File No. 1401/1460-30
Form C24-1489-3

IBM **Systems Reference Library**

# Input/Output Control System (on Disk) Specifications IBM 1401 and 1460

This publication presents the specifications for the *IBM 1401/1460 IOCS (on Disk) Program*, Version 2.

It describes the programming required to use IOCS to control the input/output of data from card reader, card punch, printer, disk, and tape files. The IOCS descriptive entries (DIOCS and DTF) and macro instructions are explained in detail. The types of processing and types of records handled by IOCS are defined.

The IBM 1401/1460 IOCS (on Disk) program is a supplement to the IBM 1401/1440/1460 Autocoder (on Disk) program. The reader should be familiar with the specifications for this program, described in the SRL publication, *Autocoder (on Disk) Program Specifications and Operating Procedures, IBM 1401, 1440, and 1460*, C24-3259. For a more complete understanding of the organization of records on disk, he should also review the disk file-organization routines.

The SRL publication *IBM 1401/1460 Bibliography, Form A24-1495*, lists publications available for other programming systems.

*Minor Revision (November 1964)*

This edition, C24-1489-3, is a minor revision of the previous edition, C24-1489-2, and incorporates Technical Newsletter N24-0200. The only changes are minor clarifications on pages 9, 28, 50, and 51.



Copies of this and other IBM publications can be obtained through IBM Branch Offices.
Address comments regarding the content of this publication to IBM Product Publications, Endicott, New York

# Contents

The IBM 1401/1460 Input/Output Control System eliminates the need for detailed programming of standardized input and output routines when a source program is written in *Autocoder* language. IOCS requires descriptive entries and macro instructions in addition to those used by the *Autocoder* program. With these, the user has access to library routines for reading and writing, blocking and deblocking, file labeling, and error checking.

IOCS for the IBM 1401 and 1460 is not a complete program in itself, but rather is a set of library routines that supplement the 1401/1440/1460 *Autocoder* program. These IOCS routines are selected and tailored automatically by the *Autocoder* processor to satisfy the particular requirements of each job. *Autocoder* generates the instructions needed according to the detailed information that the user supplies in the descriptive entries.

IOCS handles files in the:

IBM 1301 Disk Storage, Models 11, 12, 21, and 22
IBM 1311 Disk Storage Drive
IBM 729II, 729IV, 729V, or 7330 Magnetic Tape Units
IBM 1402 Card Read-Punch
IBM 1403 and 1404 Printers

The disk or tape records in the files may be:

- fixed-length, unblocked (Form 1)
- fixed-length, blocked (Form 2)
- variable-length, blocked (Form 4)

Also records on *tape* may be variable-length, unblocked (Form 3). Card and printer records are considered fixed-length, unblocked.

IBM 1401 or 1460 source programs that include IOCS statements can be assembled on an IBM 1401 or 1460 system or on an IBM 1440 system, provided the 1401/1440/1460 *Autocoder* library contains the IOCS routines specifically designed for operation with the 1401 and 1460. (The SRL publication, *Autocoder (on Disk) Program Specifications and Operating Procedures, IBM 1401, 1440, and 1460*, Form C24-3259, describes the procedure for adding routines to the *Autocoder* library and gives the specifications for the disk file on which the library resides.)

An IBM 1401 system that is used to assemble programs with IOCS must have at least:

- 4,000 positions of core storage
- One IBM 1311 Disk Storage Drive, Model 4
- One IBM 1402 Card Read-Punch, Model 1
- One IBM 1403, Model 1 or 2, or 1404 Printer
- High-Low-Equal Compare Special Feature

An IBM 1460 system that is used to assemble programs with IOCS must have at least:

- Core storage in any capacity available
- One IBM 1301 Disk Storage, Model 11 or 12, *or* one IBM 1311 Disk Storage Drive, Model 1
- One IBM 1402 Card Read-Punch, Model 3
- One IBM 1403 Printer, Model 2 or 3

An IBM 1440 system that is used to assemble programs with IOCS must have at least:

- 4,000 positions of core storage
- One IBM 1301 Disk Storage, Model 11 or 12, *or* one IBM 1311 Disk Storage Drive, Model 1
- One IBM 1442 Card Read-Punch, Model 1 or 2, *or* one IBM 1442 Card Reader, Model 4.
- IBM 1443 Printer, Model 1 or 2

An IBM 1401 system on which the resultant object program is run must have the High-Low-Equal Compare special feature. The Advanced Programming special feature (1401) or the Indexing and Store Address Register special feature (1460) is required when:

- Blocked records are processed.
- Unblocked records are processed in work areas.
- Card or printer operations are overlapped with disk-seek operations.

The Advanced Programming, or Indexing and Store Address Register, special feature is not required if unblocked records are processed in the input/output area and card or printer operations are not overlapped with disk-seek operations.

## General Description

The IOCS descriptive entries describe the logical files used in a program. A logical file is a file of like data, such as accounts-receivable records, in which all records have the same format. When a logical file is recorded on disk, it may fit in all or part of one 1301 disk module or one 1311 disk pack (possibly with other logical files in the same module or pack). Or, it may require two or more disk modules or packs. A logical file must be completely recorded on one *type* (1301 or 1311) of disk unit, however. It must *not* be recorded partly on 1311 disk packs and partly on 1301 disk units.

When a logical file is recorded on tape, it may fit on all or part of one tape, or may require two or more reels.

Two types of descriptive entries are required for an IOCS program:

DIOCS "Descriptive IOCS" that describes *generally* all the files to be processed, and the machine configuration on which the object program is to be run (Figure 1).

| Label | Operation | OPERAND |
|---|---|---|
| | DIOCS | |
| DIOCS.ORG | | 700 |
| IODEVICES | | READER,PUNCH,TAPE |
| TAPEUSE | | INPUT |
| FEATURES | | RELEASE |
| LABELDEF | | STANDARD,A |
| ALTDRIVE | | YES |
| EXITS | | YES |
| READERROR | | TAPE,4 |
| CHECKPOINT | | YES |

Figure 1. Sample DIOCS entry

DTF "Define the File" that describes an individual file in *detail* (Figure 2). A DTF entry must be included for each logical file processed by IOCS. This file may be recorded in punched cards, on tape, or on disk, or it may be printed out on continuous forms.

| Label | Operation | OPERAND |
|---|---|---|
| | DTF | FILEA |
| FILETYPE | | TAPE,,INPUT |
| MODEPAR | | LOAD |
| CHANDRIVE | | 2 |
| ALTTAPE | | 5 |
| RECFORM | | BLOCKED |
| SIZEREC | | 120 |
| BLOCKSIZE | | 600 |
| IOAREAS | | INPUTA |
| WORKAREA | | WORKA |
| EOFADDR | | ENDJOB |
| EXZADDR | | HEAD2 |
| TYPELABEL | | STANDARD,,A |
| CHECKLABEL | | ALL |
| HEADER | | MO PAYROLL,,62074,030 |
| SERIALNUM | | 54003 |
| WLRADDR | | RECERR |

Figure 2. Sample DTF Entry

The DIOCS and DTF entries are punched in *Autocoder* cards. Each entry consists of a set of cards: a header card followed by several detail cards. The number of detail cards is governed by the factors that must be specified for a particular job. These entries are explained in detail under *Descriptive Entries*. The DIOCS and DTF cards are assembled with the source program. They are inserted, in that order, immediately behind the AUTOCODER RUN, JOB, and CTL cards, and ahead of the user's source program.

The IOCS macro instructions are entered in the user's source program. They provide linkage to the IOCS library routines that read, write, block, deblock, and check records without further programming on the part of the user. Four basic macro instructions are:

GET – transfers a record from a file (tape, disk, or unit record) to an input or work area where it can be processed.

PUT – transfers a record from an output or work area to a file (tape, disk, unit-record, or printer).

OPEN – activates a file for processing.

CLOSE – deactivates a file after processing is completed.

Other macro instructions used by the IOCS are: RELSE (release), FEORL (Forced End-of-Reel), DCLOS (Dump-Close), RDLIN (Read Label Information), SEEK, SCAN, and SPACE/SKIP. The macros are punched in *Autocoder* cards and assembled with the source program.

IBM standard header and trailer labels may be processed automatically by IOCS. Therefore, it is to the user's advantage to include them whenever possible. However, records, with no header and trailer labels or tape records with nonstandard labels can be processed. In this case, the user must write the program instructions to process his labels.

IOCS transfers *tape* records with word marks (load mode) or without word marks (move mode). IOCS transfers *disk* records without word marks (move mode) in any of the IBM 1401/1460 modes of disk operation:

- Sector – data transferred by sector (100 characters). As many sectors of data as desired can be transferred by one insertion.
- Track Sectors w/Addresses – data and sector addresses transferred by track (2000 data characters and 120 address digits). One complete track is transferred by one instruction.
- Track Record – data transferred by track with this special feature installed (2980 data characters). One full track of data is transferred by one instruction.
- Track Record w/Address – data and address transferred by track with the Track Record special feature installed (2980 data characters and 6 address digits). One full track is transferred by one instruction.

Files recorded in the *track record* modes may be used in the same program with other files recorded in the *sector* mode. However, if a program uses the *track sector w/addresses* mode of operation, *all* files must be recorded in this mode.

### Processing Overlap Special Feature

When this special feature is installed in the IBM 1401 or 1460 Data Processing System, card, printer, and tape records can be read, written, and punched in the *overlap mode* by IOCS. This feature provides a reduction in over-all job time by efficient use of the high-speed processing and input/output capabilities of the 1401/1460. It is described in detail in the SRL publication, *IBM 1401/1460 Special Features, A24-3071.*

Whenever this feature is used in conjunction with IOCS, the special programming and timing considerations of overlap are completely absorbed by IOCS. Therefore, throughout his program, the user considers the records as being read or written in the *non-overlap mode*. To provide for this, he specifies an *overlap* operation in the DIOCS entry (FEATURES). He must also plan for this feature in the specifications of certain DTF entries and in the allocation of input/output and work areas in storage for certain types of records. To gain the greatest time saving advantage of this feature, two input/output areas should be allotted when tape records are processed. Card records require a work area, and they cannot utilize the read release or punch release special feature. Cards in an input card file can not be selected. All cards in the card reader stack in the normal pocket. For printer operations, an additional macro instruction (SPACE/SKIP) is provided.

The processing overlap special feature does not permit disk input/output operations to be executed in the *overlap mode*.

Throughout this bulletin, *overlap mode* is indicated whenever the processing overlap special feature affects a specification.

#### Overlap with Seek Time

To use disk-seek time efficiently, IOCS can overlap card-reader, card-punch, and printer operations with disk-seek operations. For this, the programmer specifies reader (RDROVLP), punch (PCHOVLP), or printer (PRNTOVLP) overlap in the DIOCS entries. Any non-file operation can be executed during disk-seek time by using the SEEK macro instruction. The processing overlap special feature is not required for these types of overlap.

#### Autocoder Loader

The *Autocoder* loader, which is used to load an object program into core storage, requires 205 storage positions. The loader is automatically stored in positions 1-205 if no other storage positions are specified in the *Autocoder* control (CTL) card. Of these 205 positions, the first 80 are the card-input area, which is used for reading the object-program cards. The remaining 125 positions are required for the loading instructions. By specifying a starting address (above 81) in the *Autocoder* control card, the programmer can use other storage positions for the loading instructions, if desired.

The loading instructions must be stored so that they do *not* conflict with the fixed output areas, with the index registers, or with today's date in storage (used for labeled tape or disk files) whenever any of these features is used in the source program or by IOCS. If the Advanced Programming special feature is specified in the *Autocoder* control card, IOCS uses the index registers. Therefore the loading instructions must start in storage position 100 or above. If a card-output file is included in the program and the Advanced Programming feature is specified, IOCS inserts a record mark in position 181, and the loading instructions must start in position 182 or above.

The lowest possible starting position for the *Autocoder* loading instructions is position 81 without advanced programming, and position 100 with advanced programming. When any of the following items is specified, however, the lowest starting position is:

| | *Without Adv. Prog.* | *With Adv. Prog.* |
|---|---|---|
| Labeled tape or disk file | 87 | 100 |
| Card-output file | 181 | 182 |
| Printer file | 333 | 333 |

### Types of Processing — Disk Records

The IOCS routines process disk records in random order, consecutive order, or control-sequential order.

*Random Processing* — generally applies to records *organized* in random order. Records stored at any locations within the file can be processed at any time in the program. Before a record can be located for processing, its disk address must be determined. This disk address is usually based on the control data of the record and computed by a predetermined formula used in the organization system applied to create the file.

Records *organized* in control-sequential order (or in an order unique to a particular application) can be *processed* in random order, only if the disk addresses of the records can be determined.

*Consecutive Processing* — used to process successive records on disk. Records are processed starting with a beginning disk address for the logical file and continuing in order, from sector to sector (or track to track), to the end of the logical file on disk. When data or data-with-addresses are to be transferred by track (rather than by sector), the file must be organized so that the transfer starts at the same sector location on each track. That is, addresses must be assigned in increments of twenty. Records organized in either random order or control-sequential order (or in an order unique to a particular application) can be processed consecutively if desired.

When records are processed in consecutive order, a complete logical file may be treated as one *file area*, or the file may be split and handled in two or more *file areas*. A file or a part of a file is defined as a *file area* whenever it meets *both* these requirements:

1. The records are on-line. For example, if a logical file requires several packs and the system has two IBM 1311 Disk Storage Drives available for the file, each *area* of the file can consist of two packs. When one or more 1301 modules are used, the disk records are always on-line.

2. The records contain consecutive addresses (excluding header labels, if any). If records in a file are on-line (satisfying the first requirement) and if they are written on three 1311 disk packs with addresses 000000-019999, 020000-039999, and 060000-079999, for example, the first two packs constitute one file area and the third pack constitutes another file area.

*Control-Sequential Processing* — applies only to input records organized according to the control-sequential method of file organization. In this type of organization, records are arranged in sequential order by control data, such as customer number. The sequential records are stored successively on disk (from sector to sector) wherever possible. When a sequential record is not stored in successive order, a linkage is provided to locate that record for processing. This out-of-order condition may have occurred, for example, when additions were made to a file after it was originally set up. The last field of each record is a *linkage* field that gives the address of the next sequential record if that record is filed out of order. This is a six-digit field for unblocked records, and a seven-digit field for blocked records. The seventh digit specifies the number of the record within the block. The linkage field in the out-of-order record provides the address of the next record in sequence. A record that will be addressed from a linkage field must be on-line at that time.

Logical files processed in random order, consecutive order, or control-sequential order may be included in the same program.

## *Disk Records and Storage Areas*

When files are recorded on disk, IOCS can process records that are fixed-length unblocked (Form 1), fixed-length blocked (Form 2), or variable-length blocked (Form 4). IOCS does not process variable-length unblocked records (Form 3). By proper file specifications in the DTF entries (SIZEREC and NSECTORS), IOCS can handle any fixed-length unblocked records containing 5 or more characters. For blocked records, the minimum size is 5 characters and the maximum is 999 characters.

Unblocked records are read from disk to core storage, or written on disk from storage, one record at a time. Each record in an unblocked file written in the sector mode of operation may fit in one or more sectors (Figure 3A, B).

Blocked records (Figure 3C, D) are read into core storage from disk, or written on disk, two or more records at a time. The number of records planned for a block depends on the size of the records, the most efficient use of the 100-character disk sectors, and the amount of core storage that can be reserved for the block. All records within a block may be fixed-length or variable-length—the records differ in length. Whenever variable-length blocked records are to be processed in the sector mode of operation, the sectors specified in the DTF entries must provide enough space for the largest record or block of data planned.

Although all the records in a given file must be the same type, IOCS can process several different-type files in the same program. The types of disk records that are handled by IOCS vary with the type of processing (random, consecutive, or control-sequential). When records in an input file are to be processed in random order, either fixed-length unblocked records or fixed-length blocked records may be specified for that file (Figure 4). The records in a random output file, however, must always be specified as fixed-length unblocked. When input or output records are to be processed in consecutive order, fixed-length unblocked records, fixed-length blocked records, or variable-length blocked records may be specified for a logical file. When input records are to be processed in control-sequential order, fixed-length unblocked or fixed-length blocked records may be specified.

#### REQUIREMENTS OF BLOCKED RECORDS

Several basic requirements of the records themselves must be met, in order to handle blocked disk records automatically by IOCS:

1. A block may contain a maximum of ten records with random processing, thirty with control-sequential processing, or one hundred with consecutive processing.
2. Each record in every block must contain a record mark as its last character whenever the records are to be processed in a work area. In this case, a record mark must not appear as a character anywhere else within the records. When records are processed in the input or output area, record marks are not required.
3. For variable-length records, a block-length field must be included in each block, and a record-length field in each record (see Figure 3D).

   As the name implies, *block length* is the total number of characters in the block, including itself and record marks. The block-length field must always be recorded in the first four positions of the block. The units position of the field must contain AB bits. When output records are created by IOCS, this count and the AB bits are generated automatically. The programmer must set a word mark in the high-order storage position of this field.

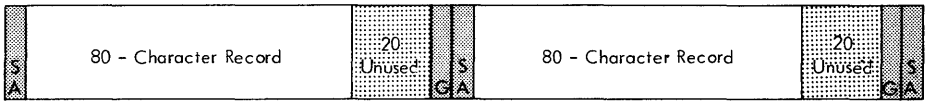
   *Record length* is the total number of characters in the record, including itself and the record mark. The record-length field is a 3-position field and must be located in the same three positions within each record in the file.

   IOCS uses the record length to locate the beginning of the next consecutive record. For this purpose, the low-order position of the field within the record must be specified in the DTF entries (SIZEREC). For example, the 15th position is specified if the record-length field is located in positions 13, 14, and 15 in each record. Furthermore, the record-length field must be defined by a word mark in the work area(s) specified for the file. When output disk records are created, the programmer must develop the length of the record to be entered in this field. Unlike block length, this is not developed automatically by IOCS routines.

#### Storage Areas

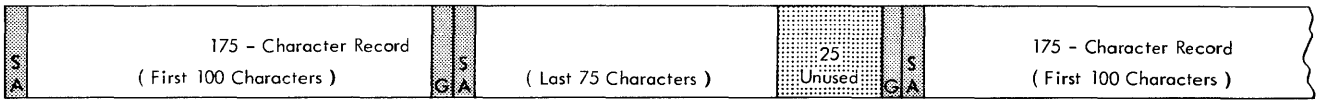
An input (or output) area must be allotted for each logical file used in operation. Individual records are processed directly in this area, or they are moved to work areas for processing. The advisability of using

FILE A. 80-CHARACTER UNBLOCKED RECORDS



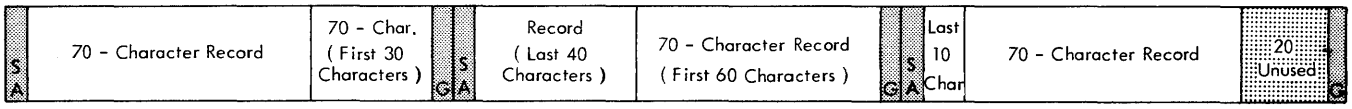

DTF RECFORM - UNBLOCKED,FIXED
DTF SIZEREC - 80
DTF NSECTORS - 1

FILE B. 175-CHARACTER UNBLOCKED RECORDS



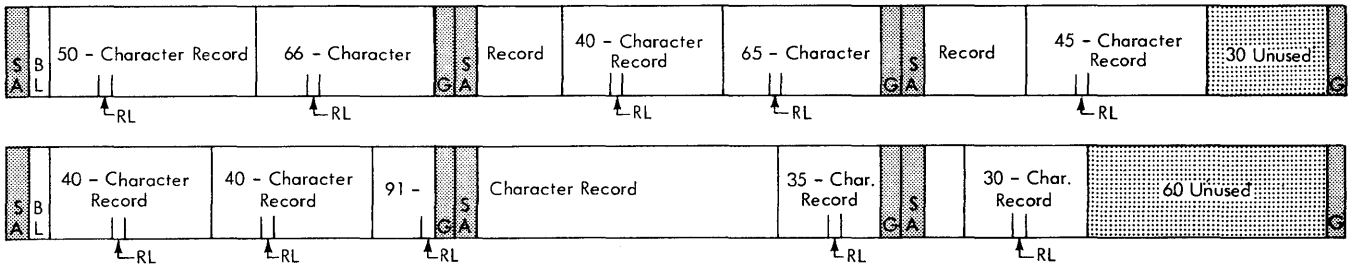

DTF RECFORM - UNBLOCKED, FIXED
DTF SIZEREC - 175
DTF NSECTORS - 2

FILE C. 70-CHARACTER RECORDS, BLOCKED 4 TO A BLOCK




DTF RECFORM - BLOCKED, FIXED
DTF SIZEREC - 70
DTF NSECTORS - 3
DTF NRECORDS - 3 ( Blocking Factor Minus One )

FILE D. VARIABLE-LENGTH BLOCKED RECORDS (LARGEST BLOCK - 300 CHARACTERS; LARGEST RECORD - 296 CHARACTERS)




DTF RECFORM - BLOCKED, VARIABLE
DTF SIZEREC - 15 (Low-Order Position of Record-Length Field )
DTF NSECTORS - 3

SA - Sector Address
G - Gap Between Sectors
BL - Block-Length Field
RL - Record-Length Field

Figure 3. Schematic of Records on Disk

work areas is affected by the complexities of the user's program and by the type of records to be handled. The programmer must predetermine the area to be used for processing and specify the proper DIOCS and DTF entries.

*Unblocked disk records* may be processed in either area.

*Fixed-length blocked records* may be processed in the input/output area, if an index register is assigned to the file to locate the individual records. Otherwise they must be processed in work areas. One method or the other must be specified for the particular file.

*Variable-length blocked records* require work areas. Indexing cannot be used to locate the individual records.

Any input, output, or work area that is set up for use by IOCS must be labelled and defined by a DA statement. The area in which records are processed should contain word marks to define the individual fields of data, as in any 1401/1460 operation. The fields can be labelled and defined by the DA statement for the area.

Whenever index registers are installed and specified in the *Autocoder* control card, they are used by the IOCS routines. Therefore, within storage positions

| TYPE OF PROCESSING | TYPES OF RECORDS | | |
|---|---|---|---|
| | FIXED-LENGTH | | VARIABLE-LENGTH |
| | UNBLOCKED | BLOCKED | BLOCKED |
| RANDOM INPUT | X | X | |
| RANDOM OUTPUT | X | | |
| CONSECUTIVE INPUT or OUTPUT | X | X | X |
| CTL-SEQUENTIAL INPUT | X | X | |

Figure 4. Types of Disk Records and Processing

87-99, the programmer may set word marks *only* in positions 87, 92, and 97. He must not set others for his own use within that area.

INPUT/OUTPUT AREAS

An input or output storage area must be followed by a group-mark with word-mark. The same label (symbolic address) named in the DA statement for the area must be specified in the DTF entry (IOAREA). If an index register is used for fixed-length blocked records, it must be specified in the DTF entry (INDEXREG) and in the DA statement.

For a logical file on disk, the size of the input, or output, area to be reserved in storage is determined by the mode of operation used for that file. If the sector mode of operation is used, the amount of storage required to contain data for each disk read or write operation is based on the number of sectors specified in the DTF entries (NSECTORS) for a record or for a block of records. In all cases, ten positions must be allotted at the beginning of the area for the Disk Control Field. Thus, the size of the reserved area must always be equal to the sum of ten plus the number of characters available on disk in the mode used (Figure 5).

| I/O AREA | CORE STORAGE POSITIONS | | | |
|---|---|---|---|---|
| | SECTOR MODE | TRACK SECTORS W/ADDRESSES MODE | TRACK RECORD MODE | TRACK RECORD W/ADDRESS MODE |
| DISK CONTROL FIELD | 10 | 10 | 10 | 10 |
| DISK CHARACTERS | 100N | 2120 | 2980 | 2986 |
| TOTAL | 10 + 100N | 2130 | 2990 | 2996 |
| N = Number of Sectors | | | | |

**Figure 5. Storage Requirements for Input/Output Areas, Disk Records.**

The DA statement for the disk input, or output, area must be labelled, and it provides core-storage positions for the entire area by specifying the:

1. ten-position disk-control field.
2. data records (indexed and/or subdivided by fields, if required).
3. Extra character-positions available on disk in the modes of operation used (when the data records do not require all disk positions).
4. Group-mark with word-mark following the entire area.

For example, the input area DA statement (Figure 6) for the blocked fixed-length records shown in Figure 3C is based on:

A. 10-position disk control field
B. 70-character records, 4 to a block
C. Index Register 1 used
D. 3 sectors required for a block
E. Sector mode of operation
F. Input area labelled "PARLIN" (Payroll In)
G. Fields labelled and defined.

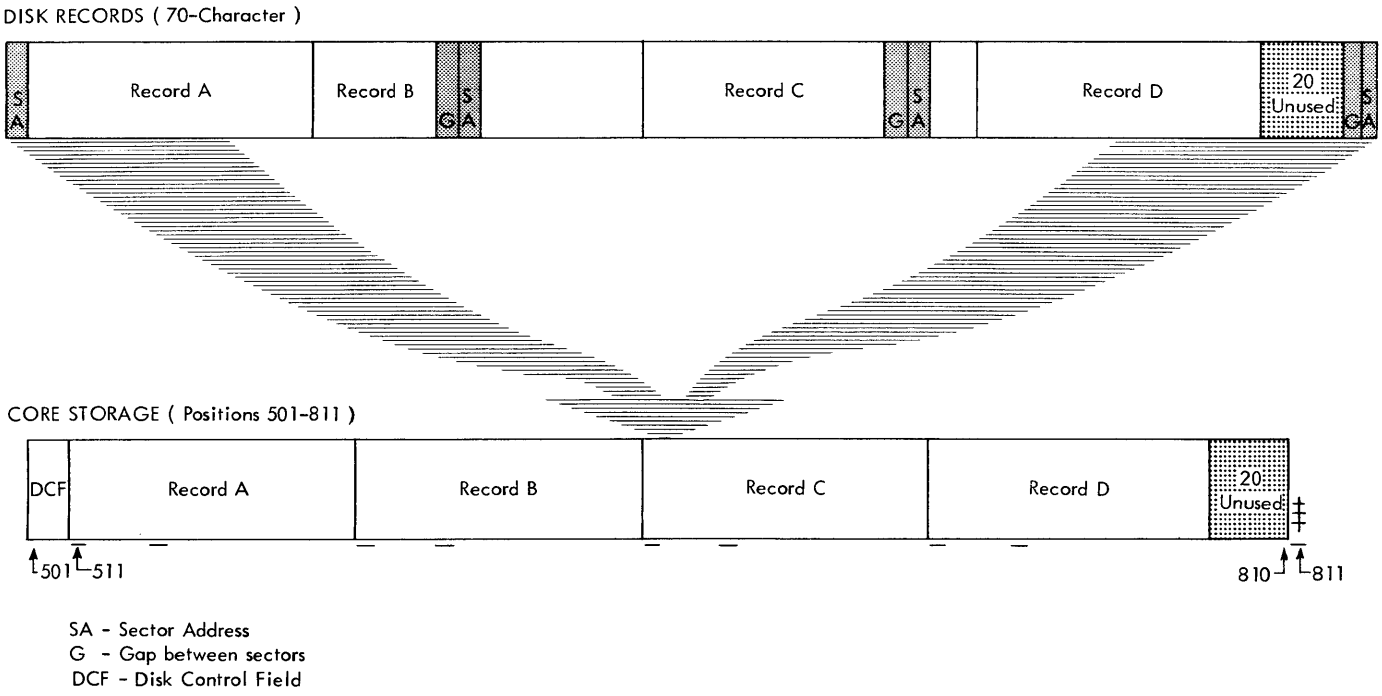
The last two entries in the illustration place the group-mark with word-mark in the position immediately following the 310 positions required for the entire area (see Figure 5). These two entries could be replaced by one entry (DA 1X20,G) by computing that 20 available disk positions remain after the area has been reserved for the disk control field (10 positions) and the data records (280 positions).

