output Label Processing

When additional header or trailer labels are required on output files, the IOCS label-writing routine may be used as a very simple method for writing the additional labels, with the following restrictions:

1. All additional labels must be 16 words in length; alpha, numeric, or a combination of both.
2. If checkpoint is used by the program, all additional header labels must have xHDRx as the first word. The letter x represents any character.

The coding for writing one additional label is illustrated below.

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ADDLOUTLBL	XZA	XX,ADDLLABL
02		RS	XX,I@CSLBAREA
03		BLX	99,I@CSWRLABL
04		B	0+I@CSIXF
05			

The first two instructions are used to move the additional label to the IOCS label area. Any set of instructions may be used to accomplish the data movement. The third instruction is required to branch, and return from, the label-writing routine. Index word 99 must be used for this purpose. The last instruction is the return to IOCS. Any number of additional labels may be written by repeating the first three instructions, one set for each additional label, with the final instruction being a Branch to 0+IOCSIXF to return to IOCS.

DC Entry for Input File Labels

Each input file to be read requires a DC entry to specify the label data to be checked and the optional program exits to be used. The form of the DC entry is as follows:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	DC	
02			+LABELMASK
03			+EXITG
04			+EXIT7
05			@AAAAAAAAAAFFFFF-RRR CCCCC@
06			

Each line is described separately below, using the line number shown for reference purposes. Note that the line numbers for the entries do not have to be as shown on the coding sheet provided entries are made on consecutive lines in the order given. All of the lines shown may not be required for each DC entry. For example, if no exits are to be used, they may be omitted, and the DC entry written on three consecutive lines.

Line 01: The label of the DC entry may be any label that the programmer desires. This label is to be entered in the operand of Line 32 of the File Specifications for the corresponding tape file.

The operand of the DC entry must be blank.

Line 02: The operand consists of a plus sign followed by a ten-digit label mask that specifies the label items to be checked and the label routine program exits to be used. The significance of each digit of the label mask is as follows:

Digit 0. A 1 in this position indicates that tape labels for the input file are to be checked. A 0 indicates that no tape labels are to be checked.

Digit 1. A 0 indicates that the identification portion of the tape label is not to be checked. A 1 indicates that the identification portion of the tape label is to be checked. If digit 0 is a 0, digit 1 is ignored.

Digit 2. A 0 in this position indicates that the file serial number portion of the tape label is not to be checked. A 1 indicates that the serial number is to be checked. If digit 0 is a 0, digit 2 is ignored.

Digit 3. A 0 indicates that the reel sequence number portion of the tape label is not to be checked. A 1 indicates that the reel sequence number is to be checked. If digit 0 is a 0, digit 3 is ignored.

Digit 4. A 0 in this position indicates that the creation date portion of the tape label is not to be checked. A 1 indicates that the creation date is to be checked. If digit 0 is a 0, digit 4 is ignored.

Digit 5-7. These digits must contain zeros.

Digit 8. A 1 in this position indicates that label routine program exit 7 (see Line 04 below) can be used during the program. A 0 indicates that exit 7 will not be used.

Digit 9. A 1 indicates that label routine program exit 6 (see Line 03 below) can be used during the program. A 0 indicates that exit 6 will not be used.

Line 03: The operand consists of a plus sign followed by an actual or symbolic address. This address specifies the location of a routine to perform additional processing of data read from the trailer record of the input file. This exit occurs after the block count in the trailer record has been compared with the block count maintained by IOCS; the block count or a message that the counts do not agree will have been typed. The user's routine may be used for processing additional data in the standard trailer record, which is located in the Input/Output Label Area. The routine may also be used to read and process additional trailer records that may be on the tape.

If a symbolic address is used in the operand, the symbol must be the same as the label of the first instruction in the routine for the additional processing. If digit 9 in the label mask is a 0, this line must be omitted.

Line 04: The operand consists of a plus sign followed by an actual or symbolic address that specifies the location of a routine to perform additional processing of data read from the input file header label. This exit occurs after the end-of-reel exit (see Line 28 under "File Specification Entries") and after the data in the standard header label has been checked as controlled by the label mask on Line 02. The user's routine may be used for processing additional data in the standard header label.

If a symbolic address is used in the operand, the symbol must be the same as the label of the first instruction in the routine for the additional processing. If digit 8 in the label mask is 0, this line must be omitted.

Line 05: Beginning with this line, 25 alphameric positions must be defined. Although the coding sheet on page 37 shows these positions defined on one line, the number of positions defined per line and the number of lines used is irrelevant provided that the total of 25 is defined. These positions correspond to fields in the header label as follows:

AAAAAAAAAA is the file identification
FFFFF is the file serial number
-RRRb is the reel sequence number
CCCCC is the creation date

Information that is to be checked against all header labels of this input file must either be entered on this line(s) when the DC entry is written or entered from an Input Label Information Card (see below) by using the RDLIN macro-instruction. Data entered on this line(s) must conform to the format listed under "Header Label." Fields that are not to be checked against the tape labels may be blank. For example, if digit 0 of the label mask is a zero, the @ characters may be placed to indicate 25 blank characters; using one line, this would require a @ character in columns 21 and 47.

Additional Input Label Processing

When additional header or trailer labels are present on input files, the IOCS label read-in routine may be used as a very simple method for reading in the additional labels, with the following restrictions:

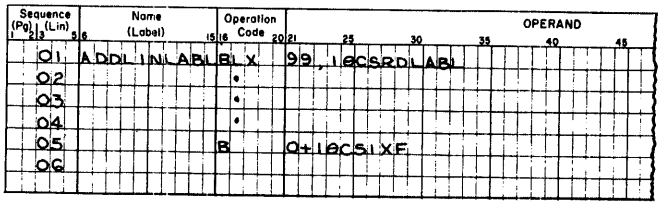
1. All additional labels must be 16 words in length; alpha, numeric, or a combination of both.

2. If checkpoint is used by the program, all additional header labels must have xHDRx as the first word. The x represents any character.

The coding for reading one additional label is illustrated below. For simplicity, the instructions for processing data within the labels are represented by a series of dots.

The first instruction is used to branch to the label read-in routine in order to read one label and return. Index word 99 must be used for this purpose.

The instructions for processing the label would follow the BLX instruction. The last instruction is the return to IOCS.



Any number of additional labels may be read by repeating the BLX instruction and processing instructions for each additional label, with the final instruction being a Branch to 0+IOCSIXF.

Input Label Information Card

To change or insert data in the alphameric portion of the label DC entry of an input file, an Input Label Information Card may be prepared and read by the RDLIN macro-instruction. The Input Label Information Card need specify only those fields that are to be inserted or changed; fields that contain all zeros will have no effect on the DC entry. These cards will be loaded from the card reader indicated in the RDLIN macro-instruction, and must be followed by a Termination Card (see below).

An Input Label Information Card is to be punched as follows and must conform to the Load Card Format:

Column(s)	Contents
1-5	The current date in the form YYDDD where YY is the units and tens position of the year and DDD is the number of the day within the year. This date is checked against the date stored in location 0109; if the dates do not agree, a programmed halt will occur.
6	The tape channel number to be used for reading the input file.
7	The number of the tape unit number on which the first reel of the file is mounted; this number will be the same as the operand of Line 03 of the File Specifications for the file.
8-10	The reel sequence number to be used for the first reel of the input file. If these columns contain 000, the reel sequence number portion of the label DC entry will not be changed.
11-15	The creation date of the input file. If these columns contain 00000, the creation date portion of the label DC entry will not be changed.
16-20	The file serial number of the input file. If these columns contain 00000, the file serial number portion of the label DC entry will not be changed.
21-40	The file identification of the input file punched in double-digit form. If this field contains all zeros, the file identification portion of the label DC entry will not be changed.
41-80	The contents of these columns is irrelevant but all columns must be punched to conform to the Load Card Format.

Termination Card

To indicate that all Input Label Information Cards have been read, a Termination Card must follow the last Input Label Information Card. The Termination Card must be punched as follows and must conform to the Load Card Format:

Column(s)	Contents
1-5	The current date in the form YYDDD; see columns 1-5 under Input Label Information Card for an explanation of YYDDD.
6-7	Zeros.
8-80	The contents of these columns is irrelevant but all columns must be punched to conform to the Load Card Format.

END-OF-REEL ROUTINE

The function of the end-of-reel routine is to process all header and trailer labels and to write end-of-file records (tape marks). It will read and check all header and trailer labels written on output tapes (if the user specifies), read and check header labels written on output tapes (if the user specifies), and write new header and trailer labels on output tapes (when specified). Label processing is important. For example, it can prevent the destruction of valuable information stored on tapes. Also, information in the labels can be checked to make certain that all tape records have been read. The following is a description, including flow charts, of the EOR routine.

Operational Description of the Routine

Block 225: The DTF address of the file being processed is set into the indexing position of index work 97. The non-indexing position of word 97 is set to 0000 for an input file or 9999 for an output file. This information will be used to obtain specifications from the DTFs and act as a switch to differentiate between input or output files in the EOR procedure. This block also sets up the user's exits for special EOR procedures by determining whether these exits (NOP) will be set to branches. This is done by checking the exit mask in the label DC entry. The user, on completion of his routine, after using any exit, may return to the EOR routine with a Branch to 0+IOCSIXF.

Block 226, IOCSOPNSW1: A branch around trailer operations and between-reel functions will be made to block 244 if the EOR routine was entered from the OPEN routine to process a header label.

Block 227: This block starts the processing of a trailer. It turns off the EOF indicator and determines if the file is an input or output file. If it is an output file, a branch is made to block 233.

Block 228: The block counter is reduced by 1 to prevent the tape mark from being counted as a block. The statistics are typed for the tape. The LABELINF of the DTF is then tested. If it is 0000, indicating no label, a BZ1 is made to block 232.

Block 229: This block reads the label and compares the block count in the DTF to that in the trailer. If they are equal, a branch is made to block 231.

Block 230: A message indicating an incorrect block count is made, followed by a halt.

Block 231, IOCSEX6: This block is an exit available to the user so that he may perform additional checking of the trailer.

Block 232, IOCSEOFEX: The trailer record is tested for EOF. If EOF is indicated (label word 2 contains EOF), a branch is made to the user's EOF routine at the address he has specified in EOFPROCD of the DTF. If no labels are specified, this exit is used for all end-of-reel conditions for input files.

Block 233, IOCSLBTM: This block starts output trailer procedures by writing a tape mark, typing trailer statistics, and checking for a trailer.

Block 234, IOCSEX1: This block completes the trailer record and provides an exit for the user to add additional information to other trailers.

Block 235: The trailer is written on the tape.

Block 236, IOCSEX2: This block provides an exit to allow the user to write additional trailers.

Block 237: A tape mark is written after the trailer or trailers.

Block 238, IOCSICEST: An exit to the close routine is made if the EOR routine was entered on a close operation.

Block 240: This block is entered only as a result of an end-of-reel condition. It will restore the BLOCKCNT field of the DTF to zero and rewind the tape as directed by the RWDPROCD digit of the DTF.

Block 241: The tape unit addresses are rotated in BASETAPE, ALT1TAPE, and ALT2TAPE, of the DTF for the file.

Block 242, IOCSEOREX: This block provides an exit for the user to alter the tape addresses in the DTF if he desires other than normal operation, such as using two files for three tape units. A check is made to determine whether an end-of-reel exit is specified. If not, this block is bypassed.

Block 243: The appropriate message concerning the tape address changes is typed and a halt is made if a manual tape change is necessary.

Block 244: The proper channel and tape addresses for this file are placed in the remaining EOR instructions.

Block 245, IOCSERSEL: The tape is tested for ready. A Branch is made to block 246 if not ready.

Block 246: The not-ready message is typed and a return is made to block 245, where the machine will hang until the tape is readied.

Block 247, IOCSOPNSW3: This block is a switch that returns control to the OPEN routine (during an OPEN operation) to rewind the tape according to OPENPROC in the DTF. After this operation, a return is made to the EOR routine to block 249.

Block 248, IOCSRWNRL: The tape is rewound as specified by the normal EOR specification code located in RWDPROCD of the DTF.

Block 249: The header density is set as specified by the TDENSITY in the DTF for the file.

Block 255: A BCX determines if labels are to be processed and, if not, a branch is made to block 270.

Block 256: The file is tested for input or output and, if output, a branch is made to block 262.

Block 257: The header label for the input file is read.

Block 258, IOCSCKLOOP: The label mask is tested and the header is checked against specifications.

Block 259: The label is tested for error and, if in error, control passes to block 260.

Block 260: This block types the error message and halts.

Block 261, IOCSEX7: A user's exit is provided so that additional input header information may be processed.

Block 262, IOCSLOPRO: The header of this output file is read, if desired. If it is not desired, control passes to block 266.

Block 263: The retention cycle is tested to determine if this tape may be written on. If the current date stored in word 109 is within the retention cycle, a branch is made to block 264.

Block 264: The error message is typed to direct the operator to check the retention date, and a halt occurs.

Block 265, IOCSEX3: This block provides an exit for the user to perform any other header checks he may want.

Block 266: The label mask is tested, the tape is backspaced, and a new label is set up if specified.

Block 267, IOCSEX4: This block provides an exit for the user to add label information if he wishes.

Block 268: The new label is written and the label statistics are typed.

Block 269: A user's exit is provided to write other labels if he desires.