WORK AREAS

Work areas are used with blocked records to separate one record from the block, thus making it possible to locate the individual fields in that record for processing. Therefore, the work area must be equivalent in length to the size of a a single record. For variable-length records, the area must provide for the largest record.

One work area may be assigned to a file when disk records are processed in random or control-sequential order. One, two, or more work areas may be assigned when disk records are processed in consecutive order. The label named in the DA statement for a work area is specified in the DTF entry (WORKAREA) if one work area is used for a file. Whenever two or more work areas are used, however, the label must be specified in the GET or PUT macro instructions, instead of the DTF entry.

Any *blocked* disk records that are processed in a work area, rather than the input/output area, must contain a record mark as the last character in the record. Whenever *unblocked* disk records that do not contain record marks are processed in a work area, that work area must be equivalent in size to the data portion of the input/output area (see Figure 5, *Disk Characters*). Also, the work area must be followed by a group-mark with word-mark.




DA STATEMENT

| Line | Label | Operation | OPERAND |
|---|---|---|---|
| 01 | PARLIN | DA | 1,X1,0 (A)(F) |
| 02 | | DA | 4,X7,0,,X1 (B)(C) |
| 03 | MANNO | | 1,,6 |
| 04 | FICA | | 17,,21 (G) |
| 05 | etc | | etc |
| 06 | | ORG | PARLIN (D)(E) |
| 07 | | DA | 1,X3,10,,6 |
| 08 | | | |
| 09 | | | NOTE: Circled letters refer to items in text |
| 10 | | | |
| 11 | | | |

Figure 6. Example of an Input Area and DA Statement, Disk Records

Whenever disk records are to be processed in a work area, a record mark may appear *only* as the last character in a record.

## *Tape, Card, and Printer Records and Storage Areas*

IOCS processes tape records that are blocked or unblocked and fixed-length or variable length. Although all the records in a given file must be the same type, IOCS can process several different types of files in the same operation. The four different combinations of blocking and length are illustrated in the *Schematic of Tape Records and Input Areas* (Figure 7). For each combination, the schematic illustrates an input area, several records read in, and the *Autocoder* DA statement for the area. The labels containing the letters DTF refer to the corresponding DTF entry specifications.

Whenever IOCS controls the input/output of any records, a record-mark may be included *only* for its normal use – to indicate the end of a record. It must not be used for *any* other purpose. When the Advanced Programming special feature is specified in the *Autocoder* control card, IOCS inserts a record mark in storage position 81 if a card-input file is specified and in 181 if a card-output file is specified.

Whenever index registers are installed and specified in the *Autocoder* control card, they are used by the IOCS routines. Therefore, within storage positions 87-99, the programmer may set word marks *only* in positions 87, 92, and 97. He must not set others for his own use within that area.

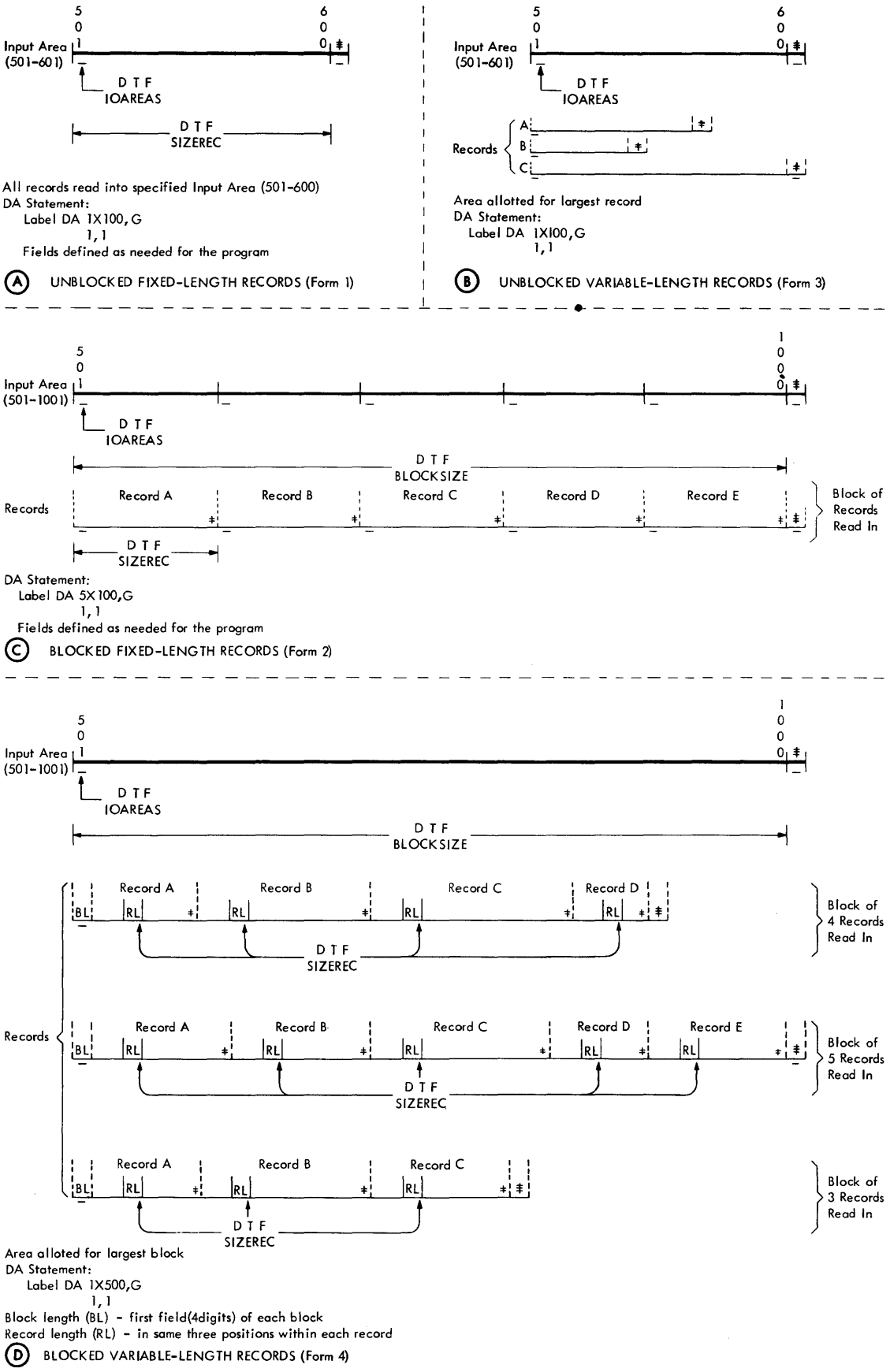



Figure 7. Schematic of Tape Records and Input Areas

### Unblocked Records

Unblocked records (Figure 7A, B) are read in, or written, one record at a time. This record type includes tape records, card input and output files, and printer operations.

INPUT/OUTPUT AREAS

Each input/output storage area allotted for these records should be equivalent in length to the size of a single record. In the case of variable-length records, the area must provide for the largest record.

For tape records, the input/output areas must be defined by DA statements and a group-mark with word-mark must follow the area. The label (symbolic address) of the DA statement must be included in the DTF entries (IOAREAS). Because unblocked records (with one input area specified) are normally processed in the input area and reference must be made to individual fields, word marks must be provided to define these fields, as in any 1401/1460 operation. The fields can be labeled and defined with high-order word marks by the DA statement. Or, if the tape contains word marks, they can be inserted by reading the tape record in the load mode.

When IOCS is to check the length of fixed-length input tape records (as specified by the DTF WLRADDR entry), one extra position must be allotted in the input area, for IOCS use. For example, if 100-character records are to be processed, the DA statement for the input area must specify 101 positions. After a record is read in from tape, the extra position is located between the data record and the group-mark with word-mark position. When a correct-length record has been read, this position contains a group mark that was generated when the tape interrecord gap was sensed. If a wrong-length record was read, this position may contain any character other than a group mark.

When an unblocked variable-length record is read in, IOCS automatically inserts a record mark immediately following the record in the input area. This replaces the group mark.

Unblocked tape records processed in the *overlap mode* may be processed in a work area, or in the input/output area if no work area is specified. When two input/output areas are specified (DTF IOAREAS), however, the records can be processed in these areas only if indexing is also DTF-specified.

Because card-read, card-punch, and printer input/output areas are fixed, they need not be defined or included in the DTF entries.

WORK AREAS

Generally, unblocked records (with one input/output area specified) do not require record work areas, as blocked records do, because fields can be readily defined and used directly in the input/output area. If a work area is desired, however, it must be defined by a DA statement. In this case the individual fields are labeled and defined by high-order word marks in this work area, rather than in the input/output areas. The label of the work area DA statement may be included in the DTF entry (WORKAREA) or referred to in the GET or PUT macro instructions.

The work area must be followed by a record mark or a group-mark with word-mark, if it is used for a *fixed-length* unblocked record that does not contain a record mark as its last character. Whenever IOCS is to move an unblocked *variable-length* record to or from a work area, that record *must* contain a record mark as its last character.

Card files that use the Read Release or Punch Release special feature, or that are processed in the *overlap mode,* require the use of a work area.

### Blocked Records

When blocked tape records are handled, two or more records at a time are read in from tape, or written on tape. The number of records in a block depends on the size of the records and the amount of storage that can be reserved for the block. The programmer must predetermine the record and block sizes and specify these in the proper DTF entries (SIZEREC and BLOCKSIZE). All records within a block may be the same length (fixed-length), or they may differ (variable-length). This affects the specifications written in the DTF entries (SIZEREC) and in the input, output, and work area DA statements.

When fixed-length blocked records (Figure 7C) are processed, the individual records and fields can be located by IOCS control in either of two ways:

1. By using the indexing feature to step over to the beginning of each record in the input area.
2. By moving each record, one at a time, to a work area.

One method or the other, but not both, can be specified. Therefore, in planning a job for fixed-length blocked records, the programmer must first determine whether he will process records directly in the input area or move them to one or more work areas for processing. For example, will he identify a control field, such as a part number, in the input area and compare it to another number, or will he move the whole record to a work area and then identify the individual control field within that area. The advisability of using work areas for fixed-length blocked records is affected by the complexities of the user's program to process data for the particular job being planned. Such factors as the amount of core storage used for the job and the index registers required for other functions must be considered. The IOCS routines for the input/output of data handle one method as readily as the other.

When variable-length blocked records (Figure 7D) are read or written, one or more work areas *must* be used. Indexing *cannot* be used to locate the individual records and fields.

REQUIREMENTS OF BLOCKED RECORDS

Several basic requirements of the records themselves must be met, to handle blocked tape records automatically by IOCS:

1. Each record in every block must contain a record mark as its last character. Therefore, the user must provide record marks in any records that will eventually be read or written by IOCS routines.
2. Fixed-length records must be padded so that all blocks are the same length. When output tape records are created by IOCS, they are automatically padded with blanks unless the user specifies some other character in the DTF entries (PADDING). Padded records are included in record counts and hash totals when these are specified (see *Control Totals*).
3. For variable-length records, a block-length field must be included in each block, and a record-length field in each record (Figure 7D).

   As the name implies, *block length* is the total number of characters in the block, including itself and record marks. The block-length field must always be recorded in the first four positions of the block. It is used by IOCS for a wrong-length-record check. The units position of the field must contain AB bits, and the high-order position must contain a word mark. When output tape records are created by IOCS, this count and the AB bits are generated automatically.

   *Record length* is the total number of characters in the record, including itself and the record mark, and is used to modify addresses. The record-length field is a 3-position field and must be located in the same positions within each record in the file. The location of the low-order position within the record must be specified in the DTF entries (SIZEREC).

   For example, the 15th position is specified if the record-length field is located in positions 13, 14, and 15 in each record. Furthermore, the high-order storage position of the record-length field must contain a word mark in the specified work area. When output tape records are created, the programmer must develop the length of the record to be entered in this field. Unlike block-length, this is not developed automatically by IOCS routines.

INPUT AREA

This area is equivalent in length to the size of the block of tape records. In the case of variable-length records, the allotted area must provide for the largest block. The area must be defined by a DA statement and followed by a group-mark with word-mark. The label of this DA statement must be included in the DTF entry (IOAREAS). The size of the input area (including the 4-position block-length field, if any) must be specified in the DTF entry (BLOCKSIZE). This does *not* include the group-mark with word-mark position or the extra position for record-length checking if that is specified in the DTF entry (WLRADDR).

When fixed-length records are processed in the input area (indexing specified) and reference must be made to individual fields, word marks must be provided to define these fields, as in any 1401 operation. The fields can be labelled and defined with high-order word marks by the DA statement. Or, if the tape contains word marks, they can be inserted by reading the tape record in the load mode. The index register used for this operation must be specified in the DTF entries (INDEXREG). It must also be specified in the DA statement if records are processed in the non-overlap mode. If they are processed in the *overlap mode,* the index register must be *omitted* from the DA statement.

If blocked records are processed in the *overlap mode,* two input areas may be specified in the DTF entry (IOAREAS) and defined by DA statements.

When IOCS is to check the length of a block of either fixed- or variable-length records (as specified by the DTF WLRADDR entry), one extra position must be allotted in the input area. For example, if blocks of five 100-character records are handled, 501 positions must be allotted, followed by a group-mark with word-mark. This can be specified by two DA statements for the area:

```
LABEL     DA     5X100
                 1,1
                 .
                 .
                 .
                 .
                 .
          DA     1X1,G
```

Field Definitions

After the block of records is read in from tape (fixed-length records or maximum-size variable-length records), the extra position is located between the last data record and the group-mark with word-mark position. When a correct-length record has been read, this position contains a group mark that was generated when the tape interrecord gap was sensed. If a wrong-length record was read, this position may contain any character other than a group mark.

OUTPUT AREA

The same principles as described for blocked-record input areas apply to output areas. In addition, consid-

eration must be given to the recording of record marks. Because newly developed records are written via the output area, record marks must be provided in this area. They can be specified in the output area DA statement for fixed-length records. For variable-length records, however, the locations of the record marks cannot be predetermined. Therefore the programmer must include program steps to insert a record mark immediately after each record.

WORK AREA

This area is equivalent in length to the size of an individual record. For variable-length records, the area must provide for the largest record. Each work area must be defined by a DA statement, and the label of this DA statement must be included in the DTF entry (WORKAREA) or referred to in the GET or PUT macro instructions. Because the work area is used for processing records, the DA statement should also label and define the individual fields with high-order word marks.

### Control Totals

The IOCS routines can provide three control totals for tape records with IBM standard labels: block count, record count, and hash total. These totals are accumulated during the run and may be recorded on (output tape) or checked against (input tape) the trailer label, according to the user's specifications. A block count is always taken, and it is checked or written when standard-label operation is specified. For unblocked records, this is the same as the record count. The record count and hash total are taken only if specified in the DTF entries (TOTALS). If a discrepancy is detected in any one of these totals when an input trailer is checked by IOCS (see DTF TOTALS), a programmed halt occurs.

When a hash total is to be taken, the particular field to be accumulated must be identified. For this, the low-order position of the field within the record must be specified in the DTF entries (TOTALS), and the field must be defined by a high-order word mark in the storage area referred to automatically by the IOCS routines. That is, the word mark must be in the work area whenever the work area is specified by the DTF entries (WORKAREA). When a work area is not DTF-specified, the word mark for this field must be in the input/output area. The hash-total field can have a maximum of ten characters.

The record count and/or hash total may also be specified for tapes with nonstandard labels. In this case they are accumulated automatically, but the user must program to check (input) or write (output) them.

Once the programmer has determined the types of records to be handled and has planned the input, output, and work areas for his job, he writes only one instruction each time the program calls for a record to be read, written, or punched. For these operations, IOCS makes two macro instructions (GET and PUT) available to the programmer. Four others (SEEK, SCAN, RELSE, and SPACE/SKIP) are provided for special conditions.

### *GET Macro*

This instruction locates the next single record for processing, and it can be written in one of two basic forms (Figure 8). In both forms, the term FILEA represents the symbolic name of the file assigned in the DTF entry. The term WORKA, in the second form, represents a work area used for the file and labelled in the DA statement. The form of the GET macro to use, and the specific functions performed, are determined by the types of records being handled and by the processing plans specified in the file descriptive entries (DIOCS and DTF).

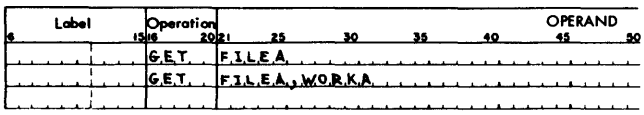



Figure 8. GET Macro Instruction

#### Tape Records and Disk Records with Consecutive Processing

FIXED-LENGTH BLOCKED RECORDS
PROCESSING IN THE INPUT AREA

Whenever records are to be processed in the input area, the first form of the GET macro (GET FILEA) is used.

Blocked fixed-length records can be processed in the input area, only if indexing is specified in the DTF entries (INDEXREG). In this operation (Figure 9A), the first GET instruction causes a block of records to be read from tape or disk to the input area, and it initializes the index register so that reference can be made to the first record in the block. Subsequent GET instructions increment the index register, and successive records can be processed. After all records in the block have been processed, the next GET again reads a block of records and initializes the index register.

When blocked fixed-length records are processed in the *non-overlap mode*, the particular index register specified in the DTF entry should be included in the input area DA statement. Also, the individual fields should be labelled in the DA statement, for reference throughout the program (Figure 10).

When tape records are processed with indexing in the *overlap mode*, however, two basic changes must be made in the DA statement for the input area (Figure 11, relate to Figure 10):

1. Field labelling *must* be omitted.
2. The index register *must* be omitted.

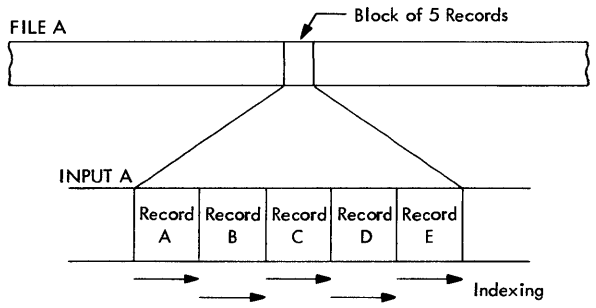
To assign labels to fields set up in the DA statement, the user must equate these labels to one-less-than the actual positions of the fields within the record. This is necessary because, with *overlap*, the index register always contains the *address* of the high-order position of the record to be processed. Indexing must be indicated either in the equate statements, as shown, or in the individual instructions throughout the program.

FIXED-LENGTH BLOCKED RECORDS
PROCESSING IN WORK AREAS

The first form of the GET macro (GET FILEA) is used whenever all records in an input file are to be processed in the same work area. For this, the label of the work area must be included in the DTF entries, and indexing must be omitted.

The second form of the GET macro (GET FILEA, WORKA) is used whenever records are to be processed in different work areas. It specifies the work area required for each separate record. It may be advantageous to set up two work areas, for example, and to specify each area in alternate GET instructions. This would permit the programmer to compare each record with the preceding one, for a control change. Whenever work areas are to be specified in GET instructions, both indexing and a work area specification must be omitted from the DTF entries.

When work areas are used (see Figure 9B, C), the first GET instruction in the program transfers a block of records to the input area, and then moves the first record in the block directly to the specified work area. Each subsequent GET instruction moves the next individual record from the input area to the work area, until all records in the block have been processed. Then a new block is automatically read in, and the operation is repeated.

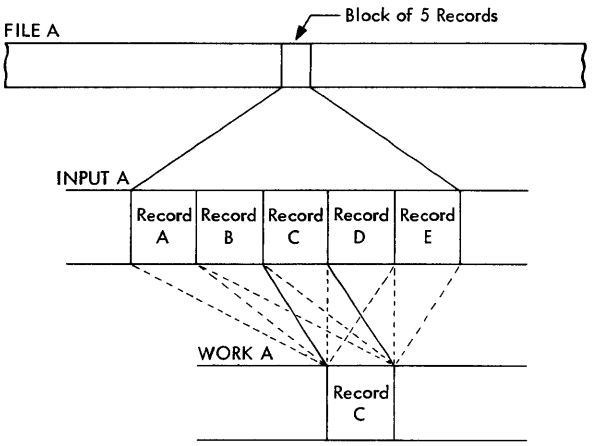



GET FILEA

( DTF INDEXREG Specified
DTF WORKAREA Not Specified )

Used for fixed-length records when processing can be performed in the input area.

Ⓐ

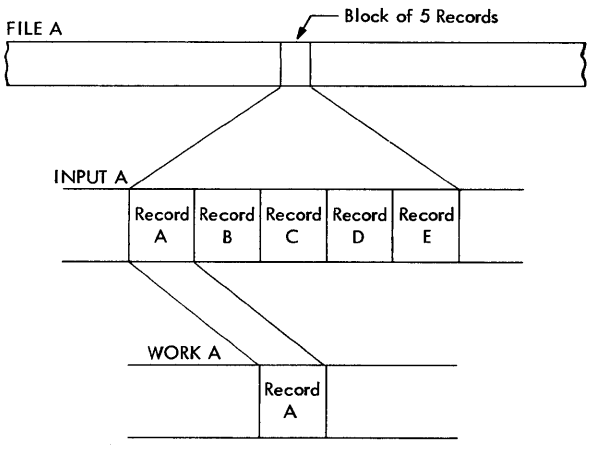



GET FILEA

( DTF INDEXREG Not Specified
DTF WORKAREA Specified )

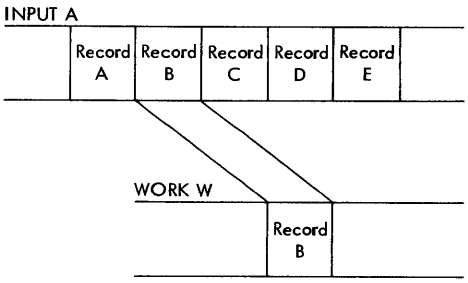
Used for fixed-length or variable-length records when one record work area is required.

Ⓑ




GET FILEA,WORKA

Next GET Instruction




GET FILEA,WORKW

( DTF INDEXREG Not Specified
DTF WORKAREA Not Specified )

Used for fixed-length or variable-length records when records are moved to different record work areas.

Ⓒ

Figure 9. Reading Blocked Tape Records

| Label | Operation | OPERAND |
|---|---|---|
| INPUTA | DA | 5,X100,,X2,,G |
| DATE | | 32,,37 |
| NAME | | 11,,30 |
| MANNO | | 4,,8 |
| YRGROS | | 70,,76 |
| YRWTAX | | 85,,89 |
| YRFICA | | 90,,94 |
| | | 1,,1 |

Figure 10. DA Statement for Indexed Tape Records, Non-Overlap Mode

Work areas specified for input files may also be specified for corresponding output files.

For *tape records,* the principle of specifying the work area in the GET instruction is also used when an *output area* is to be treated as a *work area.* When the output records are blocked in this operation, indexing must be specified in the DTF entry for the output area. If records are processed in the *non-overlap mode,* the index register is included in the DA statement for the output area, and the GET macro is written with the label of the output area in the operand field (Figure 12). If records are processed in the *overlap mode,* however, the index register cannot be specified

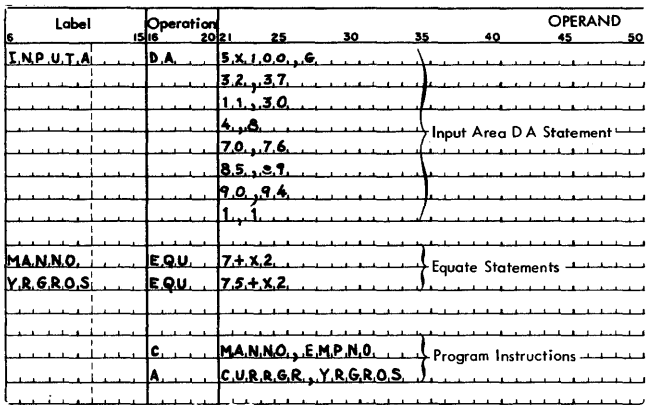



Figure 11. Indexed Records, Overlap Mode

in the output area DA statement, and must be included in each GET instruction. Instead of entering the label of the output area in the operand field, the output area is indicated by "0 + Xn" (zero + the number of the index register; Figure 13). This is necessary because, with overlap, the index register always contains the address of the high-order position of the record to be processed. The DA statement and fields for this output area must be set up in the same manner as described for the DA statement when records are processed in the input area with indexing and overlap (see Figure 11).

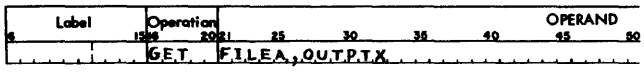



Figure 12. Output Area Used as Work Area, Non-Overlap Mode

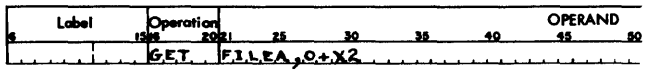



Figure 13. Output Area Used as Work Area, Overlap Mode

VARIABLE-LENGTH BLOCKED RECORDS

One or more work areas must be used to process these records. They cannot be processed in the input or output area, and indexing must not be specified in the DTF entry for this type of record. The GET instructions used and the operations performed to transfer variable-length records from tape to a core-storage work area are the same as described for blocked fixed-length records, under *Processing in Work Areas.*

UNBLOCKED RECORDS

When unblocked records are handled, each GET instruction transfers a single record to the input area. If a work area is specified in either the DTF entry or the GET instruction (see *Processing in Work Areas*), each GET then moves the record directly from the input area to that work area for processing.

If unblocked *tape records* are to be processed in *two* input areas *(overlap mode),* indexing is required. The programming for this operation is the same as that described for blocked fixed-length records processed in the input area with indexing and overlap.

## Disk Records with Control-Sequential Processing

FIXED-LENGTH BLOCKED RECORDS

The first form of the GET macro (GET FILEA) is used when disk records are processed in control-sequential order. The records may be processed in the input area if an index register is specified in the DTF entry (INDEXREG) and in the DA statement for the input area. Or, they may be processed in one work area specified in the DTF entry (WORKAREA).

With this type of processing, IOCS processes blocked records in consecutive order (see *Disk Records with Consecutive Processing*) whenever the linkage field in a record is blank. However, when the linkage field contains an address for a sequential record filed elsewhere, IOCS reads that record and makes it available to the programmer for processing. After the out-of-order record(s) has been processed, IOCS returns to the consecutive block of records and makes the next record available to the programmer.

FIXED-LENGTH UNBLOCKED RECORDS

When unblocked records are handled, each GET transfers a single record to the input area and, if a work area is specified, moves the record to that area for processing. If the linkage field in the record is blank, IOCS transfers the next consecutive record on the next GET. If the field contains an address, IOCS transfers the record at that address on the next GET.

## Disk Records with Random Processing

FIXED-LENGTH BLOCKED RECORDS

The first form of the GET macro (GET FILEA) is used when disk records are processed in random order. Each GET transfers records from disk to the input area in core storage by initiating both seek (if necessary) and read operations. However, before the programmer issues a GET instruction, he must determine which specific disk record is to be transferred. He computes the disk address of the record by the formula used in his file organization system, and moves this address to the IOCS area labelled IOCADR (IOCS Address). Then the next GET (Figure 14) transfers the block containing the specified record from disk to core storage.

The disk address computed for use by IOCS must be a seven-digit number (SSSSSSR). The first six digits

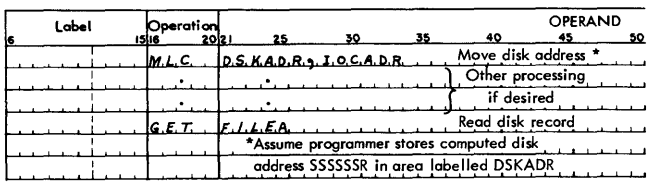



Figure 14. Instructions to Read Disk Records, Random Processing

(SSSSSS) are the actual address of the disk sector where the block of records is stored. If the block of records covers more than one sector, this is the address of the first section. The seventh digit in this address (represented by R) tells IOCS which record in the block is to be made available for processing. Because a block can contain a maximum of ten records, this may be any digit 0-9. The digit "0" represents the first record in the block, "1" represents the second record, etc.

The specified record is made available for processing in the input area, if indexing is specified in the DTF entry (INDEXREG) and in the DA statement for the input area (Figure 15). If a work area is specified in the DTF entry (WORKAREA), IOCS moves the specified record from the input area to the work area for processing.

FIXED-LENGTH UNBLOCKED RECORDS

These records are handled in the same manner as blocked records except that a single record, instead of a block, is transferred from disk to core storage by each GET instruction. The computed disk address moved to IOCADR always contains "0" in the units position. The record is available for processing in the input area, or in the work area specified in the DTF entry.

### Card Records

IOCS considers card records as fixed-length unblocked. Each GET instruction transfers a single record to the input area. If a work area is specified, each GET then moves the record directly to that work area for processing. The first form of the GET instruction (GET FILEA) is used when records are to be processed in the input area or in one work area specified in the DTF entry (WORKAREA). If records are to be processed in different work areas, the second form of the GET instruction (GET FILEA,WORKA) is used.

A work area is required for card input files whenever:

- the Read Release special feature is used
- the Processing Overlap special feature is used
- card-input operations are overlapped disk-seek operations.

For IOCS to use any of these features and move the card record to the specified work area, the Advanced Programming special feature is also required. However, the Read Release special feature can be used without advanced programming if the *user* provides programming to move the card record to the work area immediately after the GET instruction for that record.

STACKER SELECTION

Card input files should be selected to stack in pocket 1 or 2, because the IOCS card-read error routine stacks any unreadable cards in the normal pocket. This selection (with cards processed in the non-overlap mode) may be specified in the DTF entries (CARDPOC) when all cards are to be stacked in the same pocket, or it may be specified in the GET macro, but not in both. With the Read Release special feature, stacker selection (if any) must be specified in the DTF entries.

To specify stacker selection in the GET instruction, the basic form of the macro instruction is modified to indicate the pocket number (Figure 16). In the first form of the GET macro, two commas separate "FILEA" and "2". In the second form, one comma separates "WKAREA" and "1" in the operand field. Because commas are always used to separate various operands, either form enters the number as the *third* operand so that it can be recognized as a pocket number by the IOCS routines.

Stacker selection cannot be specified when card input records are processed in the *overlap mode*, or when card-read operations are overlapped with disk-seek operations. In either case, all cards are stacked in the normal read pocket.

## PUT Macro

This instruction writes, prints, or punches a record that has been processed. It operates much the same as the GET macro, but in reverse. Similar to GET it is written in one of two basic forms (Figure 17). In both forms, the term FILEX represents the symbolic name of the file assigned in the DTF entry. The term WORKX, in the second form, represents a work area used for the file. The form of the PUT macro to use, and the exact functions performed, are determined by the processing plans and file specifications (DIOCS and DTF entries).

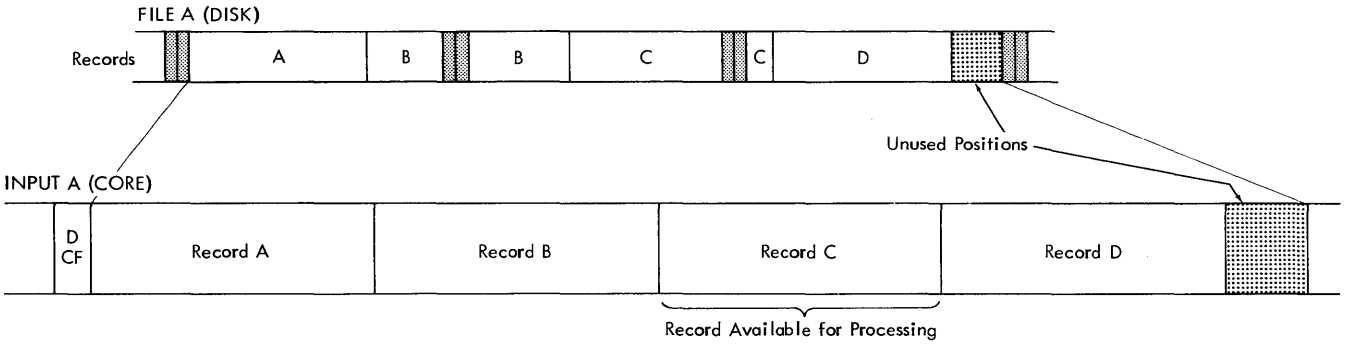
### Tape Records and Disk Records with Consecutive Processing

FIXED-LENGTH BLOCKED RECORDS

BUILDING IN THE OUTPUT AREA

Whenever records are built directly in the output area, the first form of the PUT macro (PUT ,FILEX) is used.

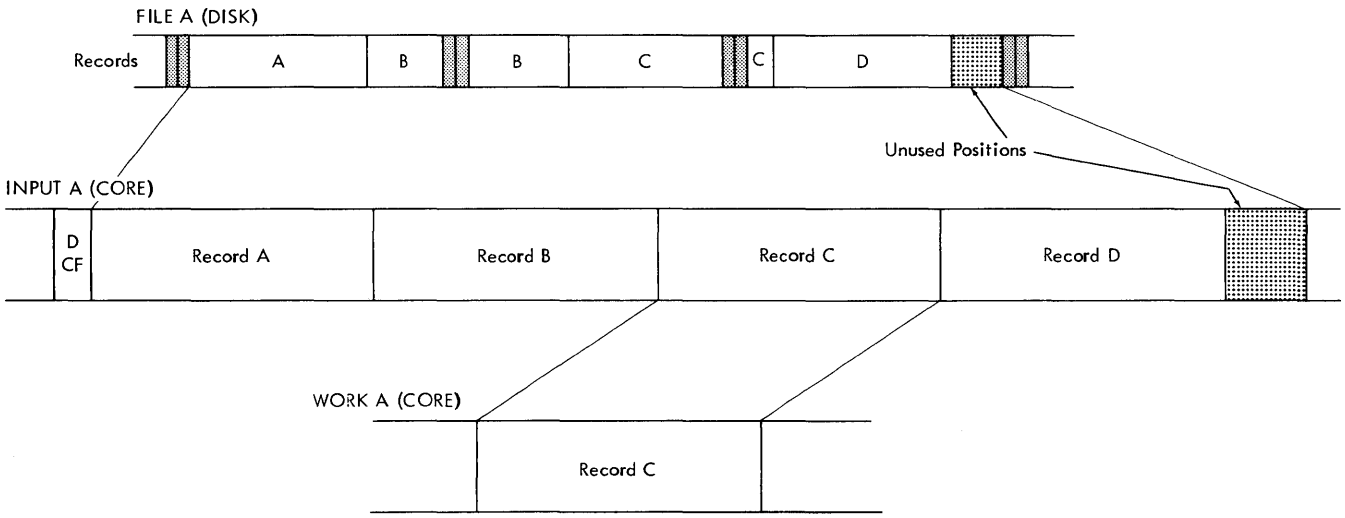
Blocked fixed-length records can be built in the output area, only if indexing is specified in the DTF entries (INDEXREG). In this operation (Figure 18A), each PUT instruction increments the index register so




GET FILEA

Blocked Records
Computed Disk Address - SSSSSS2
DTF INDEXREG Specified
DTF WORKAREA Not Specified

(A) PROCESSING IN THE INPUT AREA




GET FILEA

Blocked Records
Computed Disk Address - SSSSSS2
DTF INDEXREG Not Specified
DTF WORKAREA Specified

(B) PROCESSING IN A WORK AREA

DCF - Disk Control Field

Figure 15. Reading Blocked Disk Records, Random Processing

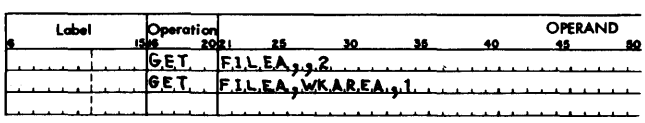



Figure 16. GET Instruction with Read Stacker Selection

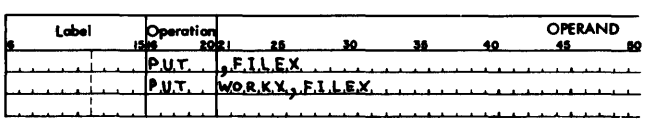
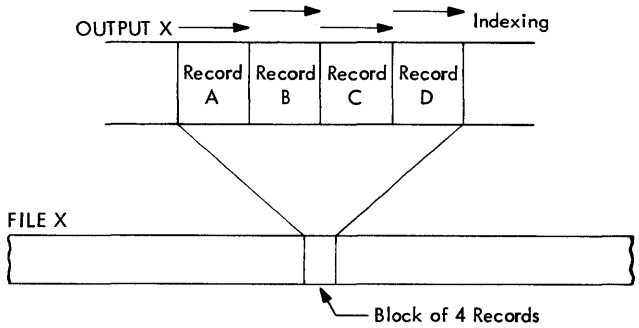



Figure 17. PUT Macro Instruction

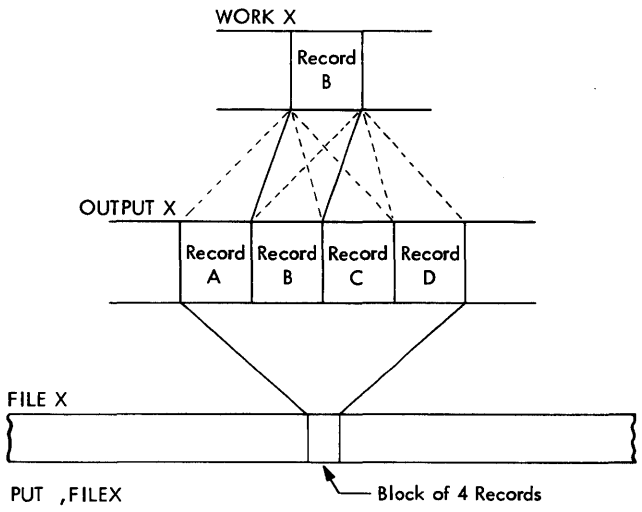



PUT ,FILEX

( DTF INDEXREG Specified
DTF WORKAREA Not Specified )

Used when fixed-length records can be built in the output area.

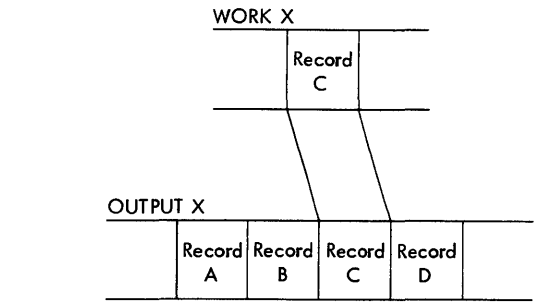
Ⓐ




( DTF INDEXREG Not Specified
DTF WORKAREA Specified )

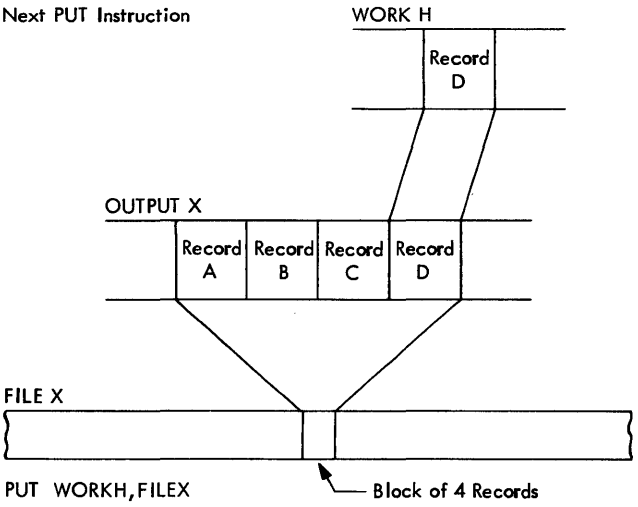
Used for fixed-length or variable-length records when one record work area is required.

Ⓑ




PUT WORKX,FILEX

Next PUT Instruction




( DTF INDEXREG Not Specified
DTF WORKAREA Not Specified )

Used for fixed-length or variable-length records when records are built in different work areas.

Ⓒ

Figure 18. Writing Blocked Tape Records

that the next record can be built in the next record-area within the output block. Also, each PUT tests to see if the output area is filled. When it is, the block of records is transferred to the output file, and the index register is initialized so that the following record will be built at the beginning of a new block.

When disk records are processed or when tape records are processed in the *non-overlap mode,* the particular index register specified in the DTF entry must be included in the output area DA statement. Also, the individual fields should be labelled in the DA statement for reference throughout the program. This is similar to the indexed *input area* DA statement (see Figure 10). However, when tape records are processed in the *overlap mode,* the index register and field labelling must be omitted from the DA statement. The DA statement and fields must be set up in the same manner as described for the input area when blocked fixed-length tape records are processed with indexing and overlap (see Figure 11).

FIXED-LENGTH BLOCKED RECORDS

BUILDING IN WORK AREAS

The first form of the PUT macro (PUT ,FILEX) is used whenever all records in an output file are to be built in the same work area. For this, the label of the work area must be included in the DTF entries and indexing must be omitted.

The second form of the PUT macro (PUT WORKX, FILEX) is used whenever records are to be built in different work areas. It specifies the work area required for each separate record. Both indexing and a work area specification must be omitted from the DTF entries, with this type of PUT instruction.

When work areas are used (Figure 18B, C), each PUT instruction moves an individual record from the specified work area to the proper location in the output area. When a block of records is completed, it is transferred to the output file.

Work areas specified for output files may be the same work areas as specified for corresponding input files.

For *tape records,* the principle of specifying the work area in the PUT instruction is also used when an *input area* is to be treated as a *work area.* That is, each record is processed (built for output) in the input area and then moved to the output area to be written on the output tape. When the input records are blocked in this operation, indexing must be specified in the DTF entry for the input area. If records are processed in the *non-overlap mode,* the index register must be included in the DA statement for the input area. Also, the PUT macro is written with the label of the input area in the operand field (Figure 19). If records are processed in the *overlap mode,* however, the index register cannot be specfied in the input area DA statement, and it

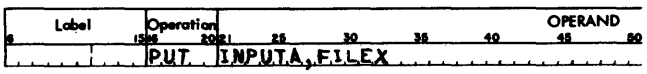



Figure 19. Input Area Used as Work Area, Non-Overlap Mode

must be included in each PUT instruction. Instead of entering the label of the input area in the operand field, the input area is indicated by "0 + Xn" (zero + the number of the index register; Figure 20). This is necessary because, with *overlap,* the index register

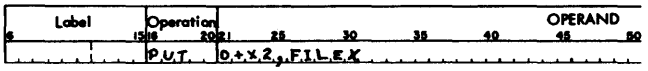



Figure 20. Input Area Used as Work Area, Overlap Mode

always contains the address of the high-order position of the record to be processed. The DA statement and fields for this input area must be set up in the same manner as described for processing records in the input area with indexing and overlap (see Figure 11).

VARIABLE-LENGTH BLOCKED RECORDS

One or more work areas may be used to build these records. The PUT instructions used to transfer variable-length records are the same as those described for blocked fixed-length records, under *Building in Work Areas.* In addition, the length of each record must be determined by the user's program and included in the record-length field. This length is then used by IOCS in a test to see if the record fits in the output block.

UNBLOCKED RECORDS

When unblocked records are handled, each PUT instruction transfers a single record from the output area to the output file. If a work area is specified in either the DTF entry or the PUT instruction (see *Building in Work Areas*), each PUT first moves the record from that work area to the output area, and then transfers it to the output file.

If unblocked tape records are to be built in *two* output areas (*overlap mode*), indexing is required. The programming for this operation is the same as that described for blocked fixed-length records processed with indexing and overlap.

## Disk Records with Control-Sequential Processing

FIXED-LENGTH BLOCKED RECORDS

The PUT instruction is used for disk records processed in control-sequential order, only if an updating operation is performed. The first form of the PUT macro (PUT ,FILEX) is always used. The file must be specified as an *input* file to be *updated,* in the DTF entries (FILETYPE and UPDATE).

Each record in a control-sequential file is made available for processing in the manner described under *GET Macro* for this type of file. When the processing for a record is complete, a PUT instruction is issued. PUT affects the record made available by the last GET issued. If the record was processed in a work area, PUT returns the record to its proper place in the *input* area. When all records in a block have been processed, a PUT instruction transfers the block from the *input* area to its original location in the *input* file. Or, when PUT is issued for an out-of-order record, that record is returned to its original location.

FIXED-LENGTH UNBLOCKED RECORDS

If a file contains unblocked records, these records can be updated in the manner described for blocked records. A single record, instead of a block, is transferred to and from core storage.

### Disk Records with Random Processing

FIXED-LENGTH BLOCKED RECORDS

The first form of the PUT macro instruction (PUT ,FILEX) is used when the disk records are processed in random order. Each PUT transfers records from core storage to disk by initiating both seek (if necessary) and write operations.

If a file is identified in the DTF entries (FILETYPE and UPDATE) as an *input* file to be *updated*, records can be read into core storage, updated, and rewritten on the same file in their original location. The block of records containing a specified random record is transferred to storage by providing IOCS with a disk address and issuing a GET instruction (see *GET Macro, Disk Records with Random Processing*). After the individual record has been updated, the block is transferred back to disk by issuing a PUT instruction. The block is transferred to the disk location specified by the disk address that was supplied to IOCS ahead of the last GET instruction for this file. Therefore, a disk address need not be supplied to IOCS ahead of the PUT instruction. When a work area is assigned to a file for this operation, PUT moves the updated record from the work area back to its original place in the *input* area and then writes the block of records on disk.

FIXED-LENGTH UNBLOCKED RECORDS

If a file specified as *input* contains unblocked records, these records can be updated in the manner described for blocked records. In this case a single record, instead of a block, is transferred to and from core storage.

Unblocked records can also be written in random order on a file identified as *output* in the DTF entry (FILETYPE). This operation makes it possible to set up a new logical file on disk using a file organization system that suits the particular requirements of the data to be stored. When a file is specified as random output, IOCS handles *only* fixed-length unblocked records and it does not recognize a DTF-specified work area. Records to be written on disk must be built in the output area before the PUT instruction is issued to write the output data. Therefore, if the programmer wants blocked records for later use, he must block them himself.

For an *output* file, the first form of the PUT instruction (PUT ,FILEX) is used and a disk address must be supplied to IOCS ahead of the PUT. Similar to a random input operation, a formula for computing disk addresses must develop a seven-digit address (SSSSSSR). This consists of a six-digit actual sector address and a seventh digit that is always zero, since IOCS treats the records as unblocked.

### Card and Printer Records

When card or printer records are handled, each PUT instruction transfers a single record from the output area to the output file. If a work area is specified, each PUT first moves the record from that work area to the output area and then transfers it to the output file. The first form of the PUT instruction (PUT ,FILEX) is used when records are built in the output area or in one work area specified in the DTF entry (WORKAREA). If records are built in different work areas, the second form of the PUT instruction (PUT WORKX,FILEX) is used.

A work area is required for card output records whenever the Advanced Programming special feature is specified. Without advanced programming, a work area is not required and records may be built in the punch output area. In this case, however, only the high-order position of the output area (101) may contain a word mark. Positions 102-180 must not contain word marks.

A work area is required for printer records whenever printer output is overlapped with disk-seek operations.

STACKER SELECTION

Card output files may be selected to stack in pocket 4 or 8. This selection may be specified in the DTF entries (CARDPOC) when all cards are to be stacked in the same pocket, or it may be specified in the PUT macro, but not in both. It is specified in PUT by modifying the basic form of the macro and writing the pocket number as the third operand (Figure 21).

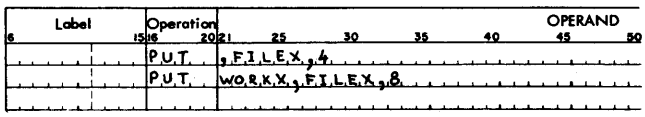



Figure 21. PUT Instruction with Punch Stacker Selection

Unlike card input files, this selection applies to card output records processed in either the *non-overlap* or *overlap mode.*

PRINTER FORMS CONTROL

Spacing and skipping of forms can be controlled by the IOCS routines. The operation may be specified in the DTF entries or in the PUT instruction, but not in both. In either case, the standard IBM 1401/1460 d-character for forms control is used to indicate the desired operation (see the SRL publication *IBM 1401/1460 System Operation Reference Manual*, A24-3067). The d-character is specified in the DTF entry (FORMCNTL) if the same operation is to be performed for each printed line, such as double-spacing. It is specified in the PUT instruction (Figure 22) whenever different spacing or skipping is to occur for different printed lines.

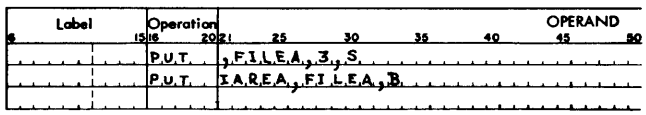



Figure 22. PUT Instruction with Printer Forms Control

The layout of a form may require certain forms control either before *or* after a particular line of printing, or both before *and* after printing. When one control (before *or* after) is required, the d-character is entered as the third operand in the PUT instruction. For control both before *and* after, the d-character for *immediate* spacing or skipping (before printing) must be entered as the third operand, and the d-character for *after print* spacing or skipping as the fourth operand.

## SEEK Macro — Disk Records

When a disk read or write operation is to be performed, the time required for seeking a record can also be used for processing by inserting a SEEK macro in the program. That is, the programmer issues the SEEK instruction first, when he is ready for a new record. He performs other processing, while the access moves to the proper cylinder, and then he issues the GET, or PUT, for the same record to actually read, or write, the record. If records are processed in random order, the seven-digit computed disk address must be supplied to IOCS ahead of the SEEK instruction.

The name of the file specified in the DTF entry must be entered in the operand field of this macro.

## SCAN Macro — Disk Records

When the Scan Disk special feature is installed and SCAN is specified in the DIOCS entry (FEATURES), this macro instruction may be included in the user's program if records are processed in random order and handled in the sector mode of operation. SCAN causes a rapid search of records by sector to locate a particular record containing specified data.

The SCAN macro (Figure 23) requires four operands, entered in this order:

1. File name. The DTF-specified name of the file to be searched.
2. Label of a storage area set up to contain the specified data for which the scan is performed. The programmer must provide a storage area that consists

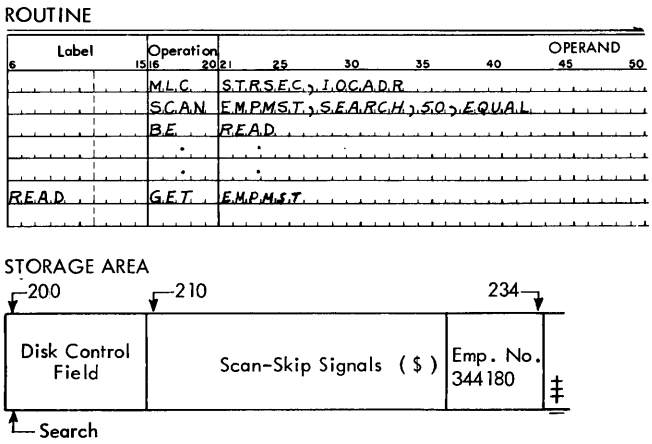



Figure 23. SCAN Macro Instruction, Disk Records

of a ten-position Disk Control Field and the field(s) that is to be searched. This data area must be arranged in a format comparable to the format of the record on disk. For example, if a six-position employee-number field to be searched is located in positions 20-25 in each record, it must be located in those storage positions following the Disk Control Field. If two or more fields contain specified data, they are treated as one large field, with the high-order position of the left-hand field becoming the high-order position of the combined field. Any positions to the left of the first data field, and between data fields, must contain scan-skip signals ($). The reserved storage area needs to be only large enough to include the last field to be searched in the record. Positions to the right may be omitted. The last position in any sector cannot be included in the scan operation. The storage area must be followed by a group-mark with word-mark.