Block 270, IOCSFLDN: The file density is set according to the control digit of the file's TDENSITY in the DTF.

Block 271: This block represents a NOP operation that is initialized by the initialization and assignment section of the checkpoint routine. The instruction is altered into a Branch to IOCSCPEOR where a decision to take a checkpoint is made.

Block 272, IOCSOPNSW2: A return to the OPEN routine is made if the EOR routine was entered during an OPEN operation.

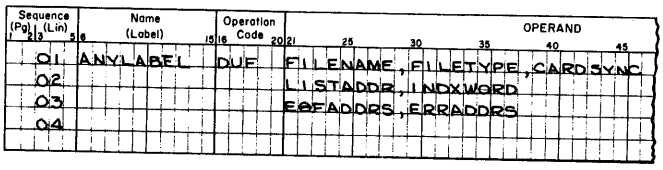
Block 273: A test is made to determine if the file is input or output, to enable a return to the proper place in the condition code routine.

DESCRIPTIVE ENTRY FOR UNIT RECORDS (DUF)

When unit record files are to be handled by the IOCS, the programmer must supply a DUF (Define Unit Record File) entry that describes the type of file and the unit record equipment to be used. The DUF entry also supplies the locations of subroutines written by the user and unique to the file. A routine will be generated from the DUF descriptive entry for each unit record file; the routine will be used during processing of records from the corresponding file.

The DUF entries are to be punched into Autocoder cards from the coding sheet. These cards are to be entered with the source program when the program is assembled. The items in the DUF entry must be entered in the same sequence as shown on the coding sheet.

From five through seven items may be named in the operand of each entry. The DUF entry is written on one or more lines of the coding sheet as follows:



The letters DUF must be placed in the operation column as shown to identify the unit record descriptive entry.

Each item in the operand is described separately below. The first five items must always appear in the operand of a DUF entry. The last two items need be included only when they are required.

The first item in the operand (FILENAME) is the name of the unit record file to be described by the DUF entry. This name will be used in the operand of macro-instructions that refer to this unit record file.

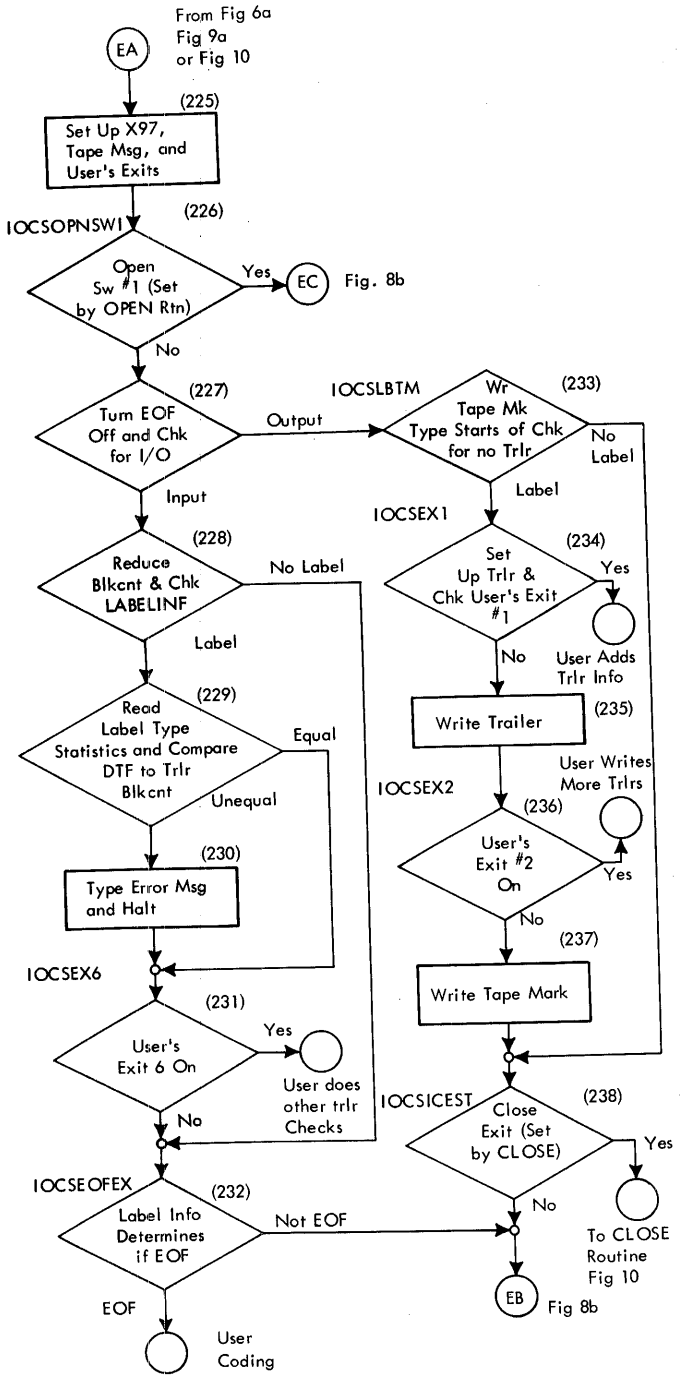


Figure 4

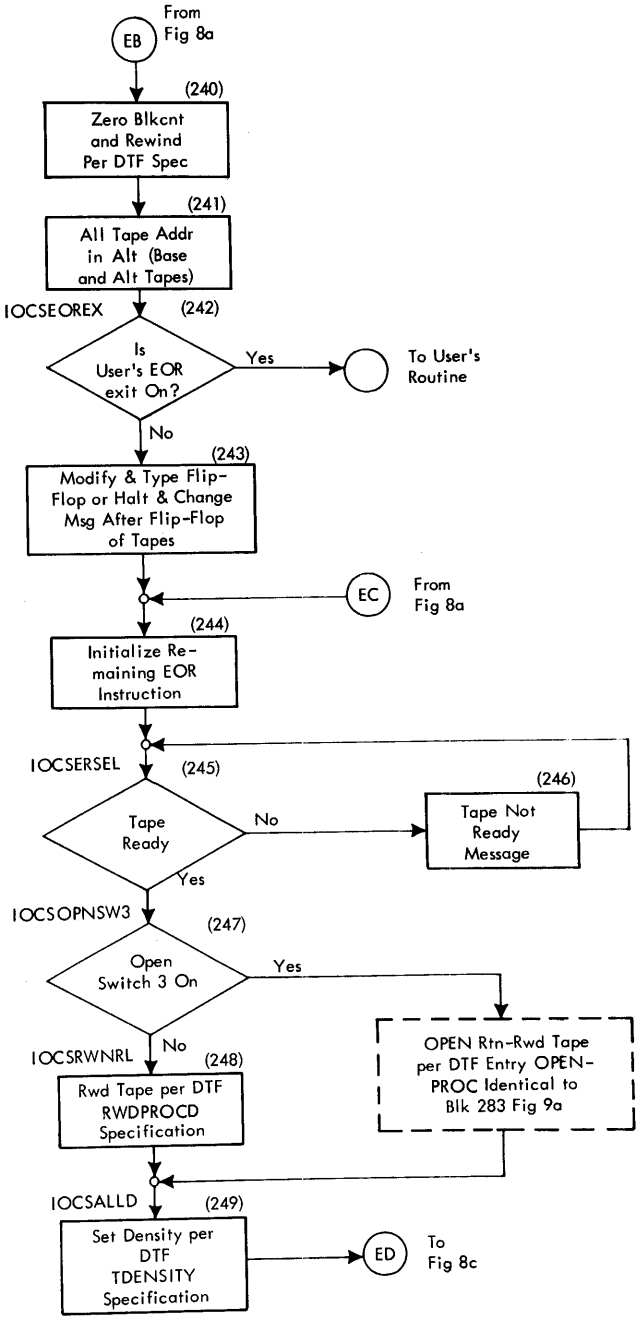


Figure 5

The flow charts shown in Figures 4 and 5 are taken from from Programming Systems Analysis Guide, 7070/7074 IOCS, Form C28-6119. All figure numbers on these charts refer to that publication.

The second item in the operand (FILETYPE) is a one-digit number from 1 to 4 to specify the type of unit record file and the operating conditions.

A 1 indicates that the unit record file is an input file, and that the input unit used by the file will never be shared with a SPOOL program.

A 2 indicates that the unit record file is an output file, and that the output unit used by the file will never be shared with a SPOOL program.

A 3 indicates that the unit record file is an input file, and that the input unit may be shared with a SPOOL program.

A 4 indicates that the unit record file is an output file, and that the output unit may be shared with a SPOOL program.

Whenever a 3 or a 4 is used, the programmer must observe the SPOOL conventions; specifically, the main program must not use index words 95 and 96 or electronic switches 29 and 30.

The third item in the operand (CARDSYNC) is a one-digit number to specify the synchronizer to be used for the unit record file. To read an input file through an IBM 7501 Console Card Reader, a 4 must be used for this item.

The fourth item in the operand (LISTADDR) is the actual or symbolic address of the RDW(s) for the unit record area. The RDWs may be generated by the DA entry that describes the unit record area, or by DRDW operations that use the labels of the DA entries in the operands of the DRDW entries.

The fifth item in the operand (INDXWORD) is either a two-digit number from 03 to 94 or a symbolic name that specifies the index word to be associated with the unit record file. The indexing portion (positions 2 to 5) of the index word will contain the location of the first word of the current unit record. The contents of the index word may be used by the programmer as necessary, provided that the contents are not changed. For example, if the program is to be assembled using Autocoder, the index word may be associated with the instructions that refer to fields within a unit record by adding the index word to the operand of the DA entry that defines the record. This method of assigning index words is described in the publication IBM 7070 Series Programming Systems: Autocoder, Form C28-6121.

The sixth item in the operand (EOFADDRS) is an optional address that may be either actual or symbolic. The address specifies the location of a card reader end-of-file routine or a printer carriage, tape channel 9 routine.

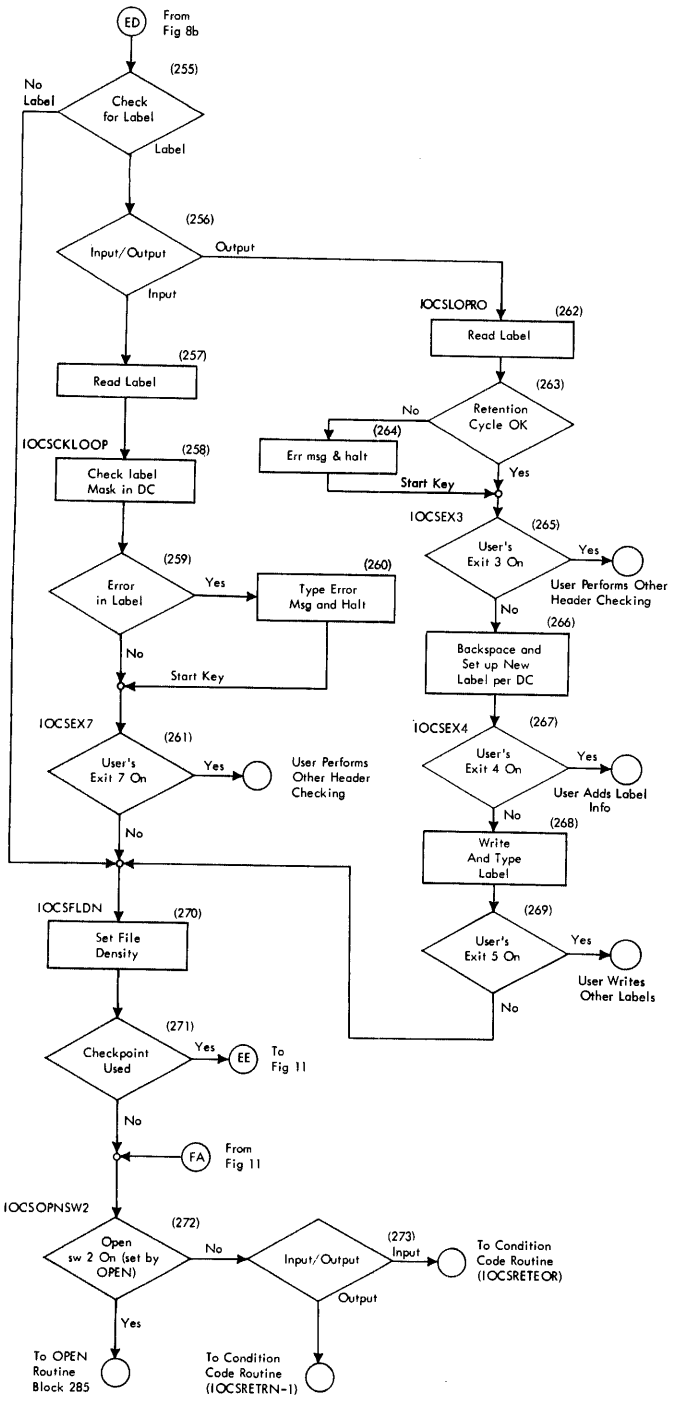


Figure 6

The flow chart shown in figure 6 is taken from Programming Systems Analysis Guide, 7070/7074 IOCS, Form C28-6119. All figure numbers on this chart refer to that publication.

When a card reader end-of-file condition occurs, the program will enter the routine specified by this address. If this item is omitted from the operand, the procedure followed by the IOCS depends on whether SPOOL programs may operate in conjunction with the main program. Either of two procedures will be followed; these procedures are the same as those described for the END macro-instruction (see procedures 2 and 3 under "END Macro-Instruction").