3. Number of sectors to be searched. This may be a maximum of 999 sectors and may be entered as a one-, two-, or three-character operand.
4. The basis on which the record is to be selected:

   Equal — the specified data is equal to the record data.

   High — the specified data is equal to or higher than the record data.

   Low — the specified data is equal to or lower than the record data.

Before the SCAN macro is issued, the sector address at which the search is to start must be moved into IOCADR (IOCS Address). This must be a seven-digit number with a *blank* in the units position. The specified data must also be moved into the proper positions in the reserved storage area.

Scanning of records stops when a record that meets the specified conditions is found, or when all sectors specified in the macro have been searched. The programmer must test the high-low-equal compare indicator to determine if a record was found. If it was, the sector address of that record is available in IOCADR (with "0" in the units position). The programmer can read the record into core storage by issuing a GET instruction, as shown in Figure 23, or he can move the address from IOCADR and store it elsewhere for later use in his program.

Figure 23 illustrates a sample routine and storage area for a scan operation based on these factors:

- Scan an employee master file (EMPMST) for employee number 344180, located in positions 20-25 of the record.
- Starting sector address stored by the programmer at STRSEC.
- Scan 50 sectors.
- Storage area labelled SEARCH.
- Transfer record from disk to core storage for processing.

After a record meeting the specified conditions has been handled, the programmer may, if he desires, plan a further search in either of two methods.

1. Search the remainder of the specified sectors for the next record that meets the specified conditions. For this, he branches back to the SCAN instruction, without moving a new address to IOCADR.
2. Restart a search of the full number of specified sectors. He must supply IOCADR with a new starting address and again issue the SCAN instruction.

### RELSE (Release) Macro — Tape Records

The release macro instruction is used in conjunction with blocked tape records. It allows the programmer to skip the remaining records in a block and continue processing with the first record of the next block. This function applies to a job in which records on tape are categorized and each category (perhaps a major grouping) is planned to start as the first record in a block. For example, sales data may be recorded and analyzed by division (major), district (intermediate), and branch office (minor). Then, if management frequently requires special analyses of sales for certain specified divisions, these analyses can be obtained quickly and efficiently with a system planned so that the records for each division start at the beginning of a new block of records. The specified division can be located readily by checking only the first record in each block. If that record is not in the specified division, the other records in the block can be ignored and the first record of the next block can be checked.

The symbolic name of the file, specified in the DTF entry, is entered in the operand field of the release instruction. When this is an input file, the next GET instruction reads in a new block of records and makes the first record available for processing. If indexing is used, the index register is initialized. Because input records are skipped in this operation, record counts and hash totals cannot be taken for this file. Therefore, IOCS-checking of these items in the trailer label should *not* be specified.

When an output file is released, the next PUT instruction causes the existing block of records to be written on tape. The new record becomes the first record of a new block. With blocked fixed-length records, any unfilled portion of the block is padded with the character specified in the DTF entry (PADDING), or with blanks. The padded records are included in any record counts or hash totals. With variable-length records and an unfilled portion, a short block is written.

When output records are built in work areas, the control number in each record can be examined (by the user's program) as the record is being built, to determine whether or not the record is the first one of a new major group. If it is, RELSE is issued and the PUT instruction for that record causes the existing block to be written *before* the record is moved from the work area to the output area. If records are built directly in the output area, however, examining the control number of a record as it is built would cause the first record of a new group to be erroneously included in the previous block when RELSE and PUT for that record are issued. Therefore, programming must be provided to make sure that the first record of a new group *is* in a new block.

### SPACE/SKIP Macro — Printer Records

When records are processed in the *overlap mode*, spacing and skipping of printer forms may be controlled by the SPACE/SKIP macro instruction, instead of the DTF entry (FORMCNTL) or the PUT instruction. The 1401/1460 d-character for the desired operation is entered in the operand field of the SPACE or SKIP instruction.

Data processing installations using one or more IBM 1311 Disk Storage Drives may have many disk packs to be stored and handled, and each pack may contain several logical files of data. To ensure that the correct logical file is used in each job, it is common practice to identify the files of data with *header* labels recorded on the disk pack, in addition to a printed label on the outside of the pack. A header label contains such factors as identification numbers, identification name, and effective dates. Files in an IBM 1301 do not have header labels.

Files on either 1301 or 1311 disk may contain trailer labels to indicate the end of the logical file written on the pack, when records are processed in consecutive or control-sequential order.

IOCS can handle 1311 disk packs with standard header and trailer labels, or without labels. When labels are used, IOCS can check the header label of an input file before processing any records, to make sure that the correct data is available. For an output file, IOCS can check all previously written header labels to determine that the portion of the pack specified (DTF FILESTART and FILEND entries) for writing records does not contain active data. If the area is available, IOCS can write a header label to identify the new output records that will be recorded.

When records in a 1301 or a 1311 are processed in consecutive or control-sequential order, IOCS automatically senses trailer labels in an input file and writes trailer labels in an output file.

### *Header Label (1311 Files Only)*

All header labels are written on the last track of a disk pack, and one 100-character sector is used for each label. A pack can contain a maximum of 19 different labeled files. The disk addresses of these labels is always 000180-000198, regardless of the address range of the remainder of the pack.

When a new disk pack is received in an installation, the header label track must be set up before the pack is used in an IOCS job. The special label addresses must be written, each sector must be identified as a header label (see item 1, *Label Identifier*), and a pack number must be assigned (see item 6, *Pack Serial Number*). A disk label program is available as part of the Disk Utility Programs for the IBM 1401/1311.

IOCS can handle four different arrangements of labeled files in the same operation:

a. One file on one pack.
b. Two or more files on one pack (multi-file pack).
c. One file on two or more packs (multi-pack file).
d. A file split into two or more sections on different packs.

These types of files are illustrated schematically by the header labels in Figure 24.

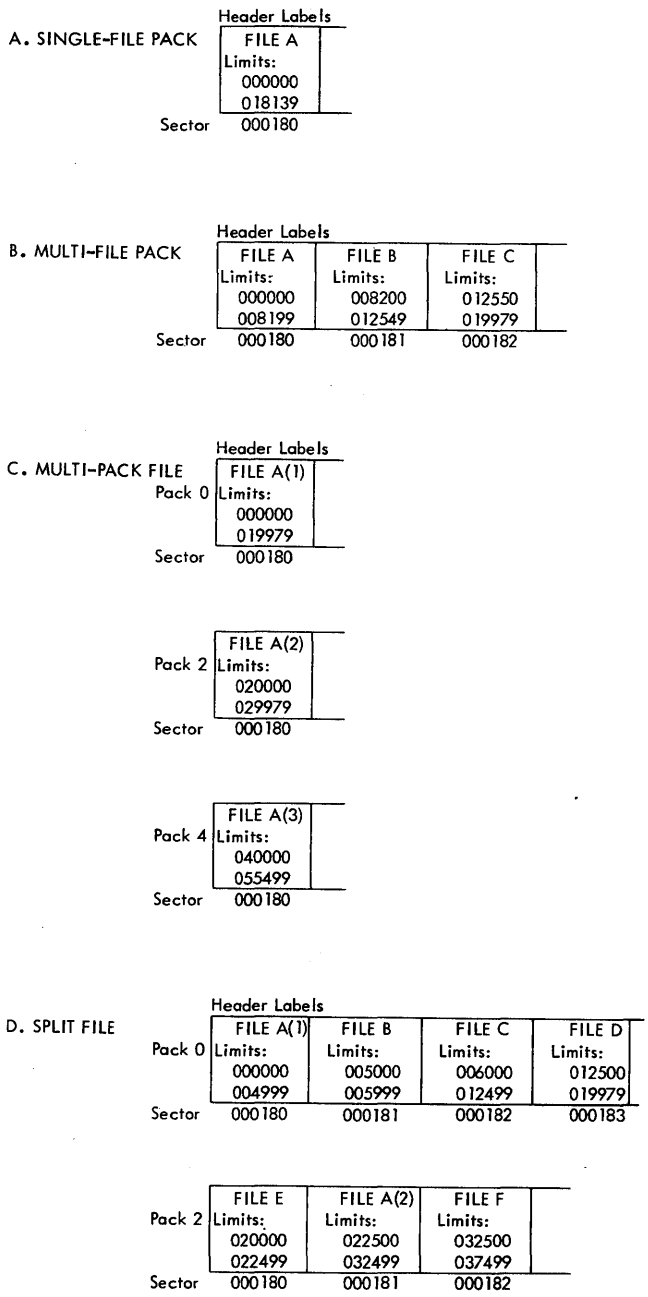



Figure 24. Schematic of Header-Label Arrangement for Files in the IBM 1311

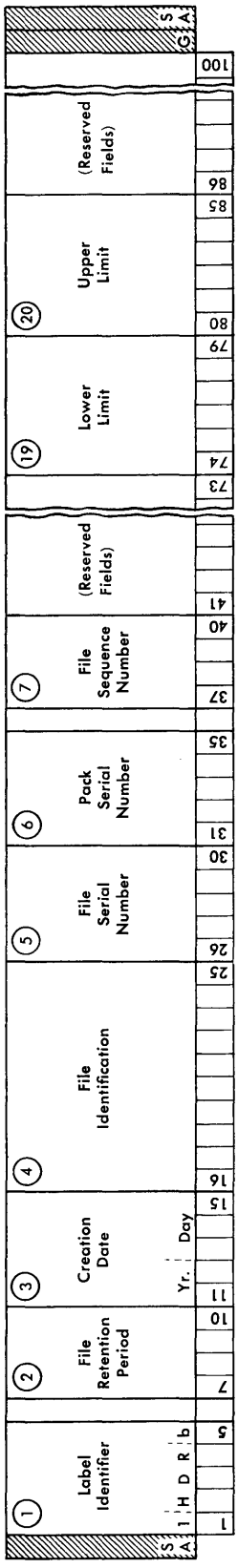



Figure 25. Schematic of Standard Disk Label Fields Processed by IOCS

In the 100-character standard header label, nine fields contain standard identifying information that can be processed by IOCS (Figure 25). Two of the fields (*Label Identifier* and *Pack Serial Number*) identify a sector as a header label. The other fields identify the logical file and are automatically written with (or checked against) information specified in the DTF entries or RDLIN cards (see *RDLIN Macro*). If a logical file requires more than one pack, the header information in the first five fields is repeated on all packs. The information in the last four fields varies (see items 6, 7, 19, 20). The remaining positions in the standard header label are reserved for other functions. They are not processed by IOCS.

When an *input* file with a header label is to be processed, the user must specify (in DTF CHECKLABEL and DIOCS DLABDEF) the checking he wants performed by the IOCS routines: complete (the nine fields handled by IOCS), partial (*File Identification* only), or none at all. If an error is detected in this check, a programmed halt occurs. After the halt, label checking may be retried on the same pack or on a new pack, whichever is desired. If the header label is correct in every respect except file limits, the file may be processed or the labels can be rechecked.

When *output* files with header labels are to be written, the user must specify this in the DIOCS entry DLABDEF. IOCS checks all labels on a pack to determine that the portion of the pack specified (in DTF FILESTART and FILEND entries) for the new output file does not contain active data. For this, a name in the *file identification* field of the old header designates a label in use, the file *limits* in the header label indicate the portion of the pack that contains data, and the *dates* in the header label indicate active or inactive data. When old records are found to be inactive, IOCS deletes their expired header label(s) by *blanking* the *file identification* field(s), and then the output operation proceeds as specified. If they are found to be active, however, a halt occurs. The user can mount a different pack, or he can use this same pack anyway. If he elects to use the same pack, IOCS automatically deletes the old label.

The nine identifying fields in the standard header label are:

1. *Label Identifier* – consists of the digit 1, the letters HDR, and a blank position.

   When a new disk pack is received in an installation and before it is used in an IOCS job, the characters 1HDRb should be written in this field as a header-label identification. This should be written in each of the nineteen header labels on the pack.

   Input File: This field identifies the sector as a header label.

   Output File: This field is retained.

2. *File Retention Period* — 4 digits.

This specifies the number of days after the creation date that the logical file is to be kept active.

Input File: When the header label is to be completely checked, the number of days originally written in this field must be specified in the DTF entry HEADER.

Output File: The number of days to be recorded must be specified in DTF HEADER, and will be written by IOCS.

3. *Creation Date* — 5 digits.

The dating system consists of 2 digits for the year, followed by 3 digits for the day of the year.

Input File: When the header label is to be completely checked, this date must be specified in the DTF entry HEADER.

Output File: The date is taken from today's date in storage. The user must load the date in storage positions 82-86. To do this, he punches a card with today's date and loading instructions, and inserts it in the object-program condensed deck:

| *Columns* | *Punch* | *Description* |
|---|---|---|
| 1-3 | 082 | Storage Location |
| 4-5 | 05 | Number of Characters |
| 6 | 0-5-8 | Word Separator |
| 7-11 | xx¦xxx (Yr¦Day) | Today's Date |

4. *File Identification* — 10 characters.

This may be a job name in an installation. It may consist of any combination of alphamerical characters and blanks, but it *must* contain at least one significant character (not a blank).

IOCS always checks this field to determine if the header label is in use. When this field contains any significant character(s), the label is considered active. When it contains all blanks, the entire label as assumed to have been deleted.

Input File: When the header label is to be checked (either completely or partially) this name must be specified in the DTF entry HEADER.

Output File: The name to be recorded must be specified in the DTF entry HEADER, and will be written by IOCS.

5. *File Serial Number* — 5 digits.

This generally represents a job number in an installation.

Input File: When the header label is to be completely checked, this number must be specified in the DTF entry SERIALNUM.

Output File: The number to be recorded must be specified in the DTF entry SERIALNUM, unless the user plans to make this number the same as the *pack* serial number. If the pack serial number is used and the job requires two or more packs, the number of the *first* pack becomes the file serial number on all packs.

6. *Pack Serial Number* — 5 digits

This is generally the sequential number of the pack within the whole installation. As soon as a new disk pack is received in the installation, it should be identified in this field with the next available number. The same number should be written in each of the nineteen header labels on the pack.

When a file area that consists of two or more packs is processed, the pack serial number of only the first pack is checked.

Input File: When the header label is to be completely checked, this number must be specified in the DTF entry PACKSERIAL.

Output File: This number must be specified in the DTF entry PACKSERIAL. IOCS verifies that the correct pack is mounted. If it is not, a halt occurs. After the halt, operation can be started using the same pack, if desired, or the pack can be replaced. In this case, the new pack is checked before data is processed.

7. *File Sequence Number* — 4 digits.

This field is used to number the packs in a logical file. The number of the first pack is 0001 unless the user specifies some other number in the DTF entry FILSEQ.

Input File: When the header is completely checked, IOCS checks file sequence automatically. For this, either 0001 or the specified number is used to check the first pack. Any additional packs are checked with the next sequential number.

Output File: IOCS writes either 0001 or the specified number on the first pack, and thereafter serially numbers any additional packs.

19. *Lower Limit* — 6 digits.

This is generally the sector address of the first record, or block of records, in a logical file or area of a logical file. However, if a file (or file area) requires two or more packs, this starting address is written on the first pack only. On succeeding packs, this field contains the first address of the *pack*.

Input File: When the header label is completely checked, this number is checked against the *starting* address specified for the file area. This limit field applies only to files of input records processed on consecutive or control-sequential order. File limits have no significance when input records are processed in random order.

Output File: The number specified in the DTF entry FILESTART is written in this field on the first pack of a logical file. If the file is processed

in consecutive order and consists of two or more file areas, IOCS writes the number specified by FILESTART on the first pack of the first file area. On the first pack of other file areas, IOCS writes the number specified by a RDLIN card. This is an end-of-records condition (see *EOR Trailer*). On succeeding packs in any file area, IOCS automatically writes the first address of the *pack*.

This field is used when output records are written in either random or consecutive order.

20. *Upper Limit* — 6 digits.

This is generally the sector address of the last record, or block of records, in a logical file or area of a logical file. However, if a file (or file area) requires two or more packs, this ending address is written on the last pack only. On preceding packs, this field contains the last address of the *pack*.

Input File: When the header label is completely checked, this number is checked against the *ending* address specified for the file or area of the file. This limit field applies only to files of input records processed in consecutive or control-sequential order. File limits have no significance when input records are processed in random order.

Output File: The number specified in the DTF entry FILEND is written in this field on the last pack of a logical file, when that file consists of only *one* file area. If the file is processed in consecutive order and consists of more than one file area, however, IOCS writes the number specified by FILEND on the last pack of the first file area. On the last pack of other file areas, IOCS writes the number specified by a RDLIN card. This is an end-of-records condition (see *EOR Trailer*). On preceding packs in any file area, IOCS automatically writes the last address of the *pack*.

This field is used when output records are written in either random or consecutive order.

## *Trailer Label (1301 and 1311 Files)*

When an area of disk storage is allotted for a logical file, it should be made large enough to accommodate future expansion of the file. Therefore, when records are processed in consecutive or control-sequential order, the upper limit frequently does not agree with the end of the records on the disk. A trailer label is used to indicate the actual end of the records written at any particular time. A trailer label is also used to indicate the end of a portion of a logical file processed in consecutive order, when the rest of the file is recorded on another disk module or pack. Trailer labels have no significance when records are processed in random order.

Trailer labels are written in a record area equivalent to the data records with which they are associated, and they are written in the same mode of operation as the data records. However, only the first five positions of the record are used. The trailer label consists of the character 1EOFb or 1EORb.

### EOF (End-of-File) Trailer

The letters EOF indicate the end of all records for a particular logical file processed in either consecutive or control-sequential order. This trailer label (1EOFb) is always written immediately following the last data record.

On an *input file*, IOCS branches to the programmer's end-of-file address (specified in the DTF entry EOFADDR) when it detects this trailer label, or when the ending address specified in the DTF entry FILEND (or RDLIN card) is reached, whichever occurs first. On an *output file*, IOCS automatically writes this trailer label when the CLOSE macro instruction is issued.

### EOR (End-of-Records) Trailer

The letters EOR indicate that this is the end of the records on this disk module or pack, but that more records are available for this same logical file. This trailer label (1EORb) is written as the last record of a pack, in a multi-pack logical file, or of a module in a multi-module file. If sections of a file are recorded on parts of different modules or packs (see Figure 24D), this trailer label is written as the last record of each section. The EOR label is used by IOCS, only when records are processed in consecutive order.

When an *input* file is processed and an EOR label is detected on a unit or pack that is *not* the end of a file area, IOCS automatically continues reading records from the following module or pack. In this case the address of the EOR record differs from the *ending* address specified by the DTF entry FILEND, or by a RDLIN card. However, when the EOR label is detected at the end of a file area, IOCS branches to the user's routine specified in the DTF entry EOFADDR. Under these conditions, the address of the EOR label is the same as the DTF-specified (or RDLIN-specified) *ending* address. In his routine, the user must test to determine that this *is* an *EOR condition*, not an end-of-file condition. Also in this routine, a halt must be programmed and a RDLIN macro instruction must be issued if the user plans to continue processing this logical file. When the halt occurs, the operator can

mount a new pack (if necessary), and he must insert a RDLIN card to redefine the FILESTART and FILEND specifications for the succeeding file area.

When an *output* file is processed, IOCS writes an EOR trailer record whenever it detects the end of a module or pack or the end of a file area. When an EOR trailer record is written at the end of a module or pack that does *not* correspond to the end of a file area, IOCS automatically continues writing on the following pack. When it is written at the end of a file area, IOCS branches to the user's routine. Like input, this is the routine specified by the DTF entry EOFADDR. In this routine the user must program a halt and issue a RDLIN macro instruction if he plans to continue processing this logical file. When the halt occurs, the operator can mount a new pack (if necessary), and he must insert a RDLIN card to redefine the DTF FILESTART and FILEND specifications for the following file area.

Whenever a 1311 is used and an input or output file contains header labels, an OPEN macro instruction must be issued after an EOR condition. This is required to check or write the labels for the next file area. The file must be reopened after the RDLIN instruction, but before the first GET instruction for the new file area. A CLOSE macro instruction must *not* be given on an EOR condition, however. The logical file should be closed only after all file areas have been processed.

## Tape Labels

Data processing installations using magnetic tape storage have many reels of tape to be stored and handled. To ensure that the correct tape reel of data is used in each job, it is common practice to identify the tape itself with *header* and *trailer* labels, in addition to a printed label on the outside of the reel. A header label is the first record on tape and contains such factors as identification numbers, identification name, and effective dates. If a job requires more than one tape reel of data, the same header information (with the exception of tape and reel numbers) is repeated on all reels. A trailer label is at the end of data on tape and provides control totals and an indication that the reel is, or is not, the last reel for the job.

The format of both header and trailer labels may be any of three IBM standard forms, or it may be a different arrangement (nonstandard) planned by the user. The IOCS routines can process tapes with either standard or non-standard labels, or without labels. However, only standard labels are automatically written on output tapes and automatically checked on input (or output) tapes. Non-standard labels are written or checked by the user's program, outside the IOCS routines. In each job the user must specify in the DIOCS entries (LABELDEF) and in the DTF entries (TYPELABEL) the type of labels to be processed.

IOCS processes three types of tape labels as standard:

1. The IBM standard tape label – 120 characters. This type of label (Figure 26) is referred to as *Type A* in this publication.
2. The 80-character label. This type of label (Figure 27) is referred to as *Type B* in this publication. This label was originally used for tapes processed in the IBM 1401, 1410, and 7070 systems.
3. The 84-character label. This type of label (Figure 28) is referred to as *Type C* in this publication. This label was originally used for tapes processed in the IBM 7090 system.

When an *input* tape with a standard header label is to be processed, the user must specify (in DTF CHECKLABEL) the checking he wants performed by the IOCS routines: complete (the standard identifying information), partial (*File Identification* only), or none at all. If an error is detected in this check, a programmed halt occurs. After the halt, operation can be started using the same tape, if desired, or the tape can be replaced. In this case, the header of the new tape is checked before data is processed.

Whenever complete or partial checking is specified for an input tape with a standard header label, the standard trailer-label fields are also checked. If an error is detected, a programmed halt occurs. Operation can be resumed, if desired, or the error can be displayed for the operator to investigate.

When an *output* tape is to be written, the effective dates in the *old* header should be checked to ensure that the data on the tape is no longer active and may be destroyed. Therefore, IOCS automatically checks the retention period in the old header against the time elapsed between the creation date and today's date. If an error is detected, a programmed halt occurs. Then, the tape may be changed, or operation may be started using the same tape.

For IOCS to check the retention cycle in the old header label of an output tape and write today's date in the new header label, the date must be stored in positions 82-86. To load the date, the operator punches a DATE card with today's date and loading instructions, and inserts it in the object-program condensed deck:

| *Columns* | *Punch* | *Description* |
|---|---|---|
| 1-3 | 082 | Storage Location |
| 4-5 | 05 | Number of Characters |
| 6 | 0-5-8 | Word Separator |
| 7-11 | xx;xxx (Yr;Day) | Today's Date |

### *The IBM Standard Tape Label — Type A*

The schematic (Figure 26) shows the standard header- and trailer-label fields that can be processed by IOCS.

#### Header Label

In this 120-character label, the first 66 positions are used for nineteen tape header-label fields. They contain standard identifying information, which can be processed by IOCS. The next 34 positions are reserved for other functions and must not be used for tape header-label data in a 1401/1460 system. The last 20 positions are blank and may contain additional header-label information supplied and handled by the user, if desired.

Whenever complete checking (DTF CHECKLABEL ALL for an input tape) or writing (output tape) is specified for header labels, the first seven fields are always processed. (The eighth field is reserved for future use.) However, fields 9-19 are checked or written *only if at least one* of them is specified in its corresponding DTF entry When ALL is specified in DTF

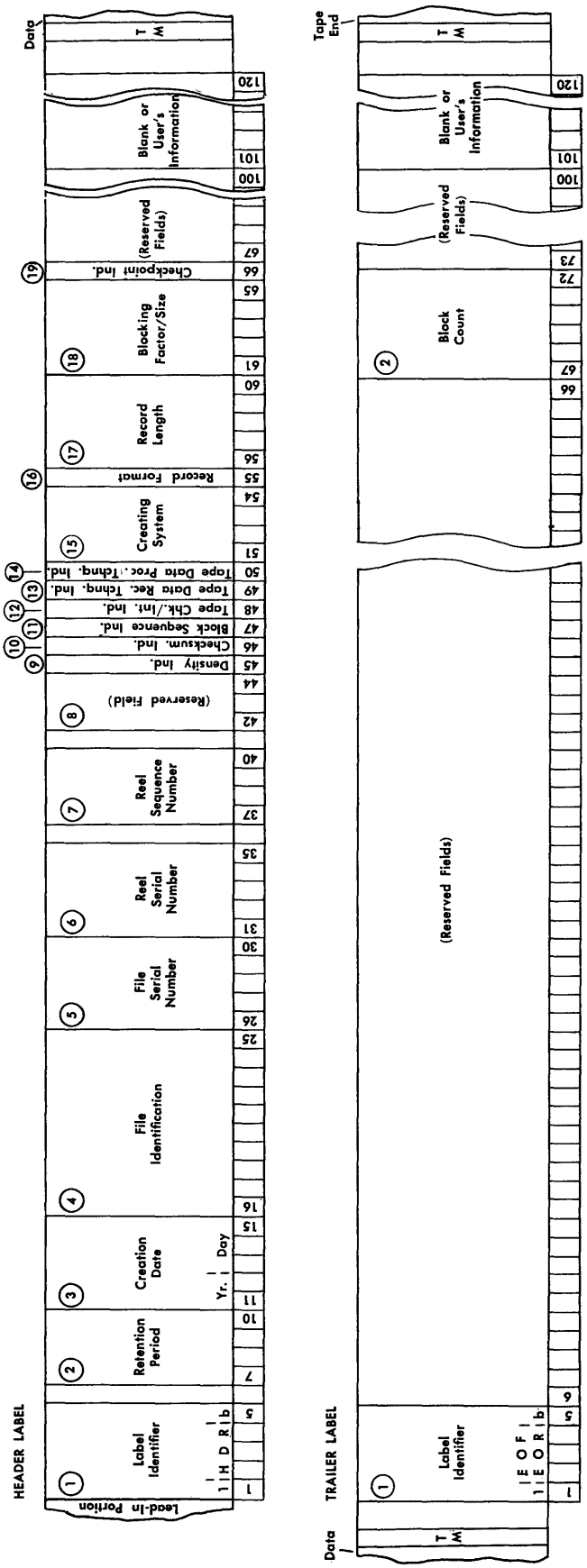



Figure 26. Schematic of IBM Standard Tape Label Fields Processed by IOCS (Type A)

CHECKLABEL and a DTF specification is given for any one of these eleven fields on an *input* tape, IOCS checks all nineteen fields. When standard labels are written and a DTF specification is given for any one of these eleven fields for an *output* tape, IOCS writes the data specified for all nineteen fields.

Following the header label, IOCS automatically writes a tape mark when tape output files are processed. For an input tape, IOCS automatically bypasses this tape mark.

The nineteen identifying fields are:

1. *Label Identifier* – consists of the digit 1, the letters HDR, and a blank position.

   When a new reel of tape is received into a system of labeled files in an installation and before it is used in an IOCS job, the characters "IHDRb" should be written in this field as a header-label identification.

   Input Tape: This field identifies the first record as the header label.

   Output Tape: This field is written automatically by the IOCS routines.

2. *Retention Period* – 4 digits.

   This field indicates the number of days after the creation date that this file is to be retained.

   Input Tape: When the header label is to be completely checked, the number of days must be specified in the DTF entry HEADER.

   Output Tape: The number of days to be recorded must be specified in the DTF entry HEADER.

3. *Creation Date* – 5 digits.

   The contents of this field indicate the year and the day of the year this file was created. The year (00-99) is entered in the first 2 positions of the field, followed by the day of the year (001-366) in the last 3 positions.

   Input Tape: When the header label is to be completely checked, this date must be specified in the DTF entry HEADER.

   Output Tape: The date is taken from today's date in storage. The user must load the date in storage positions 82-86.

4. *File Identification* – 10 alphamerical characters or blanks.

   This field identifies the entire file by name, such as a job name.

   Input Tape: When the header label is to be checked (either completely or partially), this name must be specified in the DTF entry HEADER.

   Output Tape: The name to be recorded must be specified in the DTF entry HEADER.

5. *File Serial Number* – 5 alphamerical characters or blanks.

   This field generally identifies the entire file by number, such as a job number.

   Input Tape: When the header is to be completely checked, this number must be specified in the DTF entry SERIALNUM.

   Output Tape: The number to be recorded must be specified in the DTF entry SERIALNUM, unless the user plans to have this number the same as the *reel* serial number. If DTF SERIALNUM is not included, the reel serial number of the *first* reel is written as the file serial number on all reels.

6. *Reel Serial Number* – 5 alphamerical characters or blanks.

   This is the number of the tape reel assigned within the installation. When the reel is first received, and before it is used for any data, it should be assigned the next available number. Generally, the number is also written on the physical label of the reel for visual identification. This number is not affected by IOCS routines.

7. *Reel Sequence Number* – 4 digits.

   This indicates the order of the reel within a multi-reel file. IOCS automatically assigns the number 0001 to the first reel of data, unless some other number is specified in the DTF entry REELSEQ.

   Input Tape: When the header label is completely checked, IOCS checks this field in all reels for the file. For this, either 0001 or the specified number is used to check the first reel. Any additional reels are checked with the next sequential number.

   Output Tape: IOCS writes either 0001 or the specified number on the first reel, and thereafter serially numbers any additional reels.

8. *Reserved* – 3 positions.

   This field is reserved for future use. At this time the field should be blank for a 1401/1460 IOCS job.

9. *Density Indicator* – 1 digit.

   This field indicates the density used for the data recorded on the tape. A code specifies high or low density:

| *Code* | *Meaning* |
|---|---|
| 0 | Not applicable |
| 1 | Low Density |
| 2 | High Density |

   For either checking or writing, IOCS considers this code as "0" unless code "1" or "2" is specified in the DTF entry DENSITY.

   Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

   Output Tape: When fields 9-19 are written, IOCS writes "0" or the code specified in the DTF entry DENSITY. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

10. *Checksum Indicator* – 1 digit.

    This field contains a code to indicate the type of checksum, if any is present:

| *Code* | *Meaning* |
|---|---|
| 0 | Not applicable or no checksum present. |
| 1-9 | To be assigned by IBM for various types of checksums. |

    For either checking or writing, IOCS considers this code as "0" unless code "1-9" is specified in the DTF entry CHKSUM.

    Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

    Output Tape: When fields 9-19 are written, IOCS writes "0" or the code specified in the DTF entry CHKSUM. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

11. *Block Sequence Indicator* – 1 digit.

    This field contains a code to indicate the type of block sequencing, if applicable:

| *Code* | *Meaning* |
|---|---|
| 0 | Not applicable or no block sequence field used. |
| 1-9 | To be assigned by IBM for various types of block sequence fields. |

    For either checking or writing, IOCS considers this code as "0" unless a code "1-9" is specified in the DTF entry BLKSEQIND.

    Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

    Output Tape: When fields 9-19 are written, IOCS writes "0" or the code specified in the DTF entry BLKSEQIND. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

12. *Tape Checking/Interpreting Indicator* – 1 digit.

    This field indicates the coding used to write data on tape:

| *Code* | *Meaning* |
|---|---|
| 1 | Binary |
| 2 | BCD |
| 3 | Not applicable |

    For either checking or writing, IOCS considers this code as "3" unless code "1" or "2" is specified in the DTF entry TPCHKIND.

    Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

    Output Tape: When fields 9-19 are written, IOCS writes "3" or the code specified in the DTF

entry TPCHKIND. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

13. *Tape Data Recording Technique Indicator* — 1 digit.

This field contains the number of bits (not including a check bit) used as a unit (byte) for characters recorded on the tape. The IBM 729 or the IBM 7330 attached to a 1401/1460 system processes data recorded in 6-bit code. Therefore, for either checking or writing, IOCS assumes this field contains "6." The digit "6" may be specified in the DTF entry TPRECTCH, if desired.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes a "6" in this field. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

14. *Tape Data Processing Technique Indicator* — 1 digit.

This field indicates the number of bits (not including a check bit) out of a byte that are to be treated as a unit in processing. For either checking or writing, IOCS considers this field as blank, unless a number is specified in the DTF entry TPPROTCH.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes a blank or the number specified in the DTF entry TPPROTCH. If none of the fields 9-19 is DTF-specified, this field will be blank.

15. *Creating System* — 4 digits.

This field identifies the system that created this tape file. For either checking or writing, IOCS assumes this field contains "1401," unless some other system is specified in the DTF entry CREATSYS.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes "1401" or the system number specified in the DTF entry CREATSYS. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

16. *Record Format* — 1 digit.

This field contains a code that indicates the record format of the file:

| *Code* | *Meaning* |
|---|---|
| A | Fixed-length, unblocked records |
| F | Fixed-length, blocked records |
| W | Variable-length, unblocked records |
| X | Variable-length, blocked records |

For either checking or writing, IOCS considers this field as blank unless a code is specified in the DTF entry FMRECORD.

Input Tape: When the header label is completely checked this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes a blank or the code specified in the DTF entry FMRECORD. If none of the fields 9-19 is DTF-specified, this field will be blank.

17. *Record Length* — 5 digits.

For fixed-length data records, this field contains the number of characters per record. For variable-length data records, this field contains the number of characters in the largest possible record in this file. For either checking or writing, IOCS considers this field as blank unless a number is specified in the DTF entry LNGRECORD.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes blanks or the number specified in the DTF entry LNGRECORD. If none of the fields 9-19 is DTF-specified, this field will be blank.

18. *Blocking Factor/Size* — 5 digits.

For fixed-length data records, this field contains the number of rceords within each block (tape record). For variable-length data records, this field contains the number of characters in the largest possible block (tape record) of this file. For checking or writing, IOCS considers this field as blank unless a number is specified in the DTF entry BLKFACTOR.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes blanks or the number specified in the DTF entry BLKFACTOR. If none of the fields 9-19 is DTF-specified, this field will be blank.

19. *Checkpoint Indicator* — 1 digit.

This field contains a code to indicate the type of checkpoint records, if any are used:

| *Code* | *Meaning* |
|---|---|
| 0 | Not applicable or no checkpoint record. |
| 1-9 | To be used as required by various types of checkpoint records. |

For checking or writing, IOCS considers this code as "0" unless a code "1-9" is specified in the DTF entry CHPTRCD.

Input Tape: When the header label is completely checked, this field is checked if any field 9-19 is DTF-specified.

Output Tape: When fields 9-19 are written, IOCS writes a "0" or the code specified in the DTF entry CHPTRCD. However, if none of the fields 9-19 is DTF-specified, this field will be blank.

### Trailer Label

Positions 1-5 and 67-72 in the standard 120-character label are used for two tape trailer-label fields. They contain standard information and can be processed by IOCS. The last 20 positions are blank and may contain additional trailer-label information supplied by the user, if desired. All other positions are reserved for other functions and must not be used for tape trailer-label data in a 1401/1460 system.

The two trailer-label fields in the standard 120-character label are:

1. *Label Identifier* – Consists of the digit 1, the letters EOF or EOR, and a blank position.

   This field indicates either the end of data for this file (1EOFb), or the end of data on this reel with another reel of data to follow for this file (1EORb).

   Input Tape: This field identifies the record as a trailer label and indicates if another reel follows.

   Output Tape: The characters "1EOFb" are written automatically by IOCS routines when a CLOSE macro instruction is issued. The characters "1EORb" are written automatically by IOCS routines when the tape reflective marker is sensed or when a FEORL macro instruction is issued.

2. *Block Count* – 6 digits.

   A count of the number of blocks processed is always accumulated automatically by the IOCS routines.

   Input Tape: When the trailer is checked, the count accumulated is compared to the count recorded on the tape when it was written.

   Output Tape: The accumulated count is written by IOCS routines.

## *Standard 80-Character Label – Type B*

The schematic (Figure 27) shows the standard header- and trailer-label fields that can be processed by IOCS.

### Header Label

In this 80-character label, the first 40 positions are used for seven fields that contain standard identifying information, and the remaining 40 positions are blank. The first two fields (*Label Identifier* and *Tape Serial Number*) permanently identify the reel itself. The other five fields identify the job and are automatically written (or checked) with information specified in the DTF entries. Information may be written or checked in the last 40 positions, if desired, by the user's program (see *Label Operation*).

The seven identifying fields are:

1. *Label Identifier* – consists of the digit 1, the letters HDR, and a blank position.

   When a new reel of tape is received in an installation and before it is used in an IOCS job, the characters 1BLNK should be written in this field as a temporary header identification. The first time records are written by IOCS, 1BLNK is automatically replaced with the standard header identification (1HDRb).

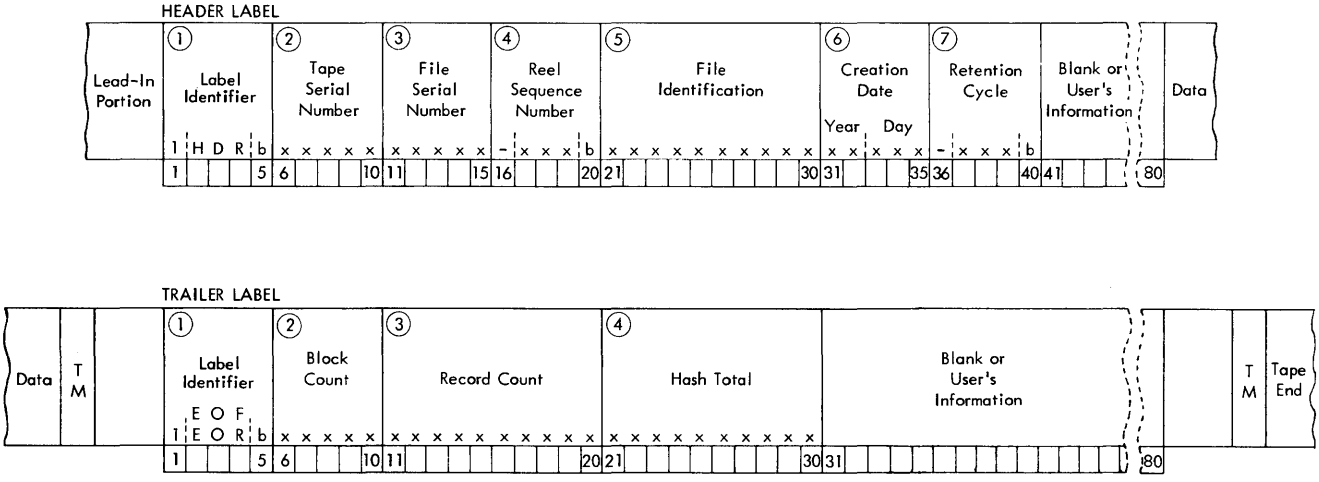



Figure 27. Schematic of Standard 80-Character Tape Label Fields Processed by IOCS (Type B)

Input Tape: This field identifies the first record as the header label.

Output Tape: This field is written automatically by the IOCS routines.

2. *Tape Serial Number* – 5 digits.

   This is the sequential number of the tape reel within the whole installation. As soon as a new reel of tape is received in the installation, it should be identified in this field with the next available number. The IOCS routines do not affect this number.

3. *File Serial Number* – 5 digits.

   This generally represents a job number in an installation.

   Input Tape: When the header is to be completely checked, this number must be specified in the DTF entry SERIALNUM.

   Output Tape: The number to be recorded must be specified in the DTF entry SERIALNUM, unless the user plans to make this number the same as the *tape* serial number. If the tape serial number is used and the job requires two or more reels, the tape number of the *first* reel becomes the file serial number on all reels.

4. *Reel Sequence Number* – consists of a minus sign, 3 digits, and a blank position.

   This field is used to number the reels in a multireel job. The number of the first reel is 001 unless the user specifies some other number (3 digits) in the DTF entry REELSEQ.

   Input Tape: When the header is to be completely checked, reel sequence is checked automatically.

   Output Tape: Reels, after the first, are serially numbered automatically.

5. *File Identification* – 10 alphamerical characters or blanks.

   This field identifies the entire file by name, such as a job name.

   Input Tape: When the header is to be checked (either completely or partially) this name must be specified in the DTF entry HEADER.

   Output Tape: The name to be recorded must be specified in the DTF entry HEADER.

6. *Creation Date* – 5 digits.

   The dating system consists of 2 digits for the year, followed by 3 digits for the day of the year.

   Input Tape: When the header is to be completely checked, this date must be specified in the DTF entry HEADER.

   Output Tape: The date is taken from today's date in storage. The user must load the date in storage positions 82-86.

7. *Retention Cycle* – consists of a minus sign, 3 digits, and a blank position.

   This specifies the number of days after the creation date that the tape is to be kept active.

   Input Tape: When the header is to be completely checked, the number of days (3 digits) must be specified in the DTF entry HEADER.

   Output Tape: The number of days (3 digits) to be recorded must be specified in DTF HEADER.

### Trailer Label

In the standard 80-character trailer label, the first 30 positions are used for four fields, which can be processed by IOCS, and the remaining 50 positions are blank. The first field indicates whether or not this is the last reel of data for a job. The other three fields provide control totals and are written (or checked) with totals accumulated during the run, if specified in the DTF entries. The last 50 positions may contain information supplied and handled by the user, if desired. (See *Label Operation.*)

The four fields in the standard trailer label are:

1. *Label Identifier* – consists of the digit 1, the letters EOF or EOR, and a blank position.

   The letters EOF (end of file) indicate that this is all the data for the job. EOR (end of reel) indicates that this is the end of a reel of tape but one or more reels follow for the same job.

   Input Tape: The field identifies the record as a trailer label and indicates if another reel follows.

   Output Tape: The letters EOF are written automatically when a CLOSE macro instruction is given. The letters EOR are written automatically when the tape reflective spot is sensed or when a FEORL macro instruction is given.

2. *Block Count* – 5 digits.

   A count of the number of blocks processed is always accumulated automatically by the IOCS routines. In the case of unblocked records, this count is the same as a record count.

   Input Tape: When the trailer label is to be checked, the count accumulated is compared to the count recorded on the tape when it was written.

   Output Tape: The accumulated count is written automatically.

3. *Record Count* – 10 digits.

   This is a count of the number of records processed. With unblocked records, this is the same as the block count.

   Input Tape: When a trailer that contains a record count is to be checked, this must be specified in the DTF entry TOTALS.

   Output Tape: This count is accumulated and written automatically if it is specified in the DTF entry TOTALS. If it is not specified, blanks are written in this field.

4. *Hash Total* – 10 digits.

   A hash total is the total of some item, such as customer number, for processing-control purposes. It is not intended to be a significant total such as the total dollar amount of sales would be.

   Input Tape: When a trailer that contains a hash total is to be checked, the low-order position of the same field selected when the tape was written must be specified in the DTF entry TOTALS.

   Output Tape: The hash total is accumulated and written automatically if the low-order position of the selected field in the rceord is specified in the DTF entry TOTALS. If it is not specified, blanks are written in this field.

## *Standard 84-Character Label – Type C*

The schematic (Figure 28) shows the standard header- and trailer-label fields that can be processed by IOCS.

### Header Label

In this 84-character label, the first 60 positions are used for eight tape header-label fields. They contain standard identifying information, which can be processed by IOCS. The next 12 positions are reserved for other functions and must not be used for tape header-label data in a 1401/1460 system. The last 12 positions are blank and may contain additional header-label information supplied and handled by the user, if desired.

The eight identifying fields are:

1. *Label Identifier* – consists of the digit 1, the letters HDR, and two blank positions.

   When a new reel of tape is received in an installation and before it is used in an IOCS job, the characters "1HDRbb" should be written in this field as a header-label identification.

   Input Tape: This field identifies the first record as the header label.

   Output Tape: This field is written automatically by the IOCS routines.

2. *Tape Serial Number* – consists of a blank position and 5 digits.

   This is the number of the tape reel assigned within the installation. When the reel is first received, and before it is used for data, it should be assigned the next available number. Generally, this number is also written on the physical label of the reel for visual identification. The IOCS routines do not affect this field.

3. *File Serial Number* – consists of a blank position and 5 digits.

   This field identifies the entire file by number, such as a job number.

   Input Tape: When the header is to be completely checked, this number must be specified in the DFT entry SERIALNUM.

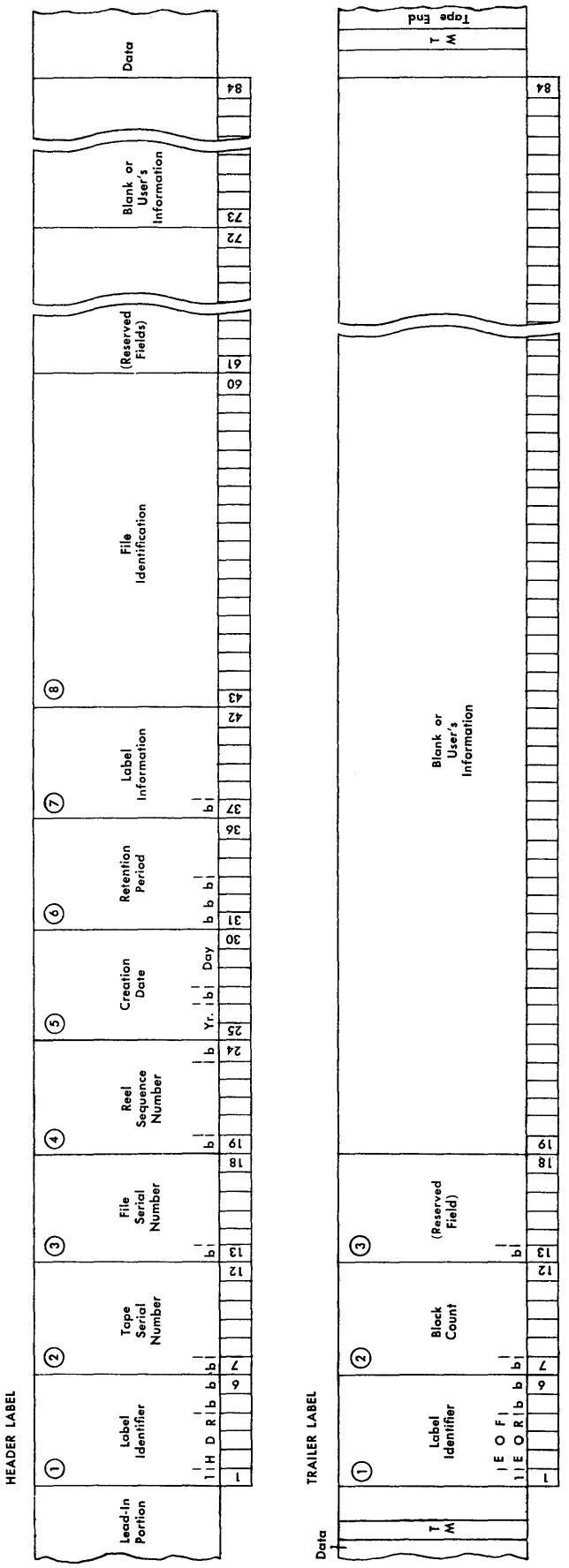



Figure 28. Schematic of Standard 84-Character Tape Label Fields Processed by IOCS (Type C)

Output Tape: The number to be recorded must be specified in the DTF entry SERIALNUM, unless the user plans to make this number the same as the *tape* serial number. If DTF SERIALNUM is not included, the tape serial number of the *first* reel is written as the file serial number on all reels.

4. *Reel Sequence Number* – consists of a blank position, 4 digits, and a blank position.

   This field is used to number the reels of data in a multi-reel file. The number of the first reel of data is 0001 unless some other number is specified in the DTF entry REELSEQ.

   Input Tape: When the header label is completely checked, IOCS checks this field in all reels for the file. For this, either 0001 or the specified number is used to check the first reel. Any additional reels are checked with the next sequential number.

   Output Tape: IOCS writes either 0001 or the specified number on the first reel, and thereafter serially numbers any additional reels.

5. *Creation Date* – consists of 2 digits, a blank position, and 3 digits.

   The dating system consists of 2 digits for the year, followed by a blank, and then followed by 3 digits for the day of the year.

   Input Tape: When the header label is to be completely checked, this date must be specified in the DTF entry HEADER.

   Output Tape: The date is taken from today's date in storage. The user must load the date in storage positions 82-86.

6. *Retention Period* – consists of 3 blank positions and 3 digits.

   This indicates the number of days this file is to be retained after the creation date.

   Input Tape: When the header label is to be completely checked, the number of days must be specified in the DTF entry HEADER.

   Output Tape: The number of days to be recorded must be specified in the DTF entry HEADER.

7. *Label Information* – consists of a blank position and 5 digits.

   This field contains five separate codes that indicate factors applicable to this data file. The codes available for the individual columns of this field are:

| *Column* | *Factor* | *Code* |
|---|---|---|
| 37 | ... | Blank |
| 38 | Density | 0 – Low<br>1 – High |
| 39 | Character Coding | 0 – BCD<br>1 – Binary |
| 40 | Checksum | 0 – No checksum<br>1 – Checksum present |
| 41 | Block Sequence | 0 – No block sequence field used<br>1 – Block sequence field present |
| 42 | Checkpoint Record | 0 – No checkpoint record<br>1 – Checkpoint record present |

   For either checking or writing, IOCS considers each position of this field as blank unless a code is specified in the DTF entry LABINFO.

   Input Tape: When the header label is completely checked, IOCS checks the columns of this field against the codes specified in the DTF entry LABINFO, or as blank if codes are not DTF-specified.

   Output Tape: IOCS writes blanks or the codes specified in the DTF entry LABINFO.

8. *File Identification* – 18 alphamerical characters or blanks.

   This field identifies the entire file by name, such as a job name.

   Input Tape: When the header is to be checked (either completely or partially), this name must be specified in the DTF entry HEADER.

   Output Tape: The name to be recorded must be specified in the DTF entry HEADER.

### Trailer Label

The first 12 positions in the standard 84-character label are used for two tape trailer-label fields. They contain standard information and can be processed by IOCS. The next 6 positions are reserved for use with the IBM 7090 system. The last 66 positions are blank and may contain additional trailer-label information supplied and handled by the user, if desired.

The three trailer-label fields in the standard 84-character label are:

1. *Label Identifier* – consists of the digit 1, the letters EOF or EOR, and two blank positions.

   The letters EOF (end-of-file) indicate that this is all the data for this file. The letters EOR (end-of-reel) indicate the end of data on this reel but that another reel of data is to follow for this file.

   Input Tape: This field identifies the record as a trailer label and indicates if another reel follows.

   Output Tape: The characters "1EOFbb" are written automatically by IOCS routines when a CLOSE macro instruction is issued. The characters "1EORbb" are written automatically by IOCS routines when the tape reflective marker is sensed or when a FEORL macro instruction is issued.

2. *Block Count* – consists of a blank position and 5 digits.

   A count of the number of blocks processed is always accumulated automatically by IOCS routines.

   Input Tape: When the trailer label is checked, the count accumulated is compared to the count recorded on the tape when it was written.

Output Tape: The accumulated count is written by IOCS routines.

3. *Reserved for "Unit Control Word"* — consists of a blank position and 5 digits.

   This field is reserved for those users who intend to process this reel of data on the IBM 7090 system. It can be handled by using the appropriate exit from the IOCS routines.

## Label Operation

Tape labels are processed by IOCS *open, close,* and *end-of-reel* routines. The operations performed when a file is opened or closed, or when an end-of-reel condition occurs, are affected by the types of labels and the specifications in the DIOCS and DTF entries.

If *standard labels* are specified, the first record read from the tape reel, or written on it, is assumed by IOCS to be a header label. Similarly, the last three records on the reel are assumed to be a tape mark, a trailer label, and another tape mark. The IOCS routines set up an area (120, 84, or 80 positions) into which the label is read from tape for automatic checking, or in which information is built from the DTF specifications for writing an output label. This area has a symbolic address of IOCSLB (IOCS Standard Label). The programmer can use this address if he wishes to modify the standard label. However, he must *never* leave any word marks in this IOCSLB area.

If *nonstandard labels* are specified, IOCS does not set up the IOCSLB area, but does provide *exits* for the programmer to read, write, and/or check the labels himself.

If *no labels* are specified, IOCS assumes that the first record on tape consists of data, *not* header information, and that no record follows the tape mark at the end of data.

### Exits

In order to modify standard labels or to handle nonstandard labels, seven *exits* from the IOCS routines are provided to branch to the programmer's own routines. Each of the exits occurs at a specific time in the IOCS open, close, or end-of-reel routine and is used for a specific function. The programmer must specify the appropriate one for the operation he wishes to perform, by selecting the corresponding DTF entry (EX1ADDR, EX2ADDR, --------EX8ADDR) and entering the symbolic address of his routine in the operand field. In his program, the user returns to the IOCS routines by branching to IOCSRE (IOCS Return) at the end of his own routine. An eighth exit is provided to permit including a user's routine *before* a trailer label is written, when the end of a reel of output tape is signalled by the reflective spot on the tape.