The program will enter the routine specified by this address if a carriage, tape channel 9, condition occurs during the printing of a unit record file. If this item is omitted from the operand, the channel 9 condition will have no effect and the program will continue normally.

Whenever the sixth item is omitted and the seventh item is included, the omission must be indicated by a comma; i.e., two commas will appear between the fifth and the seventh items.

The seventh item in the operand (ERRADDRS) is an optional address that may be either actual or symbolic. The address specifies the location of an error routine that will be entered when an error occurs during the execution of a macro-instruction that refers to the file named in the first item of the DUF entry. If this item is omitted and the file is an input file, an error in reading a card will cause the error record to be typed on the console typewriter. After typing the error record, the machine will halt to allow the operator to correct the error card immediately. If the error card can be corrected later, the operator may press the Start key to read the next card and resume processing. Errors that must be corrected before continuing the program require that the corrected card be fed into the card reader before pressing the Start key. If this item is omitted and the file is an output file, an error in printing or punching a record will cause the error record to be typed on the console typewriter. The error record will also be printed or punched. After typing and printing or punching the error record, processing will resume automatically.

INPUT/OUTPUT MACRO-INSTRUCTIONS FOR UNIT RECORD FILES

The input/output macro-instructions are written as Autocoder instructions with the operand containing the symbolic location(s) of the information to be processed. The symbolic location may be either the name of a unit record file defined by File Specifications or the name of an area in storage defined by a DA entry. The operation portion of the macro-instruction will be the mnemonic representation of one of the functions to be performed by IOCS. Each of these functions will be described separately below.

Each macro-instruction will cause a sequence of instructions to be selected; the number of instructions required will depend on the function and on the information given in the File Specifications. These instructions will be selected and inserted into the object program during Autocoder assembly.

GET Macro-Instruction

To read in a record to be processed, the GET macro-instruction can be used. The GET macro-instruction should be written as follows:

Sequence (Pg)(Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	GET	CARDINFILE
02	ANYLABEL	GET	CARDFILE TO WORKAREA
03			

The operand contains the name of a unit record file that contains the card to be read; this name must be the same as the name used as the first item in the operand of the DUF entry for that file.

The contents of the card will be read into the area defined by an RDW(s) that is located at the address specified in the fourth item of the operand of the DUF entry for the input file named by the GET.

PUT Macro-Instruction

To punch one card or print one line, the macro-instruction PUT can be used. The PUT macro-instruction should be written as follows:

Sequence (Pg)(Lin)	Name (Label)	Operation Code	OPERAND
01	ANYLABEL	PUT	CARDINFILE IN CARDOUTFILE
02	ANYLABEL	PUT	TAPEFILE IN CARDOUTFILE
03	ANYLABEL	PUT	WORKAREA IN CARDOUTFILE
04	ANYLABEL	PUT	CARDOUTFILE
05			

The operand contains the name of the record that is to be printed or punched, and the name of the unit record output file. The name of the record to be included in an output file may be either the name of an area or the name of another file. The examples shown above illustrate how the various types of records are placed into output files. The name of an output file is to be used following the word IN, which must be preceded and followed by a single blank character. The unit record output file name used in the PUT macro-instruction must be the same as the name used as the first item in the operand of the DUF entry for that file. Names of tape files must be the same as the operand of the first entry of the File Specifications for the corresponding tape file. Storage areas used in a PUT macro-instruction must be named in a DA entry.

Before being printed or punched, the records to be included in the unit record output file will be moved to an output area defined by an RDW(s) that is located at the address specified in the fourth item of the operand of the DUF macro-instruction for the output file named by the PUT. Records to be printed or punched may consist of a maximum of 16 words.

PROVISIONS FOR TAPE ERRORS

If a tape operation results in an error, the error correction routines incorporated in IOCS will repeat the operation several times. Different correction procedures are used for input and output files when repeating the tape operation; these procedures are described separately below. Repetition of the tape operation will correct most errors; normal processing of the program will be resumed after the error has been eliminated. Errors that are not corrected automatically may be handled in various ways depending on whether the error occurs while processing an input or an output file.

Correction of Output Tape Errors

The error correction routine used when an error in writing an output file occurs will repeat the writing operation up to 25 times. If the first rewriting attempt does not eliminate the error, each of the last 24 rewriting attempts will be preceded by a Tape Skip operation.

When errors are not eliminated by the repeated tape operation, the error correction routines scan the storage area concerned with the tape operation. These routines check for correct double-digit codes in alphameric words and for valid characters (i.e., each digit must be represented by a 2-out-of-5-bit code). These error correction procedures cover three conditions that are normally responsible for an error in writing an output file, namely:

1. Faulty tape. If a section of tape has been damaged by improper handling or has become dirty, the Tape Skip operations in the rewriting procedure will usually pass over the unusable section and allow the program to continue.

2. Invalid digits. These errors are usually the result of a failure to correct read errors that occurred previously (i.e., invalid digits have been allowed to enter storage).

3. Incorrect double-digit characters. Alphameric words containing double-digit combinations that do not represent valid 7070 characters cannot be entered into 7070 storage by a read operation. Therefore, any such characters in an alphameric word are usually caused by a programming error that creates the invalid double-digit combinations in storage.

Records containing invalid digits or incorrect double-digit characters can be corrected manually from the console through the procedure explained under "Programmed Halts and Messages."

Correction of Input Tape Errors

When an error in reading an input file occurs, an error correction routine will repeat the reading operation and return to the main program after the record has been read correctly. The error procedure consists of nine reading attempts plus a tape cleaner routine. This procedure is executed up to ten times. Errors that are not corrected in ten attempts will be handled according to the procedure specified by Line 14 of the File Specifications. The operand of Line 14 determines the error correction procedure as follows:

If the operand is 00 or blank, a message will be typed and the machine will halt. The operator then locates the word(s) containing the error. To correct an error, the operator may manually store the correct data in the word. Operation of the program is resumed after the error(s) has been corrected.

If the operand is a number from 10 to 49, the program will scan the record area to locate words containing invalid characters. Any word that contains one or more invalid characters will have its sign set to alpha and the contents of the word will be changed to five asterisks. After all words containing an invalid digit(s) have been changed to five asterisks, the block of records will be written on a "dump" tape designated by the number in the operand. The first digit of the number specifies the tape channel to which the tape unit containing the dump tape is connected; the second digit specifies the number of the tape unit that contains the dump tape.

If the operand is 50, the block of records that contains the error(s) will be typed on the console typewriter and the machine will halt. The operator may then proceed to correct the error(s) manually as explained above for a blank operand.

If the operand is 60, the program will continue rereading the block of records until either the error is eliminated or the operator intervenes. The operator may manually enter any of the above procedures to allow correction of the error or to write the block of records on the dump tape.

PROGRAMMED HALTS AND MESSAGES

All IOCS programmed halts are Halt and Proceed (HP) instructions. The halts are listed according to the halt number (digit positions 6 through 9 of the program register typeout). Thus the typeout

1542 -0000002911

would refer to Halt 2911. If an associated message is typed, the text appears in capital letters to the right of the halt number. In the list below, each halt number is accompanied by an explanation and the action to be taken by the operator.

Each halt number is unique. The first digit is either a 2 to indicate that the running of the program may continue after the halt, or a 0 to indicate that no corrective action may be taken at the console. The second digit is a 9 and has no special significance.

The third digit is a number that denotes the subroutine in which the halt occurred: 1 denotes the end-of-reel routine, 2 denotes the checkpoint routine, 3 denotes the restart routine, and 6 denotes any other macro-instruction or subroutine in IOCS. The fourth digit denotes the number of the halt in the subroutine.

Halt Number	Message
2910	TPZZ COUNT DTF-XXXXX TRL-YYYYY Explanation: The block count, XXXXX, in the DTF does not agree with the count, YYYYY, on the trailer of input tape ZZ. Action: The operator may press START to ignore the discrepancy.
2911	TPZZ CK RET DT-XXXXX HDR-YYYYY Explanation: The current date, XXXXX, in location 109 indicates that the retention cycle, YYYYY, for output tape ZZ has not elapsed.
	TPZZ DT ERR LI-XXXXX HDR-YYYYY Explanation: The date, XXXXX, in the label information DC for input tape ZZ does not agree with the creation date, YYYYY, in the label of tape ZZ.
	TPZZ RN ERR LI-XXXXX HDR-YYYYY Explanation: The reel sequence number, XXXXX, in the label information DC for input tape ZZ does not agree with the reel sequence number, YYYYY, in the label of tape ZZ.
	TPZZ FS ERR LI-XXXXX HDR-YYYYY Explanation: The file serial number, XXXXX, in the label information DC for input tape ZZ does not agree with file serial number, YYYYY, in the label of tape ZZ.
	TPZZ ID ERR LI-XXXXX HDR-YYYYY Explanation: The file identification, XXXXX, in the label information DC for input tape ZZ does not agree with the file identification, YYYYY, in the label of tape ZZ. Action: When any of these messages are received, the operator may press START to ignore the indicated check or remount the correct reel on the indicated tape and press PROGRAM RESET and START.
2912	TPXX LAB ERR CY Explanation: An unusual condition occurred when reading a label on tape XX. Y denotes the condition code at the time. Probably there is no label on the tape. Action: The operator may replace the tape and then press START, or press START to repeat the reading.
	TPXX TM ERR CY Explanation: An unusual condition occurred when writing a tape mark on tape XX. Y denotes the condition code. Action: The operator may press START to retry.
2920	Explanation: The tape on which the checkpoint record is being written has reached the end of reel. Action: The operator should change the checkpoint reel and press START.
2921	Explanation: An error in writing the checkpoint record has failed to clear after 25 retries. Action: The operator may press START to retry 25 more times.
2930	Explanation: An error in reading the checkpoint tape has failed to clear after nine retries. Action: The operator may press START to use the next checkpoint record for restarting.
2931	Explanation: The correct checkpoint record was not found. Action: The operator should begin the restart procedure again.

Halt Number	Message
2932	Explanation: The tape from which the checkpoint record is being read has reached end of reel. Action: The operator should change the checkpoint reel and press START.
2933	LABEL, ERROR, START TO BYPASS LABEL Explanation: The existing header tape label on the tape to be used as the checkpoint tape cannot be read correctly. Action: Press START to bypass the existing label.
2934	Explanation: A header label has not been written correctly on the checkpoint tape in five attempts. Action: Press START to try five more times.
2935	RESTORE CONSOLE ETC. FOR RESTART Explanation: At this point the restart routine is ready to restore memory and restart the program. Action: The operator should set the alteration switches and overflow switches as they are to be set for the running of the program.
2936	Explanation: The restart program record has not been written correctly in five attempts. Action: Press START to try five more times.
2937	TPXX LABEL Explanation: The contents of the label will be typed on this line. Action: In restarting, the label on tape XX is typed. If the tape is unlabeled, or if label typing has not been selected, the message will be merely TPXX. If the tape reel known or ascertained to be mounted on this tape unit is the correct one, press START. If an incorrect tape reel is mounted, mount the correct reel on the tape unit. Then, if SPOOL programs are to be executed, store the word +0000000000 into accumulator 1 (9991) and press PROGRAM RESET and START. If SPOOL programs are not being executed, press COMPUTER RESET and START. If multiple tape files are written on the tape reel and the tape is not positioned at the desired tape file, press PROGRAM RESET and START; the tape will be spaced forward to the next tape file, at which point the message and halt will be repeated.
2938	Explanation: A tape mark has been read on a data tape before the proper number of records were read to position the tape for restarting. Action: The operator should mount the correct data file tapes, load the Restart Initiator, and press START.
2939	Explanation: Shared files with alternating tape units are being used. The same unit number applies to both the first reel of the shared checkpoint file and the reel of the shared checkpoint file that contains the checkpoint record to be used for restarting. The first reel of the shared file has been mounted. The processor now needs the reel containing the checkpoint record for restarting. Action: Remove the first reel of the shared checkpoint file, and mount the reel containing the checkpoint record for restarting on the same tape unit.
2960	Explanation: There has been a failure to write a segment mark or a tape mark. Action: Press START to retry five times.
2961	TPXX RDSM ERRY Explanation: There is an error in reading a segment mark on tape XX. Y denotes the condition code at the time. Action: The operator may press START to continue as if the operation had been completed correctly.
2962	Explanation: An error was detected after the execution of a unit record read instruction in the DUF1 macro. (The contents of the card in error are typed.) Action: Have the card in error repunched, put the new card in its proper place, and press START.
2963	DATE ERROR CARD Explanation: The RDLIN macro-

Halt Number	Message
	instruction is being executed. The date in the card against which labels are to be checked does not agree with the date in location 0109. Action: Press START to ignore this card and read the next card.
	ERROR CARD Explanation: The RDLIN macro-instruction is being executed. There is an error in a card against which labels are to be checked. Action: The operator may press START to ignore this card and read the next card.
	CH DR ERR Explanation: The message is typed during execution of the RDLIN macro-instruction. The channel and unit in the DTF is in error. Action: The operator may press START to ignore this card and read the next card.
2964	Explanation: An error was detected after the execution of a unit record read instruction in the DUF3 macro-instruction. (The contents of the card in error are typed.) Action: Have the card in error re-punched, put the new card in its proper place, and press START.
2965	EOJ Explanation: This message is typed at the conclusion of the END macro-instruction to indicate the end of job. The message will not be typed when SPOOL is used.