The specific functions of the eight exits (Figure 29) are:

- *Exit 1* allows the programmer to modify a *standard output trailer.* This is used to enter information in the *User's Information* area of the label, or to change any information in the fields handled by the IOCS. In his routine, the programmer moves the information to the desired positions in IOCSLB and returns to IOCS by IOCSRE. The trailer is then written on tape by the IOCS routines.
- *Exit 2* is used to build and write *nonstandard output trailers* or *additional labels* following a standard output trailer. The user must provide his own area for building the trailer information and must program to write the trailer on tape before returning to IOCS by IOCSRE.

The totals accumulated by IOCS throughout the run are available at these symbolic addresses:

| | | |
|---|---|---|
| Hash Total | IOCnHS | (10-position field) |
| Record Count | IOCnRC | (10-position field) |
| Block Count | IOCnBK | (5- or 6-position field) |

The letter "n" is the number of the original tape unit assigned to the file in the DTF CHANDRIVE entry. The hash total and record count are accumulated only if they are specified in the DTF TOTALS entry.

| EXIT NO. | HEADER | | TRAILER | | USE |
|---|---|---|---|---|---|
| | Standard | Non-Standard | Standard | Non-Standard | |
| 1 | | | Output | | Modify standard trailer |
| 2 | | | | Output | Build and write non-standard trailers |
| 3 | Output | | | | User checks old standard header |
| 4 | Output | | | | Modify standard header |
| 5 | | Output | | | Read and check old nonstandard headers; build and write non-standard headers |
| 6 | | | Input | Input | Additional check of standard trailer; read and check nonstandard trailers; modify reel count |
| 7 | Input | Input | | | Additional check of standard header; read and check nonstandard headers |
| 8 | | | | | User's routine at end of reel (reflective spot) on output tape |

Figure 29. Exits from IOCS Routines for Tape Header and Trailer Labels

- *Exit 3* permits the user to check a *standard header* on an *output* tape, instead of the IOCS *open* routine checking the retention cycle automatically. The programmer can obtain the header information for checking from IOCSLB.
- *Exit 4* allows the programmer to modify a *standard output header*. This is used to enter information in the *User's Information* area of the label, or to change any information in the fields handled by IOCS. In his routine, the programmer moves the information to the desired positions in IOCSLB and returns to the *open* routine by IOCSRE. The header is then written on tape by the IOCS *open* routine.
- *Exit 5* is used to build and write *nonstandard output headers* or *additional labels* following a standard output header. The user must provide his own area for building the header information and must program to write the header on tape before returning to the *open* routine by IOCSRE. When nonstandard labels are specified, this exit can also be used to read and check the old header on an output tape.
- *Exit 6* permits:
  A. Additional checking of a *standard input trailer.*
     The programmer obtains the information to be checked from IOCSLB.
  B. reading and checking *additional labels* following a standard input trailer.
     The user must provide his own area for entering the trailer information and he must program to read and check the information. Programming must be returned to IOCS by branching to IOCSRE.
  C. reading and checking *nonstandard input trailers.*
     The user must provide his own area for entering the trailer information and he must program to read and check the information. Programming is returned to IOCS by branching to IOCSRE on an *end-of-reel* condition, but a branch to the end-of-file address (specified in DTF EOFADDR) should be programmed for an *end-of-file* condition.
     The totals accumulated by IOCS throughout the run are available at these symbolic addresses whenever Exit 6 is used:

| | | |
|---|---|---|
| Hash Total | IOCnHS | (10-position field) |
| Record Count | IOCnRC | (10-position field) |
| Block Count | IOCnBK | (5- or 6-position field) |

     The letter "n" is the number of the original tape unit assigned to the file in the DTF CHANDRIVE entry. The hash total and record count are accumulated only if they are specified in the DTF TOTALS entry.
  D. modifying the reel sequence count.
     This count is stored in a field with a symbolic address of IOCnSQ (IOCS Sequence), in which "n" is the original drive number assigned to the file in the DTF CHANDRIVE entry. The count is not increased before branching from Exit 6, but is automatically increased by one after returning to IOCS by IOCSRE.
- *Exit 7* permits:
  A. additional checking of a *standard input header.*
     The programmer obtains the information to be checked from IOCSLB.
  B. reading and checking *nonstandard input headers* or *additional labels* following a standard input header.
     The user must provide his own area for entering the header information and must program to read and check the information before returning to the open routine by IOCSRE.
- *Exit 8* allows the programmer to enter his own routine when the reflective spot is reached, and before a tape mark is written, at the end of a reel of *output tape.* At the end of his routine, the programmer must issue either a FEORL macro instruction to write an end-of-reel trailer and process the header on the next reel, or a CLOSE macro instruction to write an end-of-file trailer. He cannot return by IOCSRE.

**Nonstandard Labels**

For completely automatic operation by IOCS routines, a header or trailer label may be considered *standard* only if it has the length (120, 84, or 80 characters) and format of one of the three types (A, B, or C) specified as standard. Thus, a label is *nonstandard* if it:

1. has a different format.
2. has a different length, regardless of whether or not the fields are in the standard format.
3. is an additional label after the first. That is, any additional labels are considered nonstandard regardless of format. IOCS considers only *one* header and *one* trailer label as standard. If the first label is standard length and format, *standard* can be specified in the DTF entries for automatic reading, writing, and/or checking of the first label. The additional labels are then read, written, and/or checked using the proper exits.

A modification of these rules can be made to gain some of the advantages of automatic standard-label operation for *nonstandard* labels. That is, a *nonstandard* label can be automatically read into, or written from, the IOCSLB area by specifying the labels as *standard* and by specifying the size of the label (A, B, or C) in the DTF entry (TYPELABEL). However, checking *must* be programmed by the user (via Exit 6 or 7) and *must not* be specified in the DTF entry (CHECKLABEL). Furthermore, the information for labels on an output tape *must* be entered in IOCSLB by use of Exit 1 or 4, and cannot be specified in the DTF entries.

Before the first record can be read from any input file, or written on any output file, that file must be readied for use by the IOCS routines. Similarly, after all records have been processed for a file, that file must be removed from use. For these operations IOCS makes two basic macro instructions (OPEN and CLOSE) available to the programmer. Three others (RDLIN, FEORL, and DCLOS) are provided for special conditions.

## *OPEN Macro*

This instruction (Figure 30) is used to activate each file that is to be used: card reader, card punch, disk, tape, and printer. To open a file, the symbolic name of the file assigned in the DTF entry is entered in the operand field. Two or more files may be opened with one instruction. In this case any tape, card and/or printer-file names must be entered in the operand field ahead of any disk-file names. Any disk-file name, or group of names, must always be preceded by the letter *D*, regardless of whether disk files are opened alone or with other files. Tape card, and printer-file names may be preceded by the letter *T*, but this is not required. IOCS assumes that files are in this category if a *D* is not included. A maximum of ten operands (including *T* and *D*) may be entered in one OPEN instruction. Each operand, except the last, must be followed by a comma.

For the card reader, card punch, or printer, OPEN simply makes the file available for reading, punching, or writing.

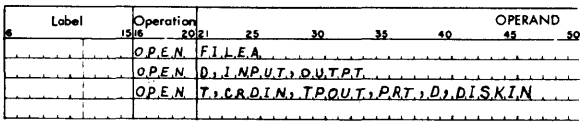



Figure 30. OPEN Macro Instruction

### Disk File

The OPEN instruction makes a disk file available for reading or writing. When a 1311 disk *input* file has a header label, IOCS checks the label as specified in the DTF entry CHECKLABEL. The header-label fields are checked with the information supplied by the programmer in the DTF entries or in RDLIN information cards (see *RDLIN Macro*). When a disk *output* file requires a header label (specified in the DIOCS entry DLABDEF), IOCS checks all header labels on the pack to ensure that no active data occupies the area specified for the new output file. Then IOCS writes the header-label information supplied in the DTF entries or in RDLIN information cards.

If an input file of disk records processed in consecutive or control-sequential order is re-opened after a CLOSE, processing continues with the next record in sequence.

### Tape Input File

The OPEN instruction rewinds the tape according to the specification in the DTF entry, (REWIND). If the tape does not contain a header label, it is ready for the first GET instruction to read data.

When an input tape contains a *standard header label*, the header is automatically read and checked as specified in the DTF entry (CHECKLABEL). The header fields are checked with the information supplied by the user in the DTF entries (HEADER, SERIALNUM, and REELSEQ) or in RDLIN information cards (see *RDLIN Macro*). Additional checking of the standard header in the IOCSLB area, or reading and checking of additional (nonstandard) labels, may be done by using Exit 7. In either case, programming must be returned to the *open* routine by branching to IOCSRE. A tape mark (if any) is passed if specified in the DTF entry (TYPELABEL), and the file is ready for the first GET instruction.

When the tape contains *nonstandard header labels*, the programmer can read and check the label information by using Exit 7. At the end of his routine, he must return to the IOCS *open* routine by branching to IOCSRE. Then a tape mark (if any) is passed if specified in the DTF entry (TYPELABEL) and the file is ready for the first GET instruction.

### Tape Output File

The OPEN instruction rewinds the tape according to the specification in the DTF entry (REWIND). If header labels are not used, the tape is ready for data.

A reel of tape used for output generally contains data from a previous job. To make sure that this data is no longer active and may be destroyed, IOCS reads the header label and checks the retention cycle when *standard labels* are specified. If the user does not want automatic checking, however, he can program to check the retention cycle and any other fields of the standard header (in the IOCSLB area), by using Exit 3 and returning to the open routine via IOCSRE. The tape is

then backspaced and ready for the new output header. This is written automatically, with the information supplied by the user in the DTF entries (HEADER, SERIALNUM, and REELSEQ) or in RDLIN information cards (see *RDLIN Macro).* The new standard header may be modified by using Exit 4, and additional (nonstandard) header labels may be built and written by using Exit 5. In either case, programming must be returned to the *open* routine by branching to IOCSRE. If a Type A standard header label has been written, IOCS writes a tape mark and the tape is then ready for data. If a Type B or Type C standard header label has been written, the tape is ready for data (IOCS does *not* write a tape mark).

When *nonstandard labels* are specified, the programmer can read and check the old header label, backspace the tape, and build and write his output header label, using Exit 5. After the label is written on tape, programming must be returned to the IOCS *open* routine by branching to IOCSRE. The tape is then ready for data.

## CLOSE Macro

This instruction (Figure 31) is used to deactivate each file that has been used: card reader, card punch, disk, tape, and printer. To close a file, the symbolic name of the file assigned in the DTF entry is entered in the operand field. Two or more files may be closed with one instruction. In this case, any tape, card, and/or printer-file names must be entered in the operand field ahead of any disk-file names. Any disk-file name, or group of names, must always be preceded by the letter *D,* regardless of whether disk files are closed alone or with other files. Tape, card, and printer-file names may be preceded by the letter *T,* but this is not required. IOCS assumes that files are in this category if a *D* is not included. Each letter (T or D) and file name, except the last, must be followed by a comma.

A file may be closed at any time by issuing this macro instruction. If further processing of this same file is required, the file must be opened again. If a *labeled disk* file is reopened, the header-label specifications must be redefined by issuing a RDLIN macro instruction (see *RDLIN Macro — Disk and Tape Records).*

For the card reader or printer, CLOSE simply makes the file unavailable for use. When the last card in a card input file is read, IOCS branches to the end-of-file routine (specified in the DTF entry EOFADDR) where the programmer issues the CLOSE instruction. For the card punch, this instruction causes a final dummy card to be fed, so that the last card record can be checked. (Checking occurs on the card feed cycle following punching.) Then the file is made unavailable for use.

Whenever overlapping of card punch or printer operations with disk-seek operations is planned (specified in the DIOCS entries PCHOVLP or PRNTOVLP) and a record remains in the output area at the end of a job, CLOSE punches, or prints, that record.

### Disk File

The CLOSE instruction makes a disk file unavailable for use. When disk records are processed in consecutive or control-sequential order, IOCS detects the end of an *input* file by reading the EOF trailer label, or by reaching the *ending limit* of the file, whichever occurs first. Following the last record of an input file, IOCS automatically branches to the programmer's end-of-file address specified in the DTF entry EOFADDR. In his routine the programmer may perform any processing of data required to finish his job, and he normally issues the CLOSE instruction. The end of a *disk output* file is determined by the user's program, and CLOSE causes an EOF trailer label to be written.

### Tape Input File

The CLOSE macro instruction rewinds, or rewinds and unloads, the input tape according to the DTF specification (REWIND), and it deactivates the file.

In most jobs, a file of input data is *ready* to be closed after all data in the file has been read. When *standard labels* are DTF-specified, IOCS automatically reads and checks the trailer label, and determines that this *is* an end-of-file condition. Then IOCS automatically branches to the end-of-file address specified in the DTF entry (EOFADDR). In this end-of-file routine the programmer issues the CLOSE macro instruction. He may also, in this routine, perform any processing of data that he requires to finish his job. If no processing is required and CLOSE is the only instruction, it must be labelled with the symbolic address specified in the DTF entry (Figure 32).

Before closing an input file the programmer may, if

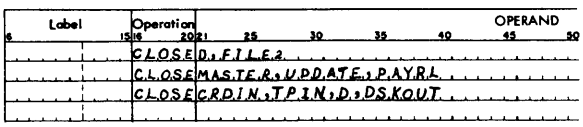



Figure 31. CLOSE Macro Instruction

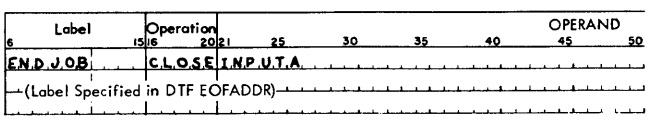



Figure 32. CLOSE Instruction for an Input File

he wishes, completely check a *standard trailer* himself, rather than checking automatically by IOCS (DTF CHECKLABEL not specified). In this case, he uses Exit 6 and he must program to branch to his end-of-file address (specified in DTF EOFADDR), where he issues the CLOSE instruction.

When *nonstandard labels* are specified, the trailer label may be read and checked using Exit 6. At the end of his checking routine, if an end-of-file condition exists, the programmer should branch to his end-of-file address (specified in DTF EOFADDR) where he issues the CLOSE instruction. If the user does not care to check his nonstandard trailer label (does not use Exit 6), or if labels are not DTF-specified, IOCS automatically branches to the DTF-specified end-of-file address, where CLOSE is issued.

If, for some reason, the user plans to close an input file before all the data and the tape mark have been read, he writes the CLOSE instruction at the desired place in his program. This merely rewinds, or rewinds and unloads, the tape and deactivates the file. No trailer labels are read or checked.

#### Tape Output File

When all the data has been processed for an output tape, the CLOSE instruction writes the last block if necessary (partially filled with data) and causes a tape mark to be written. Then, if labels are not specified, it rewinds, or rewinds and unloads, the tape as specified by the DTF entry (REWIND) and deactivates the file.

When labels are specified:

1. Standard—IOCS writes the *EOF* indication and the totals accumulated during the run. The standard trailer can be modified by using Exit 1, and additional (nonstandard) trailers can be built and written by using Exit 2. In either case, programming must be returned to the *close* routine by branching to IOCSRE.
2. Nonstandard—The programmer can build and write an output trailer using Exit 2. This should include an end-of-file identification for later use on *input*. Programming must be returned to the *close* routine by branching to IOCSRE.

After a trailer label is written, IOCS writes another tape mark, rewinds or rewinds and unloads the tape as specified by the DTF entry (REWIND), and deactivates the file.

In some cases, the data to be written on an output tape may require slightly more tape footage than is available ahead of the reflective spot at the "end" of the tape. Because a few records can be written beyond a reflective spot still leaving enough tape for threading and feeding, the programmer can plan to pass this spot and finish his output data. For this he uses Exit 8 to branch to his routine where he writes the rest of his data and issues his CLOSE instruction.

### End-of-Records for Disk Files

When records are processed in consecutive or control-sequential order, a logical file may be contained in two or more disk modules or packs. In this case, when the end of records in one module or pack is reached, IOCS locates the subsequent section of the file.

### End-of-Reel for Multi-Reel Files

When all the tape on a reel has been read or written but the job is not complete, additional reels must be processed. At the end of the reel, processing of data must be interrupted to process the trailer label, to change the reel, and to process the header of the new reel. These functions are completely handled by the IBM 1401/1460 IOCS routines when standard labels are specified, and partially handled when nonstandard labels are specified. The end of a reel of tape is signalled by a *tape mark* for input files, and by a *reflective spot* on the tape for output files.

#### Tape Input File

When *standard labels* are DTF-specified, IOCS reads and checks the trailer label, according to the DTF specifications (CHECKLABEL and TOTALS), and determines that this is *not* an end-of-file condition. Thus, another reel must be processed. The completed tape reel is automatically rewound, or rewound and unloaded, as specified in the DTF entry. If an alternate tape-drive number is specified in the DTF entry (ALTTAPE), the change to the alternate drive is made automatically. If not, a halt occurs to allow the operator time to change the reel on the tape unit. Processing restarts automatically in the first case, or by pressing the start key after the halt. The header on the new reel is read and checked in the same manner as on the proceding reel, and the processing of data resumes.

When *nonstandard labels* are specified, the trailer label can be read and checked using Exit 6. As part of the checking, the programmer determines that this is *not* an end-of-file condition and then returns to IOCS by branching to IOCSRE. Then IOCS performs the same end-of-reel functions as for standard labels: rewinds, or rewinds and unloads, the tape as specified; changes the drive if DTF-specified, or halts to allow the operator to change the tape reels; and permits reading and checking the header of the new reel before processing of data continues.

If labels are not used, an end-of-reel condition cannot be distinguished from an end-of-file. IOCS always assumes an end-of-file condition and branches to the user's EOF address specified in the DTF entry.

### Tape Output File

When the reflective spot indicates that the end of a reel of tape has been reached, a tape mark is written after the last block of data for that tape has been written. Then, if labels are not specified, the tape reel is rewound, or rewound and unloaded, according to the DTF specification, but the file is not deactivated. The operator may install another reel of tape and press the start key to continue operation.

If *standard labels* are specified:

- the totals accumulated during the run (block, record, and hash) are written,
- the standard trailer can be modified using Exit 1,
- additional (nonstandard) labels can be built and written using Exit 2.
- a tape mark is written following the trailer label,
- the tape is rewound, or rewound and unloaded, as specified.

The EOR indication is automatically included in the trailer. Then, if an alternate tape drive is DTF-specified, the change to the alternate drive is made automatically. If not, a halt occurs to permit mounting another reel on the tape unit. Precessing restarts automatically if an alternate drive has been specified, or it is restarted by pressing the start key after the halt. The header on the new reel is written in the same way as on the completed reel, and the processing of data is resumed.

If *nonstandard labels* are specified, the programmer can build and write an output trailer by using Exit 2 and returning to IOCS via IOCSRE. This label should include an end-of-reel identification for later use on *input*. Then IOCS performs the same end-of-reel functions as for standard labels; writes a tape mark; rewinds, or rewinds and unloads, the tape; changes the drive if DTF-specified, or halts to allow the operator to change the tape reel; and permits writing the header on the new reel before the processing of data continues.

As in an end-of-file operation (see *CLOSE Macro, Tape Output File*), the programmer can branch to his own routine to write additional records on the reel when the reflective spot is sensed, by using Exit 8. Because this is an end-of-reel condition rather than an end-of-file, however, he must cause an end-of-reel operation when he has finished writing his data records. For this he writes a FEORL (Forced End-of-Reel) macro instruction in his routine. Then label processing and reel changing are handled according to his specifications, before processing of data resumes.

## *RDLIN Macro — Disk and Tape Records*

Some time after the user's source program and the IOCS routines have been assembled to create the object program, it may be necessary, or advantageous, to change disk or tape header-label information, disk drive numbers, or disk file limits from those originally specified in the DTF entries. For example, the date in the header label for a recurring job, such as accounts receivable, changes constantly, and the correct date must be specified for checking when the file is used as input. If the correct date is not specified, the program halts. A change in the header-label information for output files would be needed whenever one standard program could be used to create files for several different jobs in which the type of processing, the blocking factor, etc., were the same, but the data was different. The label would have to vary in each run, to properly identify the file. Also, a disk pack may, at some time be mounted on a drive different from the one originally specified in the DTF entries. The starting and/or ending disk address may also change from that originally specified.

With preplanning and use of the RDLIN (Read Label Information) macro instruction, the information specified for checking or writing standard header labels can be altered at object time without reassembling. Disk drive-number and limit specifications can also be changed at that time. While the user is writing the source program, he determines which input/output files will require a periodic change. He includes a RDLIN instruction in his program ahead of the OPEN instruction for each of these files, and also ahead of any initializing instructions that set word marks in the card read-in area.

The symbolic name of the file, specified in the DTF entry, is entered in the operand field of this RDLIN instruction. Two or more files may be included in one instruction (Figure 33). Any disk-file name, or group of names, in the operand field must always be preceded by the letter *D*. If any tape-file names are included in the same RDLIN instruction, they must be entered in the operand field ahead of the disk-file names. Tape-file names may be preceded by the letter *T*, but this is not required. IOCS assumes that files are in this category if a *D* is not included. Each letter (T or D) and file name, except the last, must be followed by a comma.

Then, when the object program is run, a RDLIN information card must be available in the card reader at the time the RDLIN macro is executed. If two or more files are named in the RDLIN macro instruction, a separate card must be inserted for each file. These cards must be in the same sequence as the file names in the instruction. Each RDLIN information card con-

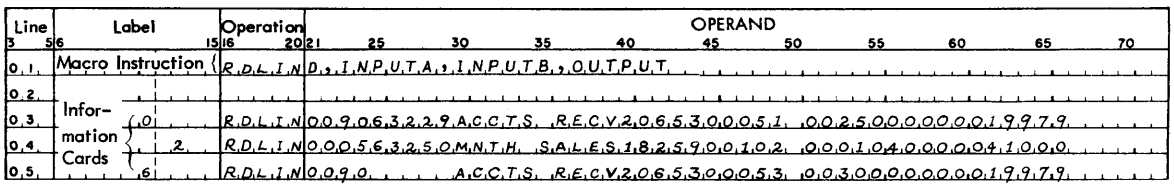



Figure 33. Example of RDLIN Macro Instruction and Information Cards, Disk Records

tains the current information required at this time. The RDLIN macro causes the card to be read and the information to be entered in the DTF area. If a RDLIN information card is not included, a halt occurs. After inserting the card, processing can be resumed.

Whenever a RDLIN information card for a tape or disk file is read, any detail card record in the card input area is destroyed. Therefore, if card records are processed in the input area (not a work area), card data must be processed before a RDLIN macro instruction is issued, or it must be moved (including word marks) to a temporary storage area and saved for later processing. If OVERLAP, RELEASE, or RDROVLP is specified in the DIOCS entries and a RDLIN instruction is issued, a card record *must* be saved and then returned to the card input area after the RDLIN card is processed.

### Disk File

The RDLIN instruction and information cards can be used to change, *prior to opening the disk file*, the DTF specifications for:

- checking or writing a header label (1311 only).
- drive numbers of a labeled or unlabeled file.
- file limits of a labeled or unlabeled file.

They can also be used to give new limits and drive numbers *during the run*, when these must be redefined for an area of a disk file processed in consecutive order. This is required when a logical file on disk:

- Consists of two or more 1311 disk packs, but not all packs can be on-line at the same time. When operation is stopped to mount a pack, the RDLIN card is inserted.
- Consists of two or more modules, or packs on-line, but the addresses are not consecutive. For example, a file of three 1311 disk packs in which the first two have consecutive addresses, but the third does not, requires a RDLIN card before the third pack is processed.

The RDLIN information card for a labeled 1311 disk file *must contain* the drive numbers and the entire header information (except *Label Identifier* 1HDRb), *not just* the fields that are changed. These cards (Figure 33) are punched in *Autocoder* format, with disk drives in columns 11-15, RDLIN in the operation field, and the header information in the operand field in the same sequence as the header label:

| *Columns* | *Information* |
|---|---|
| 11 | Drive on which Pack 0 is mounted |
| 12 | Drive on which Pack 2 is mounted |
| 13 | Drive on which Pack 4 is mounted |
| 14 | Drive on which Pack 6 is mounted |
| 15 | Drive on which Pack 8 is mounted |
| 16-20 | RDLIN |
| 21-24 | File Retention Period |
| 25-29 | Creation Date |
| 30-39 | File Identification |
| 40-44 | File Serial Number |
| 45-49 | Pack Serial Number |
| 50 | Blank |
| 51-54 | File Sequence Number |
| 55-60 | Lower Limit |
| 61-66 | Upper Limit |

Columns 11-15 represent five disk packs, and the punching in these columns represents the five available disk drives. The disk drive control number is punched in the appropriate column. This arrangement of pack and drive numbers corresponds to that used for the DTF NATnPACK entries. For example, if a disk pack with addresses 040000-059999 (pack 4) is mounted on disk drive 2, a "4" is punched in column 13. If that disk pack is mounted on the first drive, however, a "0" is punched in column 13. If a logical file requires two or more packs, the drives on which all packs are mounted must be specified. For example, if addresses 000000-019999 and 020000-039999 are used for data, columns 11 and 12 must be punched with the drive numbers on which those packs are mounted. Any pack columns not required for the file must be blank.

For a 1301 disk file, columns 11-15, 55-60, and 61-66 of the RDLIN card may be used. Similar to a 1311, columns 11-15 represent five address ranges, and the punching in these columns represents the five available disk modules to which the addresses may be assigned. Columns 55-66 are used for lower and upper file limits.

| *Columns* | *Information* |
|---|---|
| 11 | Module to which addresses 000000-199999 are assigned |
| 12 | Module to which addresses 200000-399999 are assigned |
| 13 | Module to which addresses 400000-599999 are assigned |
| 14 | Module to which addresses 600000-799999 are assigned |
| 15 | Module to which addresses 800000-999999 are assigned |
| 55-60 | Lower Limit |
| 61-66 | Upper Limit |

The numerical portion of the disk module control character is punched in the appropriate column (11-15). This corresponds to the arrangement used for the NATnPACK entries.

RDLIN cards for a 1301 file, or an unlabeled 1311 file, processed in consecutive or control-sequential order must contain both the drive numbers and the file limits, regardless of which specification is changed. RDLIN cards for an unlabeled file processed in random order require only the appropriate drive numbers. The upper and lower limits need not be included in the RDLIN card for a random input file (labeled or unlabeled).

#### Tape File

The RDLIN information cards for a tape file must contain the entire header information (except *Label Identifier* and *Tape Serial Number*), not just the fields that are changed. These cards are punched in *Autocoder* format, with RDLIN in the operation field and the header information in the operand field in the same sequence as the header label.

The header information must correspond to the fields of the particular standard header label that is used for the file. The format of the RDLIN card associated with each of the three types of standard header labels is given in the following lists.

RDLIN card for Type-A standard header label, when fields 9-19 are *not* processed:

| *Columns* | *Header Information* | *Header-Label Field Number* |
|---|---|---|
| 16-20 | RDLIN | |
| 21-24 | Retention Period | 2 |
| 25-29 | Creation Date | 3 |
| 30-39 | File Identification | 4 |
| 40-44 | File Serial Number | 5 |
| 45-48 | Reel Sequence Number | 7 |
| 80 | "A", if DIOCS LABELDEF specified label size "D" | |

RDLIN card for Type-A standard header label, when fields 9-19 are processed:

| *Columns* | *Header Information* | *Header-Label Field Number* |
|---|---|---|
| 16-20 | RDLIN | |
| 21-24 | Retention Period | 2 |
| 25-29 | Creation Date | 3 |
| 30-39 | File Identification | 4 |
| 40-44 | File Serial Number | 5 |
| 45-48 | Reel Sequence Number | 7 |
| 49-52 | blank | 8 |
| 53 | Density | 9 |
| 54 | Checksum | 10 |
| 55 | Block Sequence | 11 |
| 56 | Tape Checking/Interpreting | 12 |
| 57 | Tape Data Recording Technique | 13 |
| 58 | Tape Data Processing Technique | 14 |
| 59-62 | Creating System | 15 |
| 63 | Record Format | 16 |
| 64-68 | Record Length | 17 |
| 69-73 | Blocking Factor/Size | 18 |
| 74 | Checkpoint | 19 |
| 80 | "A", if DIOCS LABELDEF specifies label size "D" | |

RDLIN card for Type-B standard header label:

| *Columns* | *Header Information* | *Header-Label Field Number* |
|---|---|---|
| 16-20 | RDLIN | |
| 21-25 | File Serial Number | 3 |
| 26 | − (minus sign) | 4 |
| 27-29 | Reel Sequence Number | 4 |
| 30 | blank | 4 |
| 31-40 | File Identification | 5 |
| 41-45 | Creation Date | 6 |
| 46 | − (minus sign) | 7 |
| 47-49 | Retention Period | 7 |
| 50 | blank | 7 |
| 80 | "B", if DIOCS LABELDEF specifies label size "D" | |

RDLIN card for Type-C standard header label:

| *Columns* | *Header Information* | *Header-Label Field Number* |
|---|---|---|
| 16-20 | RDLIN | |
| 21 | blank | 3 |
| 22-26 | File Serial Number | 3 |
| 27 | blank | 4 |
| 28-31 | Reel Sequence Number | 4 |
| 32 | blank | 4 |
| 33-34 | Creation Date: Year | 5 |
| 35 | blank | 5 |
| 36-38 | Creation Date: Day | 5 |
| 39-41 | blank | 6 |
| 42-44 | Retention Period | 6 |
| 45-50 | Label Information:<br>blank<br>Density<br>Character Coding<br>Checksum<br>Block Sequence<br>Checkpoint Record | 7 |
| 51-68 | File Identification | 8 |
| 80 | "C", if DIOCS LABELDEF specifies label size "D" | |

### FEORL Macro — Tape Records

The FEORL (Forced End-of-Reel) instruction is used for *output* tape whenever the user wishes to perform the end-of-reel operations at a time other than when the reflective spot is sensed. For example, it is used in conjunction with a programmer's Exit 8 routine for output tapes. That is, when the programmer plans to use Exit 8 to write records beyond the reflective spot in an end-of-reel condition (not end-of-file), he must *cause* the end-of-reel operations after all his data is written. The FEORL instruction writes the last block of records if necessary (partially filled with data) and

writes a tape mark. Then it processes the trailer label, provides for a reel change, and processes the header label, as specified in the DTF entries.

This instruction is used for *input* tape if the programmer wants to discontinue reading data from one reel and switch to another before the tape mark signals the end of data on the first reel. For input tape, FEORL immediately rewinds, or rewinds and unloads the tape, provides for a reel change, and processes the header label as specified in the DTF entries.

The symbolic name of the file, specified in the DTF entry, is entered in the operand field of the FEORL instruction (Figure 34).

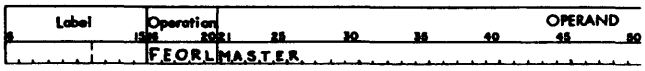



Figure 34. FEORL Macro Instruction, Tape Records

## DCLOS Macro—Tape Records

When an input tape is read, any block of records that contains a parity error can be "dumped" onto another tape for later investigation, if specified in the DIOCS entry (READERROR). This tape is activated by the *TAPE,n* specification in the READERROR entry, but it must be de-activated by using the DCLOS (Dump-Close) macro instruction (Figure 35). This instruction causes a tape mark to be written after the last record. The tape is rewound if REWIND is specified in the operand field of this instruction. It is rewound and unloaded if REWIND,UNLOAD is specified.

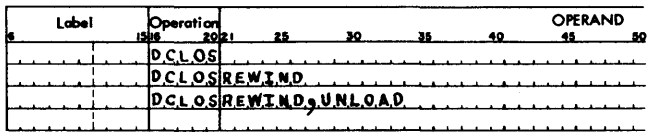



Figure 35. DCLOS Macro Instruction, Tape Records

## DIOCS Entry

The DIOCS Entry (Descriptive IOCS) specifies generally the characteristics of all the files to be processed and the machine configuration on which the program will be run. During assembly, the DIOCS Entry cards must follow the AUTOCODER RUN, JOB, and CTL cards and precede the user's source program.

The entry (Figure 36) consists of a DIOCS header card followed by a series of detail cards. The header card contains only DIOCS, in the operation field; the label and operand fields are blank.

The succeeding detail cards may follow in any order and must be blank in the operation field. Their label and operand fields are punched with the specifications for the program. When multiple operands are required for a particular label, they may appear in any order and must be separated by commas. Any detail card may have comments in the operand field, provided two blanks separate the comments from the last operand. When a particular detail entry does not apply to the job, that card must be omitted.

The DIOCS entries are grouped in the following paragraphs by the type of file to which they apply:

1. All Files
2. Disk and Tape Files
3. Disk Files
4. Tape Files

| LABEL | OPERATION | POSSIBLE OPERANDS# | MAX.CHAR* | APPLIES TO | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | DISK | TAPE | READER | PUNCH | PRINTER |
| | DIOCS† | | | X | X | X | X | X |
| DIOCSORG | | Any actual address | 5 | X | X | X | X | X |
| IODEVICES† | | READER | - | | | X | | |
| | | PUNCH | - | | | | X | |
| | | DISK | - | X | | | | |
| | | TAPE | - | | X | | | |
| | | PRINTER | - | | | | | X |
| FEATURES | | DIRECT v | - | X | | | | |
| | | SCAN | - | X | | | | |
| | | RELEASE | - | | | X | X | |
| | | OVERLAP | - | | X | X | X | X |
| RDLIN | | DISK | - | X | | | | |
| | | TAPE | - | | X | | | |
| DLABDEF v | | CHECK or IDENT | - | X | | | | |
| | | OUTPUT | - | X | | | | |
| OVLAYLBL v | | YES | - | X | | | | |
| DISKDRIVES | | Any disk drive control characters 0, 2, 4, 6, 8, +0, B, D, F, H | 10 | X | | | | |
| DUPPACKS | | YES | - | X | | | | |
| PROCESTYPE | | RANDOM | - | X | | | | |
| | | CONSEC | - | X | | | | |
| | | CTLSEQ | - | X | | | | |
| CYLDOVFLW | | YES | - | X | | | | |
| WRITEXIT | | Label of user's routine | 6 | X | | | | |
| RDROVLP | | File Name | 6 | X | | X | | |
| PCHOVLP | | File Name | 6 | X | | | X | |
| PRNTOVLP | | File Name | 6 | X | | | | X |
| SEEKCHECK v | | YES | - | X | | | | |
| RECTYPE | | UNBLOCKED | - | X | | | | |
| CSLINKAGE | | Any lengths 6, 7, 8 | 3 | X | | | | |
| TAPEUSE | | INPUT or OUTPUT | - | | X | | | |
| ALTDRIVE | | YES | - | | X | | | |
| LABELDEF | | STANDARD or NONSTANDARD or MIXED | - | | X | | | |
| | | A or B or C or D | - | | X | | | |
| EXITS | | YES | - | | X | | | |
| WLRCHECK | | YES | - | | X | | | |
| READERROR | | TAPE,n or Label of user's routine | 6 | | X | | | |
| | | SCAN | - | | X | | | |
| CHECKPOINT | | YES | - | | X | | | |

† Must be included. Other cards are included when applicable.

# Where two or more lines of operands are shown for a label, one item from each line may be entered.

* Maximum number of characters allowed by IOCS for the operand. The 6-position operands allow for a maximum-size label of 6 positions, or a smaller label with address adjustment and/or indexing.

v Applies to IBM 1311 only.

Figure 36. Specifications for DIOCS Entry Cards

## All Files

### DIOCSORG

This card indicates the location in core storage at which the programmer wants the *Autocoder* processor to begin generating the IOCS subroutines. Any actual address that is valid in an *Autocoder* ORG statement may be entered in the operand field. If this card is omitted, the IOCS subroutines start at storage location 334.

### IODevices

This card must be included, and it specifies all the input and output units used in the program. It summarizes the DTF FILETYPE entries. The operand field contains one or more of these:

READER, if any input data is in cards.
PUNCH, if any output data is punched in cards.
DISK, if any input or output records are on disk.
TAPE, if any input or output records are on magnetic tape.
PRINTER, if any printer operation occurs.

### Features

This card indicates any special features affecting input/output operations that are utilized by the program. The operand field contains:

DIRECT, if the Direct Seek feature is installed (1311 only). IOCS uses this feature to reduce access time on a disk seek operation, by allowing an access mechanism to be repositioned at a new setting without first returning to the *home* position.
SCAN, if the Scan Disk feature is installed and the user includes a SCAN macro instruction in his program to search for a record(s) containing specified data. This can be used only in the sector mode of operation.
RELEASE, if either the Read Release or Punch Release feature is used. This cannot be included with *OVERLAP*.
OVERLAP, if the Processing Overlap feature is used. This cannot be included with *RELEASE*.

## Disk and Tape Files

### Rdlin

If a RDLIN macro instruction is included in the user's program (to change standard-header-label, disk-drive, or file-limit specifications), this card must be included. The operand field contains one or both of these:

TAPE, whenever a RDLIN instruction is used for a tape file.
DISK, whenever a RDLIN instruction is used for a disk file.

## Disk Files

### DLabdef (1311 Only)

Whenever header labels on one or more 1311 disk packs are to be checked or written, this card must be included. It summarizes the specifications in the DTF CHECKLABEL entries for *input* files. One, but not both, of these operands is entered:

CHECK, whenever labels for *one or more* logical files are to be completely checked.
IDENT, when labels for one or more files are to be partially checked (*file identification*) and *none* of the labels are to be completely checked.

If header labels are to be written on one or more *output* files (random or consecutive), this card must be included with *OUTPUT* in the operand field.

### OvlaylbI (1311 Only)

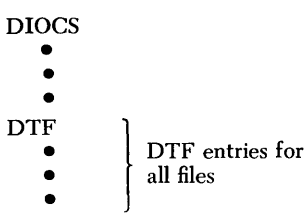
This entry may be included to reduce the amount of core storage used for a program, by causing IOCS to overlay the disk *open* routines with the user's program. This is advantageous when 1311 disk files are labeled, requiring several routines for checking or writing the labels when the files are opened. *YES* is entered in the operand field.

Whenever this OVLAYLBL entry is used, each OPEN instruction for a *disk* file must be followed immediately by an execute (EX) statement, and then by an origin (ORG) statement. Both the EX and ORG statements must have the same operand. With this entry the user can also overlay any portion of his own program associated with the OPEN, such as initializing and/or RDLIN instructions.

When this entry is used, all OPEN instructions for *disk* files must be issued before the *Autocoder* loader is cleared. The *first* disk-file OPEN instruction must be issued before any *Autocoder* LTORG, EX, or XFR statement is issued.

If more than nine files are to be opened, a second OPEN instruction should be issued immediately after the ORG statement associated with the first OPEN. This is necessary because the IOCS disk label-handling routines are located in core storage immediately following the first disk OPEN macro whenever OVLAYLBL is specified. Also, the first OPEN instruction *must* specify the larger number of files, so that the label-handling routines will not be destroyed by the second OPEN macro.

A program might contain these instructions:



```
START   SW          ┐
        CS          │
          •         │ Initializing entries
          •         ┘
        RDLIN       D,FILEA,FILEB,FILEC,. . .,FILE I
        OPEN        D,FILEA,FILEB,FILEC,. . .,FILE I
        EX          START
        ORG         START
        OPEN        D,FILEJ,FILEK
        EX          START
        ORG         START
        MCW         ┐
          •         │
          •         │ User's detail to
          •         │ process file data
          •         ┘
```

The OPEN instruction may be issued at any time in the program for card, tape, and printer files.

If the Read or Punch Release special feature is specified by *RELEASE* in the DIOCS FEATURES entry, the card input file must be opened *after* the EX and ORG statements associated with the OPEN instruction for the disk file.

**Diskdrives**

This card must be included for any program that uses disk files. A control character for each disk module or drive used by the program is entered in the operand. Because a maximum of five 1301 modules and five 1311 drives can be used in a system, up to ten characters may be entered in this card. They may be entered in any order, and each character (except the last) must be followed by a comma. The control characters for the 1301 are alphabetic. For the 1311, they are numerical. The disk modules and drives, and the associated control numbers are:

| *Disk Module/Drive* | *1301 Character* | *1311 Character* |
|---|---|---|
| 0 | +0 | 0 |
| 1 | B | 2 |
| 2 | D | 4 |
| 3 | F | 6 |
| 4 | H | 8 |

If the disk-drive assignment will be changed later by a RDLIN card, all drives in the system should be specified in this DISKDRIVES entry.

**Duppacks**

This card must be included with *YES* in the operand field, if a program requires:

- Two or more 1301 disk modules that have the same address range (for example, two modules with addresses 000000-199999).
- Two or more 1311 disk packs that have the same address range (for example, two disk drives loaded with packs each having addresses 000000-019999).

This entry is *not* required if a program uses a 1301 disk module and a 1311 disk pack that have the same address range (for example, 1301 addresses 000000-199999 and 1311 addresses 000000-019999).

**Procestype**

This card summarizes the types of processing specified in the DTF FILETYPE entries for disk records:

RANDOM, if the records in any logical file are processed in random order.

CONSEC, if the records in any logical file are processed in consecutive order.

CTLSEQ, if the records in any logical file are processed in control-sequential order.

**Cyldovflw**

This card must be included, with *YES* in the operand field, if any logical disk file is set up so that a read or write operation requires an *overflow* from one cylinder to another. For example, if 70-character records are blocked four to a block, three sectors are use for each block. If the file is recorded in order on 1311 disks and consists of more than 264 records (66 blocks), at least two cylinders are required to store the file (200 sectors per cylinder and 4 records per 3 sectors). The 67th block (records 265, 266, 267, and 268) is stored in the last two sectors of one cylinder and the first sector of the next cylinder. Therefore, when a read or write operation is performed for this block, the records must be read from two different cylinders causing an overflow condition.

Similarly, if the file of 70-character records is recorded in order on 1301 disks and consists of more than 1064 records (266 blocks), at least two cylinders are required (800 sectors per cylinder). The 267th block (records 1065, 1066, 1067, and 1068) is stored in the last two sectors of one cylinder and the first sector of the next cylinder. Reading or writing this block causes an overflow.

If blocked records are stored in order and each block requires only two sectors, however, an overflow would *not* occur during a read or write operation. The 100th block in the 1311 (400th in the 1301) would end in the last sector of one cylinder, and the 101st (or 401st) block would start at the first sector of the next cylinder.

**Writexit**

When disk records are processed in the sector mode of operation, this entry allows the programmer to process data during the rotational delay between write and write disk check operations. The length of this delay time depends on the number of sectors written. To compute the milliseconds available for processing, use the proper formula:

| *Sectors Written* | *1301* | *1311* |
|---|---|---|
| 18 or less | 31 − 1.67N | 37 − 2N |
| 19 or more | 34.3 − 1.67N + 30R | 41 − 2N + 36R |

In each formula, N equals the number of sectors. R is the quotient of N (number of sectors) divided by 18, using *only* the whole number and dropping any remainder. If 20 sectors in a 1311 are written, for example, N/18 = 1, and 37 ms are available.

The symbolic address of the user's routine is entered in the operand, and a branch from IOCS to this routine occurs whenever a disk write is completed. This branch to the user's routine occurs two or more times for the same block of records if the write disk check operation detects an error and causes re-writing of the same records. Therefore, if the routine is to be processed only once, the user must provide the necessary programming to prevent the extra processing. In the user's routine, *only GET* and/or PUT instructions for a card reader, card punch, or printer file may be issued. No other IOCS macro instructions may be used. At the end of user's routine, program control must be returned to IOCS by branching to IOCWDC (IOCS Write Disk Check).

If the programmer uses any index registers in his routine, he must save the contents of those registers at the beginning of his routine and restore them before branching to IOCWDC. This is necessary because IOCS requires that all index registers have the same contents after the user's routine that they had before.

**Rdrovlp**

Cards in the IBM 1402 Card Read-Punch may be read during disk-seek operations if this overlap card is included. The name of the reader file specified in the DTF entry is punched in the operand field of this card.

Whenever this overlap of card-reader and disk-seek operations is planned, a work area must be assigned in the DTF entry (WORKAREA) or in the GET instruction for the reader file. When a disk-seek operation is initiated, a card record is read to the input area, from which IOCS moves it to the work area on the next GET instruction for the reader file. The programmer processes all card records in the work area. If two or more disk seeks occur without an intervening GET for the reader, only the first seek causes card reading. Conversely, if two or more GET instructions are issued for the card reader without an intervening seek, each GET after the first reads a card to the input area and moves the record directly to the work area. This entry must not be included if either *OVERLAP* or *RELEASE* is specified in the DIOCS FEATURES entry.

**Pchovlp**

Cards in the IBM 1402 Card Read-Punch may be punched during disk-seek operations if this overlap card is included. The name of the punch file specified in the DTF entry is punched in the operand field of this card.

Whenever this overlap of card-punch and disk-seek operations is planned, a work area must be assigned in the DTF entry (WORKAREA) or in the PUT instruction for the punch file. The programmer builds his output record in the work area. When he issues a PUT instruction for the punch, IOCS moves the data from the work area to the output area where it waits until a disk-seek operation is initiated. At that time the card is punched. If two or more card records are PUT without an intervening seek, the data in the output area is punched on each PUT, before the next card-record is moved from the work area to the output area. Conversely, if two or more disk seeks occur without an intervening PUT for the punch, only the first seek causes card punching. If data is waiting at the end of a job, the CLOSE macro instruction for the punch file punches the last card. This entry must not be included if either *OVERLAP* or *RELEASE* is specified in the DIOCS FEATURES entry.

**Prntovlp**

Lines of data may be printed during disk-seek operations if this overlap card is included. The name of the printer file specified in the DTF entry is punched in the operand field.

Whenever this overlap of printer and disk-seek operations is planned, a work area must be assigned in the DTF entry (WORKAREA) or in the PUT instruction for the printer file. The programmer builds his output record in the work area. When he issues a PUT instruction for the printer, IOCS moves the data from the work area to the output area where it waits until a disk-seek operation is initiated. At that time the line is printed. If two or more lines of data are PUT without an intervening seek, the data in the output area is written on each PUT, before the next line is moved from the work area to the output area. Conversely, if two or more disk seeks occur without an intervening PUT for the printer, only the first seek writes a line. If data is waiting at the end of a job the CLOSE macro instruction for the printer file prints the last line.

**Seekcheck (1311 Only)**

If the Direct Seek special feature is not specified, this card may be included, with *YES* in the operand field, to reduce access time on a 1311. IOCS eliminates the seek operation whenever the access mechanism is positioned on the proper cylinder for the next read or write operation. This entry must not be used if *DIRECT* is specified in the DIOCS FEATURES entry.

**Rectype**

If *all* disk files in a program contain unblocked records, this card is included with *UNBLOCKED* in the operand field.

#### CSLinkage

This card must be included if any disk file in the program is processed in control-sequential order. It specifies the length of the linkage fields in the records, plus a record mark if any. Because the linkage field gives the *address* of the next sequential record, it is a 6-digit field for unblocked records or a 7-digit field for blocked records. The 7th digit for blocked records specifies the number of the record within the block. Furthermore, because this is the last field in the record, it may contain a record mark. If it does, that must be included in the number specified in this CSLINKAGE card. Thus, a digit 6, 7, or 8 is entered in the operand field:

| | |
|---|---|
| Linkage field for unblocked records, no record mark | 6 |
| Linkage field for unblocked records plus record mark | 7 |
| Linkage field for blocked records, no record mark | 7 |
| Linkage field for blocked records plus record mark | 8 |

If the linkage fields in two or more files differ in length, this card must specify all the lengths used. Thus, it may contain one, two, or three operands.

## Tape Files

#### Tapeuse

This card summarizes the types of tape files specified in the DTF FILETYPE entries. One, but not both, of these operands is entered:

INPUT, if *all* tape files are used for input.

OUTPUT, if *all* tape files are used for output.

If both input and output tape files are used in the program, this card is omitted.

#### Altdrive

If an alternate tape drive is to be used for any multi-reel tape file, as specified in a DTF ALTTAPE entry, this card is included. The word *YES* is entered in the operand.

#### Labeldef

Whenever one or more tape files with header and/or trailer labels are processed, this card summarizes the specifications in all the DTF TYPELABEL entries.

For type of label, *one* of these three types is entered in the operand:

STANDARD, when *all* tape files have standard header and trailer labels (Type A, B, or C).

NONSTANDARD, when *all* tape files have nonstandard header and trailer labels.

MIXED, when some tape files have standard labels and some have nonstandard labels or are not labeled.

Whenever *STANDARD* or *MIXED* is specified, the size of the standard labels must also be indicated by entering *one* of these specifications in the operand:

A, if *all* standard labels contain 120 characters.

B, if *all* standard labels contain 80 characters.

C, if *all* standard labels contain 84 characters.

D, if two, or three, different-size standard labels are used (120, 80, and/or 84).

#### Exits

If any exit from the IOCS routines is used for any tape header or trailer label operation, this card is included with *YES* in the operand field.

#### WLRCheck

If IOCS is to check for any wrong-length records, as specified in a DTF WLRADDR entry, this card must be included with *YES* in the operand field.

#### Readerror

If a parity error is detected in an input tape record larger than 12 characters (12 or less considered a noise record), the tape is automatically backspaced and reread 99 times, before the block is considered an error block. After this, an error block is automatically passed without processing and the next block is read in and processing continues, if the READERROR entry is not included. By including this READERROR entry, the programmer can specify other procedures to be followed for an error block. Either *TAPE,n* or the symbolic address of a user's read-error routine, but not both, can be entered in the operand. With either entry, *SCAN* can also be specified. The functions of these three operands are:

TAPE,n – to transfer an error block to another tape for later investigation. The letter "n" represents the number of the tape drive (1-6) used for this purpose.

Label of read-error routine – to print or punch the error block. In his routine the programmer can print the error block on the IBM 1443 Printer or the IBM 1447 Console Inquiry Printer, or he can punch it in cards in the IBM 1442 Card Read-Punch or the IBM 1444 Card Punch. The address of the high-order position of the tape input area that contains the error block is available at IOCSCN-1. The number of the tape unit from which the error block was read is available at IOCSCN-4. These address fields do not contain word marks.