MESSAGES IN IOCS

The following messages may be typed during a run that uses IOCS. After some of the messages are typed, a programmed halt will occur. In these cases, the halt is merely procedural, halting the computer to allow a manual action to be performed, and the halt number has no significance.

Message

CHGE TP XX
Explanation: The old reel on tape XX has been finished and the next reel is to be mounted on the same unit.
Action: The operator should change reels on the indicated unit, and then press START.

CHK DISABLE ON
Explanation: A search of the area is about to be made for invalid words.
Action: The operator should press the CHECK DISABLE switch on the CE console and then press START.

CHK DISABLE OFF
Explanation: The search for invalid words has been completed. All invalid words have been corrected.
Action: The operator should return the CHECK DISABLE switch on the CE console to its normal position and then press START.

bCHPT+cULDU00NNN
Explanation: This is the checkpoint identification. In this message:
c is the channel of the checkpoint tape.
U is the unit of the checkpoint tape.
L is "0" if the tape is unlabeled, or "1" if the tape is labeled.
D is a code describing the density of the checkpoint tape: "0" is low density label, low density file; "1" is low label, high file; "2" is high label, high file; and NNN is the number of the checkpoint record to be written and is assigned consecutively starting with "001."

ERR+NNNNYYYYY
Explanation: NNNN is the location containing a read or write error. YYYYY is the typeout of that location as it has been found by the IOCS error routine. A halt will occur following this message.
Action: The operator may then correct the information and press START to proceed.

INVALPHA+XY0000ZZZZ
Explanation: Positions X-Y of word ZZZZ contain an invalid double-digit alpha code.

TPXX NOT READY
Explanation: Tape XX is not in ready status. This may happen if XX is rewinding as well as if there is no tape mounted on XX.
Action: The operator should ready tape XX if it is not ready.

TPXX READ FAIL
Explanation: An error in reading tape XX failed to clear after ten sets of nine retries.
The error is treated as specified in Line 14 of the DTF for that file.

TPXX READ FAIL TP WD ERR

Explanation: The preceding error was a tape word error.

The error is treated as specified in Line 14 of the DTF for that file.

TPXX SCLR
TPXX SLR
TPXX EOS
TPXX LLR

Explanation: A short character length record, short length record, segment mark, or long length record has been read from tape XX.

Action: The operator may press START to ignore the unusual condition.

TPXY TO XZ

Explanation: The old reel on tape XY has been finished. The program has flipped to tape XZ for the next reel of the file.

TPXX WRITE FAIL

Explanation: The preceding write error remained after 25 retries, 24 of which were preceded by skips.

Action: The operator may press START for 25 more tries and 25 more skips.

TP YY XXXXX EZZ NQQ PWW

Explanation: The statistics are typed for input tape YY. XXXXX is the block count. ZZ is the number of times during the last reading of the tape reel that the IOCS tape error routine has been entered. QQ is the number of noise errors encountered during the last reading of the tape reel. Noise errors are patches of extraneous magnetism found in the inter-record gaps. WW is the number of permanent errors (errors that were not cleared in nine attempts to read).

TP YY XXXXX EZZ SK QQQ

Explanation: The statistics are typed for output tape YY. XXXXX is the block count. ZZ is the number of times the IOCS tape error routine was entered during the last time the reel was written. QQQ is the number of the skip instruction given by IOCS during the last writing of the reel.

(The contents of a card will be typed.)

Explanation: An error was detected after the execution of a unit record read instruction. The contents of the card in error are typed. This message may occur in either a DUF2 or a DUF4 macro.

(The contents of the 16 word label are typed here.)

Explanation: Contents of the label are typed due to an error in information in the label area.

(The contents of a card will be typed.)

Explanation: An error was detected after the execution of a unit record punch instruction. The contents of the card in error are typed. This message may occur in either a DUF2 or a DUF4 macro. The card is also punched invalid.

CHECKPOINT PROCEDURES

The term "checkpoint" denotes a point in the execution of a program at which previous operations are known to be correct and a record of the status of the system has been made. Checkpoint also refers to the procedure of recording the status at that point in the program. To indicate the status of the system at the time a checkpoint is reached, the checkpoint routine records:

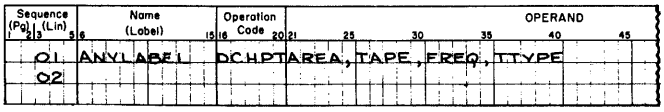
1. The contents of all accumulators.
2. The positions of all tape files.
3. The contents of a storage area(s) specified by the programmer.

Writing of checkpoint records enables an operator to restart a program that has been stopped before the normal end of the job was reached. The operator may restart the program at any checkpoint instead of returning to the beginning of the job.

To incorporate checkpoint and restart routines into programs using IOCS, CHPT must be included in the DIOCS entry. In addition to this indication in the DIOCS entry, a DCHPT descriptive entry (see below) and (if necessary) one or more CHPT macro-instructions must be included.

Checkpoint Descriptive Entry (DCHPT)

When checkpoint records are to be written, one DCHPT descriptive entry must be included in the program. The DCHPT entry should be written as follows:



The first item in the operand (AREA) is used to specify the storage area(s) that are to be written in the checkpoint record. The area may be specified by:

An address, either actual or symbolic, that specifies a list of RDWs defining the storage areas to be written. All RDWs must be plus except the last which must be minus.

The address 4999 to indicate that the storage area from 0000 through 4999 is to be written.

The address 9989 to indicate that the storage area from 0000 through 9989 is to be written.

Omission of the entry. This causes the storage area from 0000 through 4999 to be written. When this item is omitted, the comma must be entered to indicate the omission; i. e., column 21 will contain a comma.

In programs compiled for use on an additional storage machine, the area may be specified by:

An address, either actual or symbolic, that specifies a list of RDWs that define the storage areas to be written. All RDWs must be plus except the last which must be minus.

The address 14999 to indicate that the storage area from 0000 through 14999 is to be written.

The address 19999 to indicate that the storage area from 0000 through 19999 is to be written.

The address 24999 to indicate that the storage area from 0000 through 24999 is to be written.

The address 29999 to indicate that the storage area from 0000 through 29999 is to be written.

This entry may not be omitted for programs to be compiled for use on an additional storage machine. All other items are the same for both normal and additional storage use.

The second item in the operand (TAPE) is used to specify the tape channel and tape unit to be used for writing the checkpoint records. This item must be a two-digit number in the form cu, where c is the tape channel and U is the tape unit number. When checkpoint records are to be written on a tape of an output file (see below), the tape channel and tape unit used in this entry must be the same as the operands of Lines 02 and 03 (FCHANNEL and BASETAPE) of the File Specifications for the output file.

The third item in the operand (FREQ) is used to specify the frequency of writing checkpoint records. A one-digit number from 0 to 3 must be used to select the frequency. The one-digit number specifies the writing of a checkpoint record as follows:

The digit 0 indicates that a checkpoint record is to be written before the processing of each input or output reel is begun.

The digit 1 indicates that a checkpoint record is to be written before the processing of each input reel is begun.

The digit 2 indicates that a checkpoint record is to be written before the processing of each output reel is begun.

The digit 3 indicates that a checkpoint record is to be written only when CHPT macro-instructions (see below) occur in the program.

To begin processing the first reel of any file, the file must appear in the operand of an OPEN macro-instruction. If digits 0, 1, or 2 are used to specify the frequency of checkpoints, it is undesirable to write a checkpoint before beginning processing of each file named in the OPEN macro-instruction because processing of those files begins at the same point in the program. Therefore, whenever one or more tape files named in an OPEN macro-instruction satisfy the frequency conditions 0, 1, or 2, only one checkpoint record will be written.

The fourth item in the operand (TTYPE) is used to specify the type of tape on which the checkpoint records are to be written. A one-digit number from 0 to 2 must be used to specify the type of tape as follows:

The digit 0 indicates that the unit specified in the second item (TAPE) is to be used only for the writing of checkpoints and that the checkpoint tape is not to be labeled.

The digit 1 indicates that the tape unit specified in the second item (TAPE) is to be used only for the writing of checkpoints and that the checkpoint tape is to be labeled.

The digit 2 indicates that the tape unit specified in the second item (TAPE) is used by one of the output files of the program; both output data and checkpoint records will be written on one tape.

When checkpoints are written on the tape of an output file (TTYPE equals 2), the programmer must be sure that no checkpoints occur before the execution of the OPEN macro-instruction that names the output file in its operand. This precaution must be observed to ensure that the header label of the output file has been processed before the first checkpoint record is written.

CHPT Macro-Instruction

Writing of checkpoint records may be specified by use of the macro-instruction CHPT. The CHPT macro-instruction should be written as follows:

Sequence (Pg)(Lin) 1 3	Name (Label) 6	15	Operation Code 16 20	OPERAND 21 25 30 35 40 45
01	ANYLABEL		CHPT	
02				

The operand must be blank.

When the third item in the DCHPT entry (FREQ) is 3 (see "Checkpoint Descriptive Entry (DCHPT)"), checkpoint records are only written when a CHPT macro-instruction is executed. When FREQ is 0, 1, or 2, this macro-instruction may be used to specify additional checkpoints; a checkpoint record will be written each time a CHPT macro-instruction is executed in addition to the points indicated by the 0, 1, or 2.

GENERAL DESCRIPTION OF CHECKPOINT AND RESTART ROUTINES

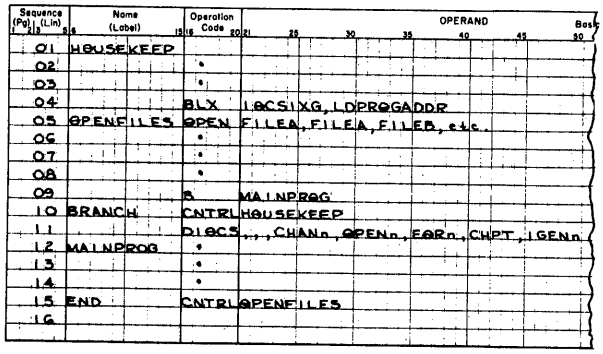
The four IOCS routines that write checkpoints and perform restarts may be described as follows:

1. An initialization and assignment routine initializes the checkpoint routine and the restart routine according to the entries in the DCHPT statement. It processes the header label (if any) of the checkpoint tape, and writes the restart program as the first record on the checkpoint tape. It is executed during an interruption in the loading of the user's program.
2. The checkpoint routine writes the contents of storage on the checkpoint tape at intervals regulated by the DCHPT statement.
3. The restart routine is loaded into storage from the checkpoint tape. It then repositions tape files through use of the block count in the DTF entries and initializes storage by reading the checkpoint record specified by the user.
4. The Restart Initiator is a routine that reads the restart routine from the checkpoint tape and then transfers control to the restart routine. In contrast to the first three routines, the Restart Initiator is not one of the subroutines generated from the DIOCS statement. Before a restart, the Restart Initiator must be loaded separately from the deck of cards (see "Restart Initiator").

INITIALIZATION AND ASSIGNMENT ROUTINE

The initialization and assignment routine is executed during an interruption in the loading of the program. Because loading is always interrupted when using checkpoint, the DIOCS statement must always be positioned in the same manner as described under "Use of OPEN1."

To make the load program available to the initialization and assignment routine, a Branch and Load Location in Index Word (BLX) instruction must be executed immediately before the DIOCS statement is loaded. This BLX instruction must load index word IOCSIXG and branch to the first executable instruction of the load program. The following typical coding example illustrates the use of the BLX instruction and would be proper for any OPEN except OPEN1 or OPEN5.



Coding suitable for use with OPEN1 or OPEN5 would be identical to the above example except for the following changes:

1. When using OPEN1 or OPEN5, all files must be opened with a single OPEN statement.
2. When using OPEN1 or OPEN5, the operand of the END CNTRL statement must not refer to the label of the OPEN statement.

USE OF SHARED FILES WITH ALTERNATING TAPE UNITS

Users of checkpoint may specify an output tape with alternating units as a shared checkpoint tape. The labels of the reels on this shared file will be handled by the IOCS label-handling procedure. (The LABELINF for this file must be restored by the user after the file has been opened.)

When using an output file with alternating tape units as a shared checkpoint file, a 2 must be placed in the CHECKPNT field of the DTF for that file. For all other cases, the CHECKPNT field of the DTF should be 1 or blank.

ADDITIONAL DETAILS ABOUT CHECKPOINT

Before a checkpoint record is written, the following message is typed:

bCHPT+cU1DU00NNN

where

c is the channel of the checkpoint tape

U is the unit of the checkpoint tape

1 is 0 if the tape is unlabeled, or 1 if the tape is labeled.

D is a code describing the density of the checkpoint tape: 0 is low density label, low density file; 1 is low density label, high density file; and 2 is high density label and high density file

NNN is the number of the checkpoint record to be written and is assigned consecutively starting with 001

The two words contained in this message are also the first two words of the checkpoint record. The first two words of the restart routine are "bCHPT" and +0000000000.

For shared files with alternating tape units the checkpoint message is as follows:

cULDARFNNN

where

c is the channel of the checkpoint tape

U is the unit of the first reel of this output file (on which the restart record is written)

L is the label code

D is the density code

A is the unit number of the reel on which the current checkpoint record will be written (This digit may be the same as or different from digit 1.)

R is the RWDPROC of the DTF for the shared checkpoint file

F is a 0 if this checkpoint record will be written on the first reel of the file, or a 1 if this checkpoint record will be written on a reel that is not the first reel of the file

NNN is the number of the checkpoint record to be written

NOTE: If a restart is necessary, the second word of the checkpoint message must be saved and inserted into the machine prior to the restart.

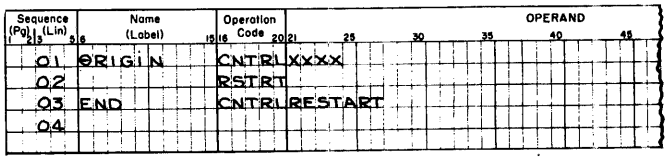
Index Words 1 and 2

Index words used by the load program (index words 1 and 2 in the case of the 7070 Condensed Card Load Program) may not be used as IOCSIXG when the DIOCS statement specifies either OPEN1 or CHPT. Index word 10 may not be used as IOCSIXG when running in the additional storage mode, using the Condensed Card Load Program.

RESTART INITIATOR

The Restart Initiator is a completely separate routine not generated by the DIOCS statement, which loads the restart routine from the checkpoint tape. The initiator should be loaded by the IBM 7070/7074 Condensed Card Load Program prior to the restart. The restart routine is then read from the checkpoint tape by the initiator and is placed in the 375 storage locations immediately following the initiator.

The initiator may be obtained by assembling a program consisting of the following three cards:



The operand portion of the ORIGIN CNTRL entry must be selected so that the Restart Initiator (which occupies about 82 locations) will not be placed in a storage area that is used for subroutines generated by the DIOCS entry in the program to be restarted. When used on an additional storage machine, the Restart Initiator should be assembled in lower core storage.

It is suggested that a convention be established whereby a given block of 82 locations will never be occupied by IOCS subroutines in programs written at an installation. Observing this convention will insure that one Restart Initiator condensed card deck will be suitable for restarting the program.

RESTART PROCEDURE

1. Load the Restart Initiator program. If SPOOL programs are not being executed, this may be accomplished by first loading the 7070/7074 Condensed Card Load Program and then the Restart Initiator. If SPOOL programs are being executed, preface the Restart Initiator with a CONLCARDMAIN card and follow the SPOOL loading procedure described in IBM 7070 SPOOL System, Form J28-6047.

2. Programmed Halt 0050 (Program Register typeout-0000000050) will occur. At this point, store into storage location 0050 the second word of the message that was typed before the checkpoint record (which is to be used for restarting) was written. Note that the form of this word differs when running in the additional storage mode.

If it is desired to cause the restart routine to type the labels of data tapes, set alteration switch 1 OFF; if not, set alteration switch 1 ON. Then press START.

3. Before the repositioning of tapes is started, a message and halt will occur for each tape to be positioned. If label typing has been selected, the

message will be TPxx LABL, followed by the label. If the tape is unlabeled, or if label typing has not been selected, the message will be merely TPxx. If the tape reel known or ascertained to be mounted on this tape unit is the correct one, press START.

If an incorrect tape reel is detected, mount the correct reel on the tape unit. Then, if SPOOL programs are being executed, store the word +0000000000 into accumulator 1(9991) and press PROGRAM RESET and START. If SPOOL programs are not being executed, press COMPUTER RESET and START.

If multiple tape files are written on the tape reel and the tape is not positioned at the desired tape file, press PROGRAM RESET and START; the tape will be spaced forward to the next tape file, at which point the message and halt will be repeated.

4. When all tape reels have been positioned, the message RESTORE CONSOLE ETC. FOR RESTART will be typed, and the computer will halt. At this time set all console switches and dials, ready unit record devices, and so forth, as appropriate to the program that is being restarted, and then press START. Execution of the program will resume.

NOTE: At this point the Load Program is set to read from the input unit used for the Restart Initiator. If the Load Program is to be used by the restarted program, it must be changed to refer to the correct input device.

Restart Procedure when Using Shared Files with Alternating Tape Units

Before loading the Restart Initiator program, the user should make sure that the first reel of the shared checkpoint file is mounted on the proper tape unit. The reel of the shared checkpoint file containing the checkpoint record desired for restarting should also be mounted on the proper unit. If these tapes have the same unit number, then the first reel of the shared files should be mounted. (Halt 2939 will occur later and the user will then mount the reel containing the desired checkpoint record on this unit.)

After the Restart Initiator has been loaded, the halt -0000000050 will occur. The user should store into location 50 a word of the form:

cULDARFNNN

(See the checkpoint message under "Additional Details About Checkpoint").

where

U is the unit of the first reel of the checkpoint tape

A is the unit on which the tape containing the desired checkpoint record is located (A may be the same as U)

F is a 0 if the desired checkpoint record is located on the first reel of file, or a 1 if the desired checkpoint record is located on a reel which is not the first reel of file.

Please note that if the desired checkpoint record is on the first reel of file, the user must place the number of this unit in position 4 of the above word.

If the desired checkpoint record is located on a reel that is not the first reel of the shared file but has the same unit number as the first reel of the file, halt 2939 will occur. The user will then change the checkpoint reel and mount the tape containing the desired checkpoint record on this reel.

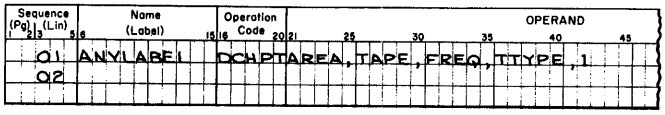
With these exceptions, the restart procedure will be the same as that used for shared checkpoint files that do not have alternating tape units.

Processing More than One Header Label During Restart

If a file that is involved in restarting has additional header labels other than the standard single header label, the first word of each label must be xHDRx, where x represents any character.

Restarting with SPOOL

If SPOOL is running during a restart, the DCHPT entry of the program to be restarted must have a fifth operand of 1.



The user must also provide an RDW list for the writing of checkpoint records. Locations 0095-0096, 0104-0105, the Initial and Final Status Words of the tapes being used for SPOOL, and the locations occupied by the SPOOL programs themselves must not be described in the RDW list as part of the checkpoint record. As many as fifteen RDWs may be specified.

PROVISIONS FOR SPOOL PROGRAMS

IOCS has been designed to accommodate one or two SPOOL programs. A SPOOL program may be initiated during the running of the main program without any interruption to the main program. Similarly,

a new main program can be initiated during the running of one or two SPOOL programs without any interruption to the SPOOL function.

The SPOOL programs are described in IBM 7070 SPOOL System, Form J28-6047-1. So that these SPOOL programs can be used with any main program, conventions regarding the allocation of index words and storage locations must be observed. These conventions are explained in detail in the publication describing the SPOOL System.

If SPOOL programs are to be run on an additional storage machine, the machine must be in the normal mode.

IOCS observes the conventions required by the SPOOL System. The programmer may proceed to write programs using the macro-instructions described in this manual with no special regard for SPOOL programs if index words are specified in symbolic form. If they are specified in actual form, the programmer must avoid using the index words required by the SPOOL programs.

The DIOCS entry of main programs that are to be run with SPOOL programs must include provisions for tape channels needed by the SPOOL program(s). For example, if a SPOOL program uses tape channel 2, the DIOCS entry must specify at least 2 channels; e.g., CHAN2 would appear in the operand.

PART III: WRITING PROGRAMS USING THE IOCS

The last part of this publication is devoted to programs using IOCS. Recommendations for writing and assembling programs are presented, and limitations of certain options are given, so that unusual situations may be avoided. Additional methods of processing are mentioned to aid programmers who must handle records that cannot be processed in the normal manner.

USE OF IOCS WITH AUTOCODER

Although IOCS is presented in a separate publication, it is actually a part of IBM 7070 Autocoder and IBM 7070 Four-Tape Autocoder. It can be included in a program by compiling the program with either of these Autocoder systems. Programs written using IOCS must satisfy minimum requirements regarding certain input and output entries, as explained below. Other information concerning the positioning of input and output entries and the use of the various input and output features is described below for programs to be processed using IBM 7070 Autocoder. This data applies to programs to be processed using IBM 7070 Four-Tape Autocoder, except for the minor restrictions specified under "Use of IOCS with Four-Tape Autocoder."

Quantity and Type of Entries

Each program using IOCS must include the following entries:

1. One DIOCS entry for programs using tape files.
2. One DCHPT entry if checkpoint records are to be written.
3. For each tape file:
   a. One DTF entry.
   b. One DA entry to define storage areas for the record. The areas must have RDWs which may be created through the DA entry or by a separate DRDW entry; areas for Form 4 records must always use a DRDW entry to create RDWs.
   c. The use of an OPEN macro-instruction that refers to the file. If the DIOCS entry specifies OPEN1, only one OPEN macro-instruction may be used in the program.
   d. At least one processing macro-instruction (e.g., GET, PUT) that refers to the file.
   e. If the file is an input file, a routine to be entered when an end-of-file condition occurs.
   f. A CLOSE or an END macro-instruction that refers to the file.
   g. One DC entry specifying label information if tape labels are to be processed.
4. For each unit record file:
   a. One DUF entry.
   b. One DA entry to define storage areas for the record. The areas must have RDWs that may be created through the DA entry or by a separate DRDW entry.
   c. At least one GET or PUT macro-instruction that refers to the file.

Positioning of Entries

The positioning of various elements of IOCS is described separately below. In general, the entries may be positioned in any 7070 storage locations, provided that Autocoder rules are observed. For example, declarative operations should be separated from the program instruction area, or the program must branch around declarative operations that are in the instruction area. The order of the source program cards for assembly by the Autocoder system is not critical except when certain processing procedures are used by the program being assembled. These exceptions are explained in the following descriptions of the positioning of the elements of IOCS:

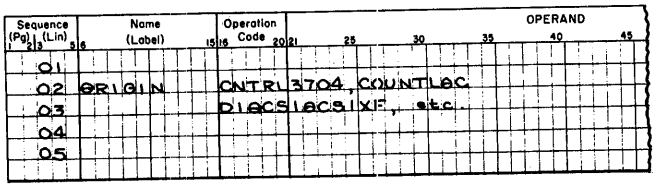
Macro-Instructions: All IOCS macro-instructions generate a series of 7070 instructions that are positioned at the point in the source program where the macro-instruction was written. In addition, coding generated out of line by IOCS macro-instructions will be positioned after the first LITORIGIN CNTRL statement in the source program.

File Schedulers: Each DTF entry generate a File Scheduler consisting of a group of 7070 instructions for reading or writing a tape file. As mentioned in the explanation of the File Specifications, all DTF entries (36 cards each) must be loaded consecutively. The File Schedulers will be generated in the same order and positioned under control of the next LITORIGIN CNTRL entry.

(When IBM 7070 Four-Tape Autocoder is used, the File Schedulers will be positioned immediately following the last DTF entry.)

Descriptive Entries: The various descriptive entries required by IOCS should be positioned in the same manner as declarative entries. That is, descriptive entries should be separated from the program instruction area, or the program must branch around them so that the program will not attempt to execute descriptive data as instructions.

Subroutines: All IOCS subroutines are positioned immediately following the DIOCS entry. If the programmer wishes to have the subroutines positioned in a specific storage area, an ORIGIN CNTRL entry may be used preceding the DIOCS entry as in the following example:

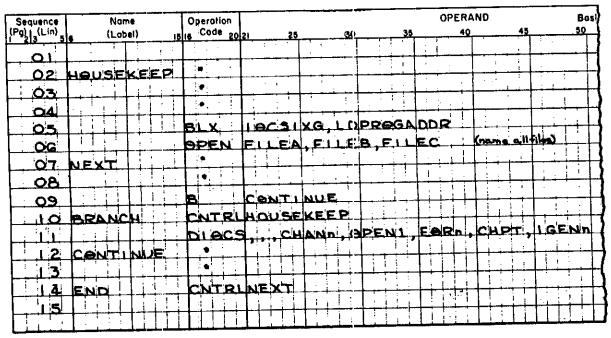


The subroutines are positioned beginning at 7070 storage location 3704.

Use of OPEN1 and OPEN5

Programs written using a DIOCS entry with OPEN1 or OPEN5 in its operand require that the loading of the program be interrupted to execute the OPEN macro-instruction. The DIOCS entry must be positioned in the program so that all the IOCS entries listed under "Quantity and Type of Entries" (except, of course, the DIOCS entry) will have been loaded when the DIOCS entry is reached.

When OPEN1 or OPEN5 is specified in the DIOCS entry, the program must be written so that IOCS subroutines will be loaded immediately after the OPEN macro-instruction is executed. This requires that a Branch and Load Location in the Index Word (BLX) instruction be executed immediately before the OPEN macro-instruction. The BLX instruction must load index word IOCSIXG and branch to the load program; the index word and the location of the load program may be specified in either actual or symbolic form. Normally, the OPEN macro-instruction will be written in a section of the program devoted to "housekeeping" and is executed during a brief interruption of the loading procedure. One of several methods of insuring that the subroutines will be loaded immediately after the OPEN macro-instruction is executed is shown in the following example:



The symbol LDPROGADDR on line 4 of the coding sheet is the address of the first executable instruction in the load program.

The BRANCH CNTRL entry will interrupt the loading procedure and branch to the instruction in symbolic location HOUSEKEEP. Instructions in the housekeeping section of the program will be executed, and the BLX instruction will branch to the load program to load the OPEN subroutine generated from data in the DIOCS entry during assembly of the program. (If the IBM 7070 Condensed Card Load Program is being used, and if the branch instruction inserted into location 0000 has not been changed or destroyed, the address LDPROGADDR in the BLX instruction may be 0000.) A BRANCH CNTRL at the end of the OPEN subroutine will interrupt the loading and the OPEN subroutine will be executed to open all tape files named in the operand of the OPEN macro-instruction. Loading of other subroutines generated from the DIOCS entry will occur next. After all subroutines have been loaded, loading of the program will resume at the point following the DIOCS entry. Initialization of accumulator 1, and index words 1 and 2, for use by the main program must not be executed during the housekeeping section of the program. The housekeeping section is executed before the OPEN subroutine is loaded because the IBM 7070 Condensed Card Load Program uses these locations and would destroy the initial settings. Loading will continue until the END CNTRL entry begins executing the program by branching to symbolic location NEXT.

When using OPEN1, OPEN5, or checkpoint, where the loading procedure is interrupted, certain IOCS entries must be loaded before the IOCS subroutines are loaded. IOCS macro-instructions may be loaded after the DIOCS entry. However, the following entries must be loaded before the DIOCS entry:

1. The DCHPT entry, if checkpoint records are to be written.
2. The DTF entry for each file.
3. The RDWs for the tape record area of each file.
4. The DC label information entry for each labeled file.

Therefore, when using Autocoder, a LITORIGIN CNTRL statement should be given before the BRANCH CNTRL statement.

If the programmer wishes to position the subroutines in specific storage locations, an ORIGIN CNTRL entry may be inserted immediately preceding the DIOCS entry.

Use of OPEN3

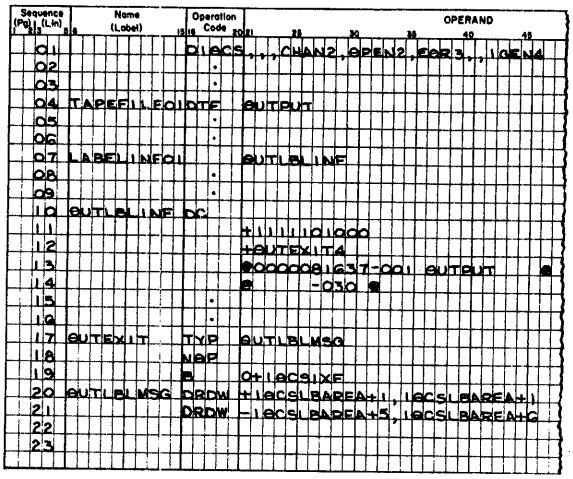
Programs written using a DIOCS entry with OPEN3 in its operand require that the branch instruction

placed in location 0000 by the IBM 7070 Condensed Card Load Program not be changed or destroyed during the running of the program. The DIOCS entry must be positioned in the program so that all the IOCS entries listed under "Quantity and Type of Entries" (except the DIOCS entry) will have been loaded when the DIOCS entry is reached. The program must contain a DCHPT entry that specifies a separate tape unit to be used for checkpoint records. The OPEN subroutine will be written on the checkpoint tape, and read into storage whenever an OPEN macro-instruction is executed.

Use of EOR3

When a program contains a DIOCS entry with EOR3 in its operand, the header labels written on output reels will not be typed automatically. This procedure reduces typing time in the end-of-reel routine. If the programmer wants some portion of the header labels of a certain output file typed, he should (1) specify EOR3 in the DIOCS, and (2) specify, in the label information DC for the file, that end-of-reel exit 4 is to be taken. In his exit 4 routine, the programmer should specify the parts of the label to be typed by giving a type instruction and RDW(s).

The coding in the following example will cause only 1HDR and the file identification to be typed:



Use of BRANCH CNTRL Entries

Whenever BRANCH CNTRL entries occur in a program, the loading of the program is interrupted to execute some instructions and then the loading procedure resumes. The loading of the various elements of IOCS must be considered when writing BRANCH CNTRL entries into a program. If a BRANCH CNTRL entry causes the execution of a section of a program that includes an IOCS macro-instruction, the programmer must be sure that the system elements needed for the macro-instruction have been loaded. For example, an OPEN macro-instruction could not be executed if the DTF entries and the related File Schedulers are not in 7070 storage.

A typical use of a BRANCH CNTRL entry in a program using the IOCS is described in detail under "Use of OPEN1." When using 7070 Autocoder, a LITORIGIN CNTRL statement should be given before the BRANCH CNTRL statement.

Priority Mask

Programs that use IOCS must not use the following instructions:

1. Priority Control (+55)
2. Stacking Latch Set (-61)
3. Stacking Latch Reset (-62)

In order for IOCS to function properly, priority conditions must be under the control of the system at all times. This control is accomplished through the use of priority masks. By avoiding the three instructions listed above, the programmer will prevent interference with IOCS.

Program Exits

Throughout the IOCS, program exits have been provided to allow branching to the user's routines for processing that is peculiar to the program being executed. Exits are available for end-of-file routines, label routines, etc. (see the File Specifications). After the user's routine has been executed, it is necessary to return to the instruction that follows the program exit. The user can accomplish this by ending each of the routines with a branch instruction. If the routine was entered from a program exit provided through the File Specifications for tape files, the routine should end with a branch instruction as specifically stated in the description of the program exit. Routines entered from a program exit provided by the descriptive entry for unit record files should end with a Branch to 0+IOCSIXH. In order that control will return to the proper point in the program, no IOCS macro-instructions may appear in a routine entered from a program exit.

Use of File Names in Autocoder Macro-Instructions

Most Autocoder macro-instructions (e.g., ZERO and MOVE) may refer to input and output files. When

the name of an input or output file is used in an Autocoder macro-instruction, the name used must satisfy the following requirements:

1. The name of the file used in the operand of the DTF entry must be identical to the name of the DA entry that defines the areas for the input or output records. Depending on the method of generating RDWs, this name may or may not be the same as the operand of the IORDWLIST entry of the File Specifications.
2. The operand of the DA entry that defines the areas must specify an index word and that index word must be the same as the operand of the INDXWRDA entry of the File Specifications.

Use of Tape Record Index Words

The contents of the index words assigned to each tape file in the File Specifications may be used by the programmer as necessary provided that the contents are not changed. The data that is contained in these index words depends on the form of record and on whether the file is for input or output. The contents of the index words for each form of record are described separately below. The index words for each file are referred to as XA and XB, and are assigned to a file as explained under "File Specifications Entries," Lines 19 and 20.

Form 1 and Form 2: XA for both input and output files using these forms of records will contain the RDW of the current record. Positions 2 - 5 of XB contain the location of the RDW of the current record; positions 6 - 9 of XB will contain the location of the last RDW of the current record block.

Form 3 -- Input Files: Positions 2 - 5 of XA contain the location of the first word in the current record; positions 6 - 9 of XA contain the location of the last word in the current record block. XB contains the length of the current record in positions 2 - 5, and positions 6 - 9 contain zeros.

Form 3 -- Output Files: Positions 2 - 5 of XA contain the location of the last word in an output area that has been filled with output data. Positions 6 - 9 of XA contain the location of the last word of the output area. XB contains the RDW of the last record moved into the output area.

Form 4: XA for both input and output files using this form of record will contain the RDW of the first part of the current record. Positions 2 - 5 of XB will contain the location of the RDW of the first part of the current record; positions 6 - 9 of XB will contain the location of the last RDW of the current record block.

Use of Unit Record Index Word

The contents of the index word assigned to each unit record file in the File Specifications may be used by the programmer as necessary provided that the contents are not changed. Positions 2 - 5 of the index word will contain the location of the first word of the current unit record.

Changing Record Length

Form 2, 3, and 4 records are each variable in length. The method of changing each form is described below.

Form 2: The length of a Form 2 record is changed by placing the record mark at the desired place in the record. The changes in the record length may be made in an input, output, or work area.

Form 3: When the length of Form 3 records is changed, the contents of XA, XB, and the record length indicator in the record must also be changed. The following precautions must be observed when changing the length of a Form 3 record.

When processing in an input area, records may be shortened but should not be lengthened. Lengthening a record will destroy the first part of the next input record.

Form 3 records processed in a work area may be shortened or lengthened as desired. When processing in a work area, it is only necessary for the user to change the contents of the record length indicator; the contents of the index words will be adjusted by IOCS when the record is moved to the output area.

When processing in an output area, records may be shortened without complications. If records are lengthened, the user must be sure that lengthening does not require more words than remain in the output area for that record block.

Form 4: The length of each section of a Form 4 record is changed by placing the record mark at the desired place in each section of the record. The changes in the record length may be made in an input, output, or work area.

Processing in Output Areas

Records may be processed or assembled in an output area if necessary. For example, records to be inserted into a file may be assembled in the output area rather than in a work area.

To process records in an output area, the next space in the output area is made available to the user by issuing a PUT macro-instruction that names only the output file in its operand; see the last example under "PUT Macro-Instruction." Processing a record in the output area requires that the PUT

macro-instruction be given before data can be placed into the output area. When using this procedure, the programmer must not use a PUT macro-instruction after the last output record has been placed in the output area, since an extra PUT used at this time will add another output record containing unwanted and unknown data to the output file. This procedure of issuing the PUT before processing the record is the reverse of the normal processing procedure. (The record is normally processed in an input or work area, and entered into the output file by a PUT or PUTX macro-instruction after processing has been completed.)

When this type of PUT macro-instruction is used with Form 3 records, the PUT entry must have a label, and the number of words in the record must be placed in positions 6 - 9 of the first instruction of the PUT macro-instruction. One method of inserting the record length is shown by the following routines, where FORM3LNGTH is the label of a field that contains the number of words for the record:

Sequence (Pg) (Lin)	Name (Label)	Operation Code	OPERAND
01		ZA1	FORM3LNGTH
02		STD1	FORM3PUT
03		*	
04		*	
05	FORM3PUT	PUT	OUTFILE
06		*	
07			

The PUT macro-instruction will alter the index words associated with the output file to refer to the next space in the output area. The RDW that defines the next space will be in XA for Form 1, 2, and 4 records or XB for Form 3 records. Processing in the output area may be done using instructions indexed by the appropriate index word for the record form being processed.

Occasionally, it may be desired to process a record in the output area using the same routine that is designed for processing in an input area; that is, using XA of the input file. The programmer may do this by writing a routine that:

1. Saves the contents of XA of the input file.
2. Moves the contents of XA of the output file (XB for Form 3 records) to XA of the input file.
3. Executes the routine which processes the record in the output area.
4. Restores the original contents of XA of the input file.

Processing Long Records

If very long records must be processed, and storage space is not available for input and output areas large enough for the maximum length records, a combination of processing in the input and output areas may be used. The input and output areas must each be capable of holding at least one half the maximum length record, and records that can not fit into the input area must be written as two tape records.

A GET macro-instruction places the first half of the record in an input area which is then moved to an output area by a PUT macro-instruction. The next GET macro-instruction will place the second half of the record in an input area. Both halves of the record may then be processed. The first half may be processed using instructions indexed by index word XA of the output file (XB for Form 3 records), and the second half using instructions indexed by index word XA of the input file.

PRECOMPILED IOCS SUBROUTINE DECK

Considerable interest has been expressed by IOCS users in a precompiled, or "prepunched," condensed card deck of the IOCS subroutines. Instead of including a DIOCS statement in a source program to cause generation of IOCS subroutines by compilation, the precompiled IOCS subroutine deck would be added to the program after compilation. The goal of this procedure is a considerable reduction of compilation time.

The procedure requires that certain conventions be followed in order to obtain proper communication between the IOCS subroutines and the IOCS elements contained in the source program (e.g., DTF entries, the DCHPT entry, and IOCS macro-instructions). The required conventions are:

1. The programmer should reserve index words 1 and 2 for programs to be used with the precompiled IOCS subroutines.
2. The programmer should reserve index words 1, 2, and 10 for programs to be run in the additional storage mode; only one configuration of the IOCS subroutines may be used.
3. The DCHPT entry, the first word of the first DTF entry, and the IOCS index words must occupy the same storage locations in all source programs to use the precompiled IOCS subroutine deck.
4. The addresses of certain symbolic labels within each subroutine must be made available to the source program. This convention may be achieved in two different ways. One way is to include EQU statements to fix the locations of all symbolic labels in the IOCS subroutines to which the IOCS macro-instructions refer. This method is limited, because any changes in the IOCS which would produce changes in the locations of the symbolic labels would necessitate reassembling the source program. The other method is to include a branch list. This list would occupy a number of fixed locations, and would be included in the precompiled IOCS subroutine deck.

It would contain a list of branch instructions. The operand of each branch instruction would contain a symbolic label appearing in the IOCS subroutines. All the symbolic labels to which IOCS macro-instructions refer would be in the list. The source program would then include EQU statements which would equate each symbolic label in the IOCS subroutines to a fixed branch list location. In this way, changes in the IOCS subroutines would not necessitate reassembly of source programs.

In programs to be used with precompiled IOCS subroutines, certain labels must be equated to their actual locations.

The user should equate the label IOCSIGEN to its actual location in the IOCS subroutine. IOCSC1S, IOCSC2S, IOCSC3S, and IOCSC4S should be equated to their actual locations in the IOCS subroutines.

IOCSRLSMOD should be equated to a branch list location. This branch list location should contain:

```
B IOCSRLSMOD + 1
```

If the programmer wishes to use the FEORN macro-instruction, the following labels must be equated to their respective locations:

Autocoder	Four-Tape Autocoder
IOC. EOR	IOCSEOR
IOC. TEF	IOCSTEF
IOC. EOFEX	IOCSEOFEX
IOC. OPNSW2	IOCSOPNSW2
IOC. RETEOR	IOCSRETEOR
IOC. MASKA	
IOC. MASKP	
IOC. SEQILB	

If the programmer wishes to use the FEOR macro-instruction, the following labels must be equated to their respective locations:

Autocoder	Four-Tape Autocoder
IOC. ICHECK	IOCSICHECK
IOC. CELOOP	IOCSCELOOP
IOC. OPNSW2	IOCSOPNSW2
IOC. RETRN	IOCSRETRN
IOC. IPSLO	IOCSIPSLO
IOC. CEBACK	IOCSCEBACK
IOC. MASKA	
IOC. MASKP	

Autocoder source programs that include the CHPT macro-instruction must equate the IOCS label IOC. MASKA to its actual location. If the IOCS macro-instructions WTM, WSM, BSP, RWD, RDSF, or RDSB are used in the source programs, the IOCS labels IOC. MASKA and IOC. MASKP must be equated to their actual locations.

Compiling Source Programs

Normal programming and compiling procedures apply to any source program that uses the condensed IOCS subroutine deck, with these exceptions:

1. The program must not contain a DIOCS statement.
2. In order to locate the symbolic labels in the IOCS subroutines to which IOCS macro-instructions refer, a branch list is used. The following labels must be equated to their respective absolute branch list locations:

Autocoder	Four-Tape Autocoder
IOC. ICLOSE	IOCSICLOSE
IOC. IEND	IOCSIEND
IOC. IOPEN	IOCSIOPEN
IOC. IRTAIN	IOCSIRTAIN
IOC. EJLOOP	IOCSEJLOOP
IOC. RLSMOD	IOCSRLSMOD (see above explanation)
IOC. CNBTST	IOCSCNBTST

3. The following index words must be equated to the locations they occupy. These locations are indicated in the program listing of the IOCS subroutines.

Autocoder	Four-Tape Autocoder
IOCSIXF	IOCSIXF
IOCSIXG	IOCSIXG
IOCSIXH	IOCSIXH

4. Care must be taken that the DTF entries and the DCHPT entry satisfy the conventions established for the IOCS subroutines.

Combining the Object Decks

The condensed IOCS subroutine deck and the condensed deck of a program that is to use these IOCS subroutines may be combined in one of two ways. If the IOCS subroutines include OPEN1, OPEN5, or CHPT, the IOCS subroutine deck must be inserted in the condensed program deck immediately following the execute card generated from the BRANCH CNTRL statement that interrupts the loading process to prepare for the execution of certain IOCS subroutines. If the IOCS subroutines include OPEN2, OPEN4, or OPEN6, and do not include

CHPT, the IOCS subroutine deck may be inserted at any point within the condensed program deck such that the subroutines will have been loaded before being executed.

The extent of these conventions makes it unfeasible for the IBM Programming Systems Department to supply a precompiled IOCS subroutine deck on a general basis. Any 7070 or 7074 user may, however, establish these conventions for his own programs, and very easily obtain the IOCS subroutine deck.

Compiling the IOCS Subroutine Deck

The condensed card IOCS subroutine deck may be obtained by compiling a small program. The program should contain the following entries:

1. An ORIGIN CNTRL to establish the beginning location for the IOCS subroutines.
2. An EQU statement to fix the location of the first word of the first DTF entry. This location must be common to all programs to use this IOCS subroutine deck. The label of the EQU entry is IOC. FTBL01, if using Autocoder, or IOCSFTBL01, if using Four-Tape Autocoder.
3. An EQU statement to fix the location of the DCHPT entry. This location must be common to all programs that use this IOCS subroutine deck. The label of the EQU entry is IOC. DCHPT, if using Autocoder, or IOCSDCHPT, if using Four-Tape Autocoder.
4. A branch list.
5. A DIOCS statement that specifies the desired configuration of the IOCS subroutines. The DIOCS statement must also fix the index word locations to be used for index words IOCSIXF and IOCSIXG. These index words must be reserved for IOCS use in all programs that use this IOCS subroutine deck.

After compilation, the final execute card and the load program should be removed from the condensed card deck and destroyed.

7072/7074 IOCS FOR ADDITIONAL STORAGE

Autocoder generates IOCS routines for use in the Additional Storage mode when the ADDSTORAGE OBJCT YES operating option control card is used.

IOCS macro-instructions, subroutines, and File Specification entries for the 7072/7074 system with additional storage are similar to the corresponding elements of the normal package. Therefore, they are used in exactly the same manner. When writing programs in the Additional Storage mode, the user should comply with all the requirements and specifications of IOCS in the normal mode. Whenever a different entry or procedure is required for the Additional Storage mode, it will be described along with the procedure for the normal case.

The following rules must be considered when using 7072/7074 IOCS for additional storage:

Block Size: The maximum number of storage words allowable in a Form 3 tape block is 9998. Existing regulations on maximum record and block length for use with the Sort 90, Merge 91, Tape Duplication, and Tape Compare programs must also be observed.

DIOCS Entry: The DIOCS entry may appear anywhere in card sequence, provided that the requirements for the use of overlaid OPEN subroutines and checkpoint routines are met.

SPOOL: SPOOL will not be provided. Therefore, neither IGEN1 nor IGEN2 should be used in the DIOCS statement.

DTF Entries: DTF entries must be placed in storage locations below 10000. Symbolic operands of the following DTF entries must refer to locations below 10000.

SLRPROCD	EOSPROCD
LLRPROCD	EORPROCD
SCLPROCD	EOFPROCD
LABELINF	

The locations referenced may contain branch instructions to routines located at or above 10000. Note that locations 0000 and 9999 may not be used as IOCS exits when these values are used to denote special options that are described for the entry.

The File Schedulers generated for each DTF entry must be placed in module zero. This is accomplished by proper use of the Litorigin Control statement(s) to force assignment of generated material to previously reserved locations below 10000.

The SPAREINF field in the File Specifications Table must not be used at any time during execution of the program.

Record Definition Words: All RDWs used by IOCS must be below location 10000. The assignment of RDWs generated for use with macro-instructions must be controlled by the use of a LITORIGIN CNTRL statement (see above). RDWs defining input or output areas must not be positioned above location 9998.

Label Information DC Entries: The final entry under any LABELINF DC entry must not be above location 9998. Also, any symbolic addresses for label exits given under a LABELINF DC entry must refer to actual locations below 10000. The referenced addresses may contain branch instructions to routines at or above location 10000.

The Restart Initiator must be assembled below location 10000.

IOCS FOR IBM 7330 MAGNETIC TAPE UNITS OF THE 7072 DATA PROCESSING SYSTEM

The File Specifications and macro-instructions that are applicable to files on 729 II and IV Magnetic Tape Units are also applicable to files on 7330 Magnetic Tape Units.

It is important to note the following when using 7330 Magnetic Tape Units:

1. Set density instructions are treated as NOPs. Therefore, Line 21 of the DTF specifications will be ineffective for files on 7330 Magnetic Tape Units.

2. If a read instruction of any type is followed (without an intervening BSP or RWD instruction) by a write instruction, an imperfectly erased inter-record gap may be produced. Therefore, a RDSF or RDSB macro-instruction on an output file will cause difficulty if any write commands are subsequently given. Under these conditions, the following two instructions should be given after the RDSF or RDSB macro-instruction:

```
BSP     OUTPUT
WSM     OUTPUT
```

USE OF THE READ BINARY FEATURE WITH IOCS

The 7072 and 7074 Data Processing Systems have an optional feature that permits them to read odd-parity binary tapes prepared by IBM 704, 709, or 7090 systems or by analog-digital converters for these systems. The File Specifications for the tape files consisting of odd-parity binary tapes are the same as those for other files with one exception: the user should insert a 0 on Line 09 (FILETYPE) under the DTF for that file. When reading binary tape, modifications to the condition code routine will provide for the overlapping of input/output operations when encountering tape records with short character length characteristics. If modifications are desired, the user should specify in the DIOCS line either IGEN5 or IGEN6 which provides for the above-mentioned overlapping when encountering short character length binary records. IGEN5 contains all the features of IGEN1 in addition to this feature. IGEN6 contains all the features of IGEN3. In this case, Line 24 (SCLPROCD) in the binary File Specifications will not be effective.

USE OF THE IOCS WITH FOUR-TAPE AUTOCODER

Programs using IOCS that are to be processed with IBM 7070 Four-Tape Autocoder must observe several restrictions regarding fields, areas, index words, and DA entries. Except for the following restrictions, the remarks given under "Use of IOCS with Autocoder" also apply to programs to be processed with IBM 7070 Four-Tape Autocoder.

When using PUT macro-instructions with Four-Tape Autocoder, the name preceding the word IN must be the name of either an RDW that defines one area or a tape input file. The use of a field name is not allowed (unless, of course, the field name is the label of an RDW that defines the field). The name of a card input file may not be used in the PUT macro-instruction if the output file is a tape file. A record from a card input file may be included in a tape output file if the unit record area is defined by one RDW; the PUT would then be written using the name of the RDW (i.e., the name preceding the word IN would be the same as the fourth item in the DUF entry of the card input file).

Index words may not be associated with fields defined by subsequent entries under a DA by entering the index word in the operand of the DA entry. To use indexing with instructions referring to fields defined under a DA, the user must enter the index word in the operand of each instruction that is to be indexed.

SUMMARY OF STORAGE, INDEX WORD, AND ELECTRONIC SWITCH UTILIZATION

The programs developed by the IBM Programming Systems Department use certain index words, electronic switches, and storage areas that must be avoided by the user when writing his own programs. The items to be avoided will depend on which of the IBM programs are to be used with the user's program. The utilization restrictions of these programs are described separately below.

IBM 7070-Series Input/Output Control System for Tape and Unit Records

The IOCS uses two index words for each tape file and one for each card file. In addition to the index words associated with the files, the IOCS uses two index words for control of the system regardless of the number of tape files; if one or more card files are included in the program, one additional index word is used by the system.

The number of words of storage will depend on the files and on use of macro-instructions in the program. The actual locations of storage words used will be assigned during Autocoder assembly. From 20 to 41 words of storage will be used for reading and writing routines for each tape file. Each GET or PUT macro-instruction will use from 3 to 8 words. Nine words will be used for the File Specifications of each tape file.

IBM 7070 SPOOL System

The SPOOL System uses index words 95 and 96. It also uses electronic switches 29 and 30.

Each SPOOL program uses 200 words of storage. The SPOOL programs will be located in the upper 400 words of storage. When using the IBM 7070 SPOOL System, the load program must be in storage locations 0300 through 0323.

IBM 7070 Condensed Card Load Program

This program uses index words 01 and 02 during the loading of a program; these index words may be used by the programmer after loading has been completed.

The program uses storage location 0000 for an instruction to branch to the load program. Normally, the program occupies storage locations 0300 through 0323, but it may be relocated to any 24 consecutive storage locations. If the IBM 7070 SPOOL System is to be used, the load program must use locations 0300 through 0323.

APPENDIX: SCHEDULER OPERATION

The most important elements of IOCS are the File Schedulers and the Channel Schedulers. This appendix has been included for those programmers who are interested in the theory of operation of these elements. File Schedulers and Channel Schedulers are described because they have complete control over the reading and writing of records and the scheduling of tape operations among tape files using a common tape channel. Other elements of the IOCS are used only at the beginning and end of tape reels, and when tape errors or unusual conditions occur.

As explained in Part I of this manual, one Channel Scheduler is required for each tape channel used in the program and one File Scheduler is required for each tape file processed by the program. All File Schedulers for files which share a tape channel operate in conjunction with the Channel Scheduler for that channel; control passes back and forth among the elements that share the same tape channel, but there is no communication with corresponding elements of files using other tape channels.

OPERATION OF SCHEDULERS

Figure 7 is a simplified flow diagram of one File Scheduler and one Channel Scheduler. The two types of schedulers will be described together because control passes from one to the other so frequently that the operation of each would be vague if they were described separately. It is recommended that the flow diagram be used in conjunction with the description of the schedulers.

The Channel Scheduler is always entered from one of the File Schedulers. A File Scheduler may be entered in one of the following modes:

1. In the priority mode, after a tape operation has been completed and a priority condition has been indicated.

2. In the normal (non-priority) mode, when an IOCS macro-instruction (e.g., PUT) refers to an unavailable area. This condition occurs when all records in an input block have been processed, or when all spaces in an output block have been filled with output records.

The operation of the schedulers when entered in each of these modes is described separately below. Operation in both modes is effected by two counters and two program switches in each File Scheduler. Frequent references to these items occur in the descriptions which follow. Therefore, they are explained here rather than as part of the operation in each mode. The counters and switches are:

Availability Counter	This counter indicates the number of tape record areas that are ready for use. For an input file, it is the number of areas that contain input records to be processed; for an output file, it is the number of "empty" areas that are ready to receive output records.
Availability Switch	This switch is turned ON whenever at least one area is available. It is turned OFF when no area is available; i.e., when the Availability Counter is zero.
Pending Counter	This counter keeps a record of the number of tape operations that are waiting to be started. For an input file, it is the number of areas which contain records that have already been processed. New input blocks must be read into this number of areas. For an output file, it is the number of areas filled with output records that have not yet been written on the output tape.
Pending Switch	This switch is turned ON whenever at least one area requires a tape operation. It is turned OFF when all areas are ready for use; i.e., when the Pending Counter is zero.

The two switches appear on the flow diagram (Figure 7); the two counters do not. Although the counters are not shown, incrementing and decrementing them is a function of the File Scheduler, and is indicated on the flow diagram.

Priority Mode

When a priority condition signals the end of a tape operation, the appropriate File Scheduler is entered in the priority mode. A check is made to determine that the tape operation just completed is correct; i.e., no unusual conditions or tape errors have occured. If the tape operation is correct, the Channel Scheduler will be entered; if not, the error routines will be executed.

The Channel Scheduler saves the contents of accumulator 1 so that the schedulers may use the accumulator. Next, the SPOOL Switch determines whether control should pass to a SPOOL program or continue in the Channel Scheduler. The SPOOL

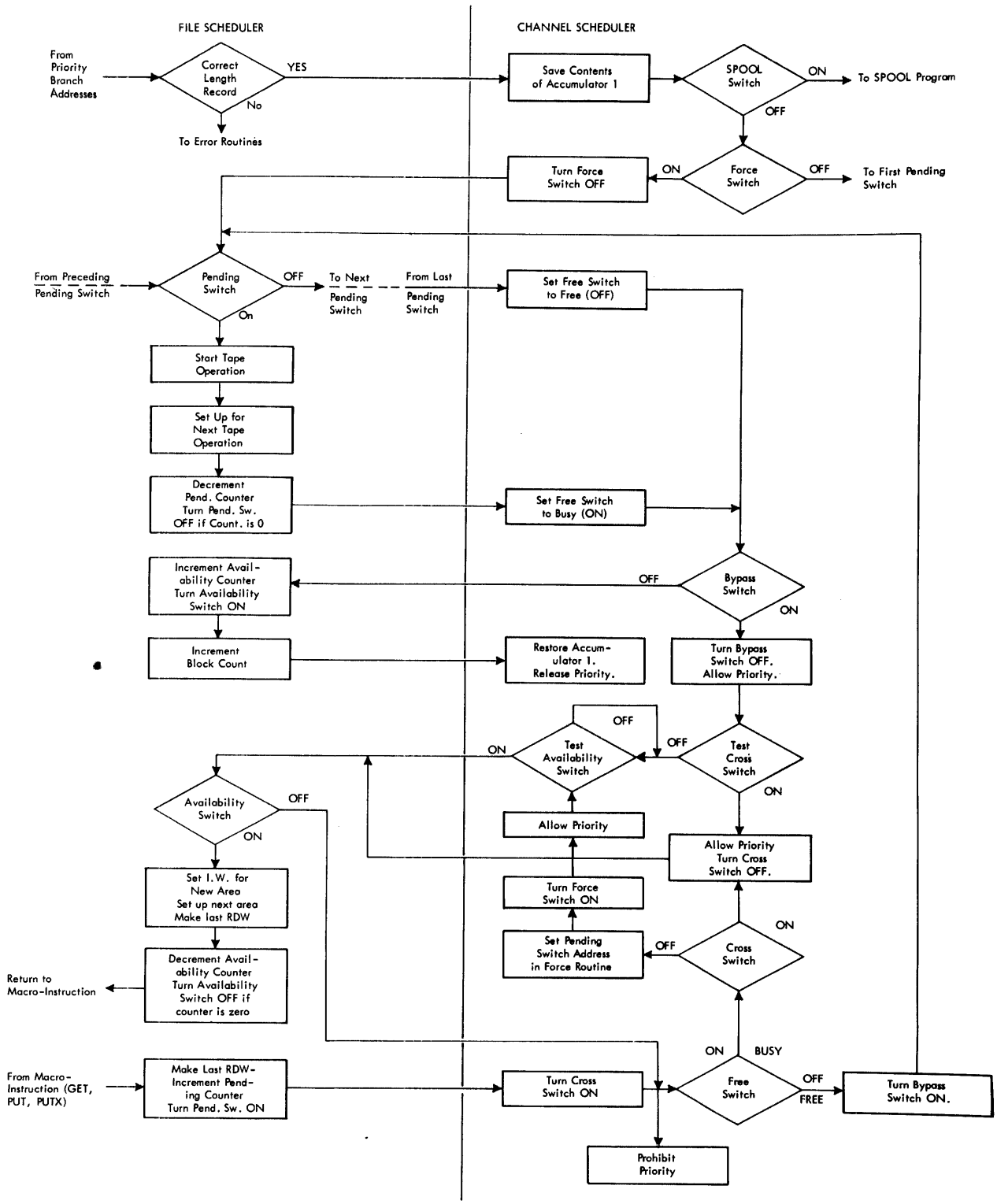

FILE SCHEDULER
CHANNEL SCHEDULER
From Priority Branch Addresses
Correct Length Record
YES
No
To Error Routines
Save Contents of Accumulator 1
SPOOL Switch
ON
To SPOOL Program
OFF
Turn Force Switch OFF
ON
Force Switch
OFF
To First Pending Switch
From Preceding Pending Switch
Pending Switch
OFF
On
To Next Pending Switch
From Last Pending Switch
Set Free Switch to Free (OFF)
Start Tape Operation
Set Up for Next Tape Operation
Decrement Pend. Counter Turn Pend. Sw. OFF if Count. is 0
Set Free Switch to Busy (ON)
Increment Availability Counter Turn Availability Switch ON
OFF
Bypass Switch
ON
Increment Block Count
Restore Accumulator 1. Release Priority.
Turn Bypass Switch OFF. Allow Priority.
OFF
Test Availability Switch
ON
OFF
Test Cross Switch
ON
Availability Switch
OFF
ON
Allow Priority
Allow Priority Turn Cross Switch OFF.
Set I.W. for New Area Set up next area Make last RDW
Turn Force Switch ON
ON
Set Pending Switch Address in Force Routine
OFF
Cross Switch
Return to Macro-Instruction
Decrement Availability Counter Turn Availability Switch OFF if counter is zero
ON
BUSY
From Macro-Instruction (GET, PUT, PUTX)
Make Last RDW-Increment Pending Counter Turn Pend. Sw. ON
Turn Cross Switch ON
Free Switch
OFF
FREE
Turn Bypass Switch ON.
Prohibit Priority

Figure 7

Switch is located at this point in the Channel Scheduler so that a tape-limited main program cannot slow down or stop a SPOOL operation by dominating the tape channel. An electronic switch is used for the SPOOL Switch; IOCS uses electronic switch 30 for SPOOL programs operating with a tape unit on tape channel 1, and electronic switch 29 for SPOOL programs using tape channel 2. The SPOOL Switch determines the action as follows:

ON A SPOOL tape operation is waiting or is in progress. Control will be passed to the SPOOL program.

OFF The SPOOL program does not need the use of the tape channel. Control will remain in the Channel Scheduler.

If the SPOOL Switch is OFF, the Force Switch determines the method of returning to a File Scheduler. The Force Switch indicates that the Pending Switch of each File Scheduler is to be checked in sequence or that control is to pass to a specific Pending Switch. A program switch (i.e., an instruction that is changed between NOP and B) is used as the Force Switch and determines the action as follows:

ON (NOP) A specific File Scheduler requires the use of the tape channel. Control passes to the Pending Switch of that File Scheduler after turning the Force Switch OFF.

OFF (B) The Pending Switches of files using this channel are to be tested in the order specified by the PRIORITY entry (Line 18) of the File Specifications. Control will pass to the Pending Switch of the File Scheduler for the file having the highest priority on this channel.

Upon reaching a Pending Switch that is ON, either as a result of testing the Pending Switches in sequence or as a result of the Force Switch being ON, a tape operation for the corresponding tape file is initiated, and the next area is set up for subsequent tape operations. The Pending Counter is decremented to indicate that a pending operation has been started. If the Pending Counter is zero, the Pending Switch will be turned OFF, and control returned to the Channel Scheduler.

The Free Switch is turned ON to indicate that the tape channel is busy; i.e., it is being used by the tape operation started in the File Scheduler. The function of the Free Switch will be described in detail later as it is encountered in the Channel Scheduler. If the Pending Switch of each File Scheduler is OFF, control will pass to the Channel Scheduler which turns the Free Switch OFF. After the Free Switch has been turned either ON or OFF, the action taken depends on the Bypass Switch. The Bypass Switch is a program switch that indicates the action as follows:

ON (NOP) The schedulers were entered in the normal (non-priority) mode. Operation in this mode is described under "Normal Mode" below.

OFF (B) The schedulers were entered in the priority mode. Control will return to the File Scheduler which started the last tape operation on this channel.

When the File Scheduler is entered from the Bypass Switch, the Availability Counter will be incremented and the Availability Switch will be turned ON. The block count for the corresponding tape file will be incremented and control will be returned to the Channel Scheduler.

The Channel Scheduler will then restore the contents of accumulator 1; the contents were saved when the Channel Scheduler was first entered in the priority mode. After accumulator 1 has been restored, the Channel Scheduler will execute a Priority Release instruction and return control to the main program. This sequence of operations will occur each time a File Scheduler is entered in the priority mode as the result of a completed tape operation.

Normal Mode

A File Scheduler is entered in the normal (non-priority) mode whenever a macro-instruction refers to a Tape Record Area that has already been used; i.e., all records in an input block have been processed or all spaces in an output block have been filled. The entry in the normal mode is shown at the lower left-hand corner of the flow diagram. After entering the File Scheduler, the sign of the last RDW that defines the used Tape Record Area is set to minus; the Pending Counter is incremented, the Pending Switch is turned ON, and control passes to the Channel Scheduler.

Upon entering the Channel Scheduler, the Cross Switch is turned ON. The function of the Cross Switch will be described in detail later, when it is used. The next action to be followed depends on the setting of the Free Switch. The Free Switch is a program switch that indicates the action as follows:

ON (B) The tape channel is busy. A tape operation may not be started at this time so the scheduler branches to the Cross Switch.

OFF (NOP) The tape channel is free. A tape operation for the tape file named in the macro-instruction can be started.

If the Free Switch is ON, the action to be taken depends on the Cross Switch. The Cross Switch is a program switch that indicates the action as follows:

ON (B) No tape operation can be started. The Availability Switch will be checked to see if a Tape Record Area is ready for use by the main program.

OFF (NOP) No tape operation can be started and no Tape Record Area is available to the main program. The Channel Scheduler will be conditioned to force a tape operation for the tape file named in the macro-instruction.

As stated above, the Cross Switch is turned ON as soon as the Channel Scheduler is entered. If the tape channel is busy (Free Switch ON), the Cross Switch will return control to the File Scheduler after turning the Cross Switch OFF. The action occurring in the File Scheduler depends on the Availability Switch.

If the Availability Switch is ON, the index word is set for the available area and the next area is set up for subsequent operations. The sign of the last RDW in the available area to be used is changed to plus. Next the Availability Counter is decremented and, if it becomes zero, the Availability Switch will be turned OFF. Control then returns to the macro-instruction in the main program that sent control to the File Scheduler. If the Availability Switch is OFF, control returns to the Free Switch in the Channel Scheduler. If the tape channel is still busy, the Cross Switch (which is now OFF) will set up the force routine to indicate the tape file that has no area available. The Force Switch is then turned ON so that the next tape operation to be started will be for the appropriate tape file. The function of the Force Switch is described under "Priority Mode." The Channel Scheduler then tests the status of the Availability Switch and, if OFF, repeats the test. (The Availability Switch will remain OFF until the forced tape operation is completed.) The test must be repeated to form a program loop so that a priority condition can cause an entry into a File Scheduler and allow the Force Switch to take effect. As soon as the test indicates that the Availability Switch is ON, control will pass to the Availability Switch in the File Scheduler. Subsequent operations are the same as explained in the preceding paragraph.

The Free Switch determines the action to be taken if the Availability Switch is OFF, as well as the action just after the macro-instruction transfers control to the File Scheduler. The action occurring if the Free Switch is ON (BUSY) is described above; if the Free Switch is OFF (FREE), the Channel Scheduler will turn the Bypass Switch ON and set a priority mask to prohibit priority conditions. Control then passes to the Pending Switch in the File Scheduler that was entered from the macro-instruction. The operations performed in the File Scheduler at this time are the same as those that occur when in the priority mode. These operations are described under "Priority Mode."

Control is then returned to the Channel Scheduler, where the Free Switch is turned ON. The Bypass Switch, which was turned ON before leaving the Channel Scheduler, indicates that control is to remain in the Channel Scheduler. A priority mask to allow priority conditions is set and the Bypass Switch is turned OFF. The status of the Cross Switch is tested to determine the next operation. If the text indicates that the Cross Switch is OFF, repeated tests of the Availability Switch will be made until it is turned ON; the Cross Switch being OFF indicates that previously the channel had been busy and the Availability Switch had been OFF. If the test indicates that the Cross Switch is ON, it will be turned OFF and control will pass to the Availability Switch in the File Scheduler; the Cross Switch being ON indicates that the channel was free when the Channel Scheduler was first entered. Operations occurring after the Availability Switch is reached are the same as those described previously.

Printed in U.S.A.

C28-6175-2

IBM®
International Business Machines Corporation
Data Processing Division, 112 East Post Road, White Plains, N.Y. 10601