The programmer must *not* issue any IOCS macro instructions in his read-error routine. At the end of

Figure 37. Specifications for DTF Entry Cards (Part 1 of 3)

| NUMBER | LABEL | OPERATION | POSSIBLE OPERANDS# | MAX. CHAR* | APPLIES TO: DISK: RANDOM PROCESSING | DISK: CONSECUTIVE PROCESSING | DISK: CTRL-SEQUENT. PROCESSING | TAPE: INPUT | TAPE: OUTPUT | READER | PUNCH | PRINTER | MUST BE INCLUDED | REMARKS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | DTF† | Any File Name | 6 | x | x | x | x | x | x | x | x | Each file | |
| 1 | FILETYPE† | | READER or PUNCH or DISK,INPUT or DISK,OUTPUT or TAPE,INPUT or TAPE,INPUT,CHECKPOINT or TAPE,OUTPUT or PRINTER | — | x | x | x | x | x | x | x | x | Each file | FILETYPE must be the first card after the DTF header card. INPUT may be omitted for disk input file. |
| | | | 1301 | — | x | x | x | | | | | | IBM 1301 used for disk records | |
| | | | RANDOM or CONSEC or CTLSEQ or CTLSEQ,RM | — | x | x | x | | | | | | Consecutive or control-sequential processing | May be omitted for random processing. |
| 2 | IOAREAS† | | Label of input or output area | 6 | x | x | x | x | x | | | | Tape and disk files | Same as label of DA statement. |
| 3 | INDEXREG | | X1 or X2 or X3 | 2 | x | x | x | x | x | | | | Index Register assigned to an input or output area | Omit if DTF WORKAREA included, or if work areas specified in GET or PUT. |
| 4 | WORKAREA | | Label of work area | 6 | x | x | x | x | x | x | x | x | For card and printer files, see text | Same as label of DA statement. Omit if DTF INDEXREG included, if work areas specified in GET or PUT, or if RANDOM OUTPUT specified. |
| 5 | RECFORM | | BLOCKED or UNBLOCKED | — | | x | | x | x | | | | Blocked records | |
| | | | VARIABLE or FIXED | — | | x | | x | x | | | | Variable-length records | |
| 6 | BLOCKSIZE | | Size of one input or output storage area | 4 | | | | x | x | | | | Blocked records | Omit group-mark with word-mark and record-length–checking positions. |
| 7 | NSECTORS | | Any sector control number | 3 | x | x | x | | | | | | Sector mode of operation | Number of sectors required for a disk record or for a block of disk records. |
| 8 | TRACKFORM | | SECTOR or TRKSECTOR or TRKRECORD or TRKRECORD,ADDRESS | — | x | x | x | | | | | | Other than sector mode of operation | |
| 9 | SIZEREC | | Length of record (fixed-length) or Low-order position of record-length field (variable-length) | 3 | x | x | x | x | x | | | | Blocked records<br>Fixed-length unblocked records | Maximum number of characters: 3 for tape records or blocked disk records, 4 for unblocked disk records. |
| 10 | NAT0PACK | | Any disk drive control number 0, 2, 4, 6, 8 | 1 | x | x | x | | | | | | Disk files | |
| | NAT2PACK | | | 1 | x | x | x | | | | | | | |
| | NAT4PACK | | | 1 | x | x | x | | | | | | | |
| | NAT6PACK | | | 1 | x | x | x | | | | | | | |
| | NAT8PACK | | | 1 | x | x | x | | | | | | | |
| 11 | CHANDRIVE | | Any tape drive number 1-6 | 1 | | | | x | x | | | | Tape records | |
| 12 | ALTTAPE | | Any tape drive number 1-6 | 1 | | | | x | x | | | | Alternate tape drive | |
| 13 | EOFADDR | | Label of user's end-of-file routine | 6 | | x | x | x | | x | | | Card or tape input file<br>Disk records with consecutive or control-sequential processing | For tape: routine entered on end-of-file only, not on end-of-reel. |
| 14 | NRECORDS | | Any number 0-99 | 2. | x | x | x | | | | | | Fixed-length blocked records | 0-9 for random, 0-29 for control-sequential. Number = blocking factor minus one. |
| 15 | UPDATE | | YES | — | x | | x | | | | | | If PUT is used for a disk input file | |
| 16 | FILESTART | | Any sector address | 6 | x | x | x | | | | | | Consecutive or control-sequential processing<br>Random processing of labeled output file | Address of first record, or block of records. |
| 17 | FILEND | | Any sector address | 6 | x | x | x | | | | | | Consecutive or control-sequential processing<br>Random processing of labeled output file | Address of last record, or block of records, in the file or first file area |

| NUMBER | LABEL | OPERATION | POSSIBLE OPERANDS# | MAX. CHAR.* | APPLIES TO: DISK: RANDOM PROCESSING | DISK: CONSECUTIVE PROCESSING | DISK: CTRL-SEQUENT. PROCESSING | TAPE: INPUT | TAPE: OUTPUT | READER | PUNCH | PRINTER | MUST BE INCLUDED | REMARKS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | SEEKEOC✓ | | YES | — | | x | x | | | | | | | Use only when one logical file is stored in a disk pack. |
| 19 | MODEPAR | | LOAD or MOVE | — | | | | x | x | | | | LOAD mode | May be omitted for MOVE mode. |
| 20 | OUTLIN | | YES | — | | | | x | x | | | | | Applies only to blocked records. |
| 21 | TYPELABEL | | STANDARD or NONSTANDARD | — | | | | x | x | | | | Labeled tape files | TM — to pass tape mark after Type B or C header label. |
| | | | A or B or C | — | | | | x | x | | | | Standard labels | |
| | | | TM | — | | | | x | | | | | | |
| 22 | CHECKLABEL✓ | | ALL or IDENT | — | x | x | x | x | | | | | Check standard labels | Omit for nonstandard labels. |
| 23 | FILSEQ✓ | | Any 4-digit number | 4 | x | x | x | | | | | | First-pack sequence number other than 0001 | Applies only to labeled files. Omit if first-pack number is 0001. |
| 24 | SERIALNUM✓ | | Any 5-digit number | 5 | x | x | x | x | x | | | | Automatic check on input File No. differs from Pack or Tape No. on output | Applies to File Serial Number only in standard labels. |
| 25 | PACKSERIAL✓ | | Any 5-digit number | 5 | x | x | x | | | | | | Check or write header label | Applies only to first pack in a labeled file. |
| 26 | REELSEQ | | Any 3- or 4-digit number | ** | | | | x | x | | | | First-reel sequence number other than 001 (Type B label) or 0001 (Type A or C label) | Applies only to standard labels. Omit if first-reel number is 001 or 0001. |
| 27 | HEADER✓ | | File Identification | ** | x | x | x | x | x | | | | Check or write standard label | Operands must be entered in sequence shown. |
| | | | Creation Date | 5 | x | x | x | x | | | | | | Omit Creation Date for output file. |
| | | | Retention Period | ** | x | x | x | x | x | | | | | Include Creation Date and Retention Period for input file, only if ALL specified in DTF CHECKLABEL. |
| 28 | TOTALS | | RECORD | — | | | | x | x | | | | Record total | Applies only to labeled tape files. |
| | | | Low-order position of hash-total field | 3 | | | | x | x | | | | Hash total | Block count is taken automatically. |
| 29 | EX1ADDR | | Label of user's routine | 6 | | | | | x | | | | Modify standard trailer | |
| | EX2ADDR | | | 6 | | | | | x | | | | Build and write additional/nonstandard trailers | |
| | EX3ADDR | | | 6 | | | | | x | | | | User checks old standard header | Eliminates IOCS-checking of retention period. |
| | EX4ADDR | | | 6 | | | | | x | | | | Modify standard header | |
| | EX5ADDR | | | 6 | | | | | x | | | | Build and write additional/nonstandard headers | Also read and check old nonstandard headers. |
| | EX6ADDR | | | 6 | | | | x | | | | | Additional check of standard trailer Read and check additional/nonstandard trailers | May also use to modify reel-sequence count. |
| | EX7ADDR | | | 6 | | | | x | | | | | Additional check of standard header Read and check additional/nonstandard headers | |
| | EX8ADDR | | | 6 | | | | | x | | | | User's routine before tape mark is written | Occurs at reflective spot on tape. |
| 30 | PADDING | | Any character except * ≠ ≠ √‾ ∨ @ | 1 | | | | | x | | | | Pad with character other than blank | Applies only to blocked fixed-length tape records. |
| 31 | REWIND | | UNLOAD or NORWD | — | | | | x | x | | | | Unload at CLOSE or end of reel Prevent rewinding | Omit if rewind at OPEN, CLOSE, and end of reel. |
| 32 | WLRADDR | | Label of user's routine | 6 | | | | x | | | | | Check tape-record length | Omit for unblocked variable-length records. |
| 33 | CARDPOC | | Any Card Pocket number 1, 2, 4, or 8 | 1 | | | | | | x | x | | Selection of card input file with Read Release special feature | Omit if selection specified in GET or PUT. With overlap, cord input may not be selected. |
| 34 | FORMCNTL | | Any 1401/1460 Space or Skip d-character | 1 | | | | | | | | x | | Omit if specified in PUT or SPACE/SKIP. |
| 35 | OVERFLOW | | 9 or 12 or 9,Label of user's routine or 12,Label of user's routine | 6 | | | | | | | | x | Carriage overflow | If operand 9 or 12 is used, carriage restores to channel 1 and processing continues. |

Figure 37. Specifications for DTF Entry Cards (Part 2 of 3)

Figure 37. Specifications for DTF Entry Cards (Part 3 of 3)

| NUMBER | LABEL | OPERATION | POSSIBLE OPERANDS# | MAX. CHAR.* | APPLIES TO: DISK: RANDOM PROCESSING | DISK: CONSECUTIVE PROCESSING | DISK: CTRL-SEQUENT. PROCESSING | TAPE: INPUT | TAPE: OUTPUT | READER | PUNCH | PRINTER | MUST BE INCLUDED | REMARKS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 36 | LABINFO | | Density Code 0 or 1<br>Character Code 0 or 1<br>Checksum Code 0 or 1<br>Block Sequence Code 0 or 1<br>Checkpoint Record Code 0 or 1 | 5 | | | | x | x | | | | Check or write Label Information field of Type C header label<br>All 5 operands must be entered in sequence shown | Applies only to Type C standard header label. Must be omitted for Type A or B. |
| 37 | DENSITY | | Code 0, 1, or 2 | 1 | | | | x | x | | | | Check or write code 1 or 2 in field 9 of Type A header label | Apply only to Type A standard header label. Must be omitted for Type B or C. |
| 38 | CHKSUM | | Any code 0-9 | 1 | | | | x | x | | | | Check or write code 1-9 in field 10 of Type A header label | |
| 39 | BLKSEQIND | | Any code 0-9 | 1 | | | | x | x | | | | Check or write code 1-9 in field 11 of Type A header label | If any entry 35-45 is included, all fields 9-19 in the Type A standard header label are checked or written. If no entry 35-45 is included, all fields 9-19 are treated as blank. |
| 40 | TPCHKIND | | Code 1, 2, or 3 | 1 | | | | x | x | | | | Check or write code 1 or 2 in field 12 of Type A header label | |
| 41 | TPRECTCH | | Digit 6 | 1 | | | | x | x | | | | | |
| 42 | TPPROTCH | | Any digit 0-9 | 1 | | | | x | x | | | | | |
| 43 | CREATSYS | | Data processing system number | 4 | | | | x | x | | | | Check a systems number other than 1401 in field 15 of Type A header label | |
| 44 | FMRECORD | | A or F or W or X | 1 | | | | x | x | | | | Check or write code A, F, W, or X in field 16 of Type A header label | |
| 45 | LNGRECORD | | Number of characters per record (fixed-length), or<br>Number of characters in longest record (variable-length) | 5 | | | | x | x | | | | Check or write length of record in field 17 of Type A header label | |
| 46 | BLKFACTOR | | Number of records per block (fixed-length), or<br>Number of characters in largest block (variable-length) | 5 | | | | x | x | | | | Check or write blocking factor in field 18 of Type A header label | |
| 47 | CHPTRCD | | Any code 0-9 | 1 | | | | x | x | | | | Check or write code 1-9 in field 19 of Type A header label | |

† Must be included. Other cards are included when applicable.

# Where two or more lines of operands are shown for a label, one item from each line may be entered.

* Maximum number of characters allowed by IOCS for the operand. The 6-position operands in the detail entries allow for a maximum-size label of 6 positions, or a smaller label with address adjustment and/or indexing.

√ Does not apply to IBM 1301.

** Maximum number of characters varies with the type of label:

| | Tape | | | Disk |
|---|---|---|---|---|
| | A | B | C | |
| REELSEQ | 4 | 3 | 4 | — |
| File Identification | 10 | 10 | 18 | 10 |
| Retention Period | 4 | 3 | 3 | 4 |

his routine, he must return to IOCS by branching to IOCREX. If *SCAN* is also specified in the READERROR entry, the user's routine is executed first. Then the scan procedure must be performed.

*SCAN* – to halt the program and allow the operator to investigate the error. The error stop switch on the tape adapter unit must be OFF for this operation. When processing stops, the operator must follow this procedure:

1. Set the tape select switch on the 1401 to "D" (Diagnostic), or turn ON the DIAGNOSTIC switch on the 1447 (1460 system).
2. Turn OFF the check stop switch.
3. Press the start key to reread the error block. This stores the characters as they are written on tape, without internally correcting any parity error. After the block is stored, the program halts again, and STORAGE ADDRESS displays the address of the next sequential instruction.
4. If a parity error was detected on the reread (step 3), the PROCESS and TAPE lights on the 1401 console are ON. Perform a storage scan operation to locate the character that contains the parity error, and correct it if possible.
5. Reset the tape select switch to "N" (1401), or turn OFF the DIAGNOSTIC switch (1447).
6. Turn the check stop switch ON.
7. Press CHECK RESET to turn OFF the PROCESS light, if a parity error occurred on the reread (step 4).
8. Press START RESET and START to process a corrected block, or press START to bypass an incorrect block and resume processing. If *TAPE,n* is also specified in the READERROR entry, the incorrect block is transferred to another tape for later investigation.

If the address of a user's routine is also specified in the READERROR entry, that routine is executed before the steps of the scan procedure can be taken.

#### Checkpoint

If any tape file contains checkpoint records, this card must be included, with *YES* in the operand field.

## DTF Entry

The DTF Entry (Define The File) describes the file, indicates methods of processing the file, and specifies symbolic addresses of routines and areas unique to the file. A DTF Entry is included for *each* input or output file that is to be processed in the program: one to six tape files, one or more logical files on disk, and one file each for the card reader, card punch, and printer. For assembly, the DTF Entry cards must follow the DIOCS Entry cards and precede the user's source program.

Like the DIOCS, a DTF Entry (Figure 37) consists of a header card followed by a series of detail cards. The header card contains DTF in the operation field and the symbolic name of the file in the operand field. The label field is blank. This symbolic file name is entered in the IOCS macro instructions referring to this file.

The first card after the DTF header card *must* be the detail entry FILETYPE, but all other detail cards may follow in any order. The detail cards must be blank in the operation field. The label and operand fields are punched with the particular specifications for the file. When multiple operands are required for a label, they may appear in any order, unless specified otherwise, and they must be separated by commas. Any detail card may have comments in the operand field, provided two blanks separate the comments from the last operand. When a particular detail entry does not apply to the file, that card must be omitted. All symbolic addresses entered as operands may be character-adjusted. Also, all symbolic addresses except input/output area labels, may be indexed. An operand consisting of a symbolic address with character adjustment and/or indexing may not exceed six characters.

The following lists show the DTF entries that apply to disk, tape, and unit record files. The numbers in parentheses refer to the numbered items in the text and in the *Specifications for DTF Entry Cards* (Figure 35).

| *Disk Files* | *Tape Files* | *Unit Record Files* |
|---|---|---|
| Checklabel (22) | Alttape (12) | Cardpoc (33) |
| EOFAddr (13) | Blkfactor (46) | EOFAddr (13) |
| Filend (17) | Blkseqind (39) | Filetype (1) |
| Filestart (16) | Blocksize (6) | Formcntl (34) |
| Filetype (1) | Chandrive (11) | Overflow (35) |
| Filseq (23) | Checklabel (22) | Workarea (4) |
| Header (27) | Chksum (38) | |
| Indexreg (3) | Chptrcd (47) | |
| IOAreas (2) | Creatsys (43) | |
| Natnpack (10) | Density (37) | |
| NRecords (14) | EOFAddr (13) | |
| NSectors (7) | EXnAddr (29) | |
| Packserial (25) | Filetype (1) | |
| Recform (5) | Fmrecord (44) | |
| Seekeoc (18) | Header (27) | |
| Serialnum (24) | Indexreg (3) | |
| Sizerec (9) | IOAreas (2) | |
| Trackform (8) | Labinfo (36) | |
| Update (15) | Lngrecord (45) | |
| Workarea (4) | Modepar (19) | |
| | Outline (20) | |
| | Padding (30) | |
| | Recform (5) | |
| | Reelseq (26) | |
| | Rewind (31) | |
| | Serialnum (24) | |
| | Sizerec (9) | |
| | Totals (28) | |
| | Tpchkind (40) | |
| | Tpprotch (42) | |
| | Tprectch (41) | |
| | Typelabel (21) | |
| | WLRAddr (32) | |
| | Workarea (4) | |

### 1. Filetype

This card *must* be included as the *first card* after the DTF header card. It states the type of file described in this set of DTF entries. The operand must be punched with *one* of these:

READER – for a card input file.

PUNCH – for a card output file.

DISK, INPUT – for input disk records. *INPUT* may be omitted, if desired, because IOCS assumes an input file whenever *OUTPUT* is not specified. Whenever the IBM 1301 Disk Storage is used, the number 1301 must be included as the second or third operand.

DISK, OUTPUT – for output disk records. Whenever the IBM 1301 Disk Storage is used, the number 1301 must be included as the second or third operand.

TAPE,INPUT – for input tape records. If this file contains *checkpoint* records that should be bypassed by IOCS, the operand *CHECKPOINT* must also be included. The entry would then read *TAPE,INPUT,CHECKPOINT*. With this operand, the checkpoint records are not included in a hash total or counted in block or record counts. Checkpoint records are assumed by IOCS to consist of two records, and they are recognized by IOCS only if the first record is identified by **CHKPT** in the first nine positions. Both records are then bypassed.

TAPE,OUTPUT – for output tape records.

PRINTER – for printed reports.

The type of processing used with disk records is also indicated in this card by:

RANDOM – if records are processed in random order. This may be omitted, if desired, because IOCS assumes random processing whenever *CONSEC* or *CTLSEQ* is not specified.

CONSEC – if records are processed in consecutive order.

CTLSEQ – if records are processed in control-sequential order. If the records contain record marks, *RM* must be included as another operand.

### 2. IOAreas

The symbolic address of the input, or output, area for a disk or tape file is specified in this entry. This must be the same label as specified in the DA statement for the area. The address may not be indexed in this entry.

If two input, or output, areas are assigned when tape records are processed in the *overlap mode*, both symbolic addresses are specified, and separated by a comma, in this entry.

### 3. Indexreg

When an index register is assigned to the input, or output, area for a disk or tape file, the number of the register is specified by *X1*, *X2*, or *X3* in the operand of this entry. This card must be included for:

- all blocked fixed-length records processed in the input area.
- blocked fixed-length tape records built in the output area.
- blocked fixed-length disk records processed in consecutive order and built in the output area.
- unblocked tape records processed in *two* input areas, in the overlap mode.
- unblocked tape records built in *two* output areas, in the overlap mode.

The index register specified in this entry must also be specified in the DA statement for the corresponding input or output area, for records processed in the *non-overlap mode*. For tape records processed in the *overlap mode*, however, the index-register specification must be omitted from the DA statement.

Whenever this entry is included for a file, the DTF WORKAREA entry must be omitted and work areas must not be specified in the GET or PUT macro instructions.

### 4. Workarea

This card is included for an *input* file if all records are to be processed in one work area, or for an *output* file if all records are to be built in one work area. It specifies the symbolic address of the work area. This must be the same as the label of the DA statement for the work area.

The same work area may be specified in the DTF entries for an input file and in the DTF entries for a corresponding output file.

Whenever this entry is included for a file, the DTF INDEXREG entry must be omitted, and work areas must not be specified in the GET or PUT macro instructions.

This card *must* be included for card files if a work area is not specified in GET or PUT and if:

- the Read or Punch Release special feature is used and the Advanced Programming feature is installed.
- the Processing Overlap special feature is used.
- card input/output operations are overlapped with disk-seek operations.

This card *must* also be included for a printer file whenever printer output operations are overlapped with disk-seek operations and a work area is not specified in the PUT instruction.

### 5. Recform

This card indicates the type of records in a tape file

and the type of disk records in a file with consecutive processing:

BLOCKED *must* be specified if records are blocked.

UNBLOCKED *may* be specified if desired, but it is not required. IOCS assumes unblocked records if. *BLOCKED* is not specified.

VARIABLE *must* be specified if records are variable-length.

FIXED *may* be specified if desired, but it is not required. IOCS assumes fixed-length records if *VARIABLE* is not specified.

Thus, this card *must* be included whenever:

- disk records are blocked and processed in consecutive order.
- tape records are blocked and/or variable-length.

### 6. Blocksize

This card must be included for any tape file with blocked records. The size of the input, or output, storage area allotted for the block is entered in the operand field. This does *not* include either the group-mark with word-mark position or the extra position for record-length checking when that feature is specified by the DTF WLRADDR entry.

If two input or output areas are allotted when tape records are processed in the *overlap mode,* the size of *one area* is specified in this entry.

### 7. NSectors

This card must be included whenever disk records are read or written in the sector mode of operation. It specifies the number of sectors required for a record (with unblocked records), or for a block of records. For example, if 70-character records are blocked 4 to a block, 3 sectors (100 characters per sector) are used for each block.

The factor specified in this card becomes the *Sector Count* portion of the *Disk Control Field* when IOCS reads or writes records.

### 8. Trackform

This card is included for disk records and specifies *one* of these operands:

SECTOR, when data is to be read or written by sector (100 characters per sector). One or more sectors of data may be transferred on any disk input or output operation. This *SECTOR* specification may be included if desired, but it is not required because IOCS assumes the sector mode of operation when neither *TRKSECTOR* nor *TRKRECORD* nor *TRKRECORD,ADDRESS* is specified.

TRKSECTOR, when data and sector addresses are to be read or written by track (2000 data characters and 120 address digits). One complete track is transferred on any disk input or output operation.

TRKRECORD, when the Track Record special feature is installed and only data is to be read or written per track (2980 data characters). One full track of data is transferred on any disk input or output operation.

TRKRECORD,ADDRESS, when the Track Record special feature is installed and both the data and address are to be read or written per track (2980 data characters and 6 address digits). One full track is transferred on any disk input or output operation.

### 9. Sizerec

This card must be included for any disk or tape file with blocked records, or with unblocked fixed-length records.

For fixed-length records (blocked or unblocked), this card specifies the number of characters in the data record, including the record mark, if any. (This does *not* include the extra position allotted in core storage for record-length checking, when that feature is specified by the DTF WLRADDR entry.)

For variable-length blocked records, this card specifies the low-order position of the record-length field. For example, if the record-length field is located in positions 13, 14, and 15 of the record, "15" is entered in the operand of this card. The record-length field is included in each record, and it specifies the number of characters in the record, including itself and the record mark.

The record-length field must be defined by a word mark in the work area(s) specified in the DTF entry (WORKAREA) or in the GET or PUT instructions.

### 10. Nat0pack, Nat2pack, Nat4pack, Nat6pack, and Nat8pack

One (or more) of these cards is included to specify the 1311 disk drive(s) and/or 1301 disk module(s) used for a logical file.

The card(s) corresponding to the sector addresses on a 1311 disk pack, or to the sector addresses assigned to a 1301 disk module, must be included. For example, if the records are written on a 1311 disk pack with sector addresses 040000-059999, or if addresses 400000-599999 are assigned to a 1301 module, the NAT4PACK card is included. The relationship of sector addresses and these *natural-pack* cards is:

| *1301 Sector Addresses* | *1311 Sector Addresses* | *Natural Pack Card* |
|---|---|---|
| 000000-199999 | 000000-019999 | 0 |
| 200000-399999 | 020000-039999 | 2 |
| 400000-599999 | 040000-059999 | 4 |
| 600000-799999 | 060000-079999 | 6 |
| 800000-999999 | 080000-099999 | 8 |

When a logical file of records covers more than one 1311 disk pack, or more than one 1301 module, two or more natural-pack cards are included. If a file is recorded on two 1311 disk packs with sector addresses NAT0PACK and NAT2PACK cards are used.

When a 1311 is used, the disk drive on which the 000000-019999 and 020000-039999, for example, the pack is mounted for this operation is specified by entering the disk drive control number in the operand field of the card selected. For example, if the pack with sector addresses 040000-059999 (NAT4PACK) is mounted on disk drive 2 (its natural drive), a "4" is punched. However, if this pack is mounted on disk drive 3 (an alternate drive), a "6" is punched. The disk drives and the associated control numbers are:

| *Disk Drive* | *Control No.* |
|---|---|
| 0 | 0 |
| 1 | 2 |
| 2 | 4 |
| 3 | 6 |
| 4 | 8 |

When a 1301 is used, the module to which the addresses are assigned is specified by entering the numerical portion of the disk module control character in the operand field of the card selected. For example, if addresses 400000-599999 (NAT4PACK) are assigned to disk module 2 (their natural module) a "4" is punched. The disk modules, control characters, and the numerical porition that is to be punched are:

| *Disk Module* | *Control Character* | *Numerical Portion Punched* |
|---|---|---|
| 0 | +0 | 0 |
| 1 | B | 2 |
| 2 | D | 4 |
| 3 | F | 6 |
| 4 | H | 8 |

### 11. Chandrive

The number (1-6) of the tape drive unit used for a tape input or output file is specified in the operand of this entry. Each logical file must be assigned to a different tape unit. In a multi-reel tape-file operation using two tape drives, this entry specifies the *original* drive unit, while the ALTTAPE entry specifies the *alternate* drive unit.

### 12. Alttape

This card specifies the number (1-6) of a tape drive unit used as an *alternate* for a tape file that has two or more reels of data. The *original* drive unit is specified in DTF CHANDRIVE.

### 13. EOFAddr

This entry must be included for any card or tape input file, and for any file of disk records processed in consecutive or control-sequential order. The user specifies, in the operand, the symbolic address for his end-of-file routine. On a last-card indication, or on a tape or disk end-of-file condition, IOCS automatically branches to this routine, where the programmer generally issues the CLOSE instruction for the file.

In some cases IOCS also branches to this routine on an EOR condition in a disk file (see *Trailer Label*).

### 14. NRecords

This card must be included whenever blocked fixed-length disk records are processed. It indicates the number of records per block by specifying a number one-less-than the blocking factor for the file. (For example, digit 9 represents 10 records per block). Any digit 0-9 may be entered in the operand when *RANDOM* is specified in the DTF FILETYPE entry. When *CTL-SEQ* is specified, any number 0-29 may be entered. When *CONSEC* is specified, any number 0-99 may be entered.

### 15. Update

This card must be included for a random or control-sequential disk input file if the PUT macro instruction is to be used for that file. *YES* is entered in the operand field.

### 16. Filestart

This card specifies the beginning of a logical file. The six-digit sector address of the first record, or block of records, in the file is entered in the operand. It must be included when:

- input or output records are processed in consecutive order.
- records are processed in control-sequential order (input file).
- output records are processed in random order and header labels are used.

When header labels are used, this card provides the address for checking or writing the *Lower Limit* field on the first pack of a logical file. Before writing the header label for an output file, IOCS checks any labels previously written on the pack to ensure that the portion of the pack between the FILESTART and FILEND specifications does not contain active data. If it does, the user can mount a different pack, or he can use this pack anyway. If he uses this pack, IOCS automatically deletes the old label by *blanking* the *File Identification* field.

If a logical file is processed in consecutive order and consists of two or more file areas, this card specifies the starting address of the first file area. The starting address of each additional file area is specified by a RDLIN card when the EOR (end-of-record) condition occurs.

### 17. Filend

This card specifies the end of a logical file. The six-digit sector address of the last record, or block of records, in the file is entered in the operand. If a file is processed in consecutive order and handled in the file areas, however, this card specifies the ending address of the *first* file area. FILEND must be included when:

- input or output records are processed in consecutive order.
- records are processed in control-sequential order (input file).
- output records are processed in random order and header labels are used.

When header labels are used, this card provides the address for checking or writing the *Upper Limit* field on the last pack of a file, or on the last pack of the *first* file area. Before writing the header label for an output file, IOCS checks any labels previously written on the pack to ensure that the portion of the pack between the FILESTART and FILEND specifications does not contain active data. If it does, the user can mount a different pack, or he can use this pack anyway. If he uses this pack, IOCS automatically deletes the old label by *blanking* the *File Identification* field.

When a multi-pack logical file is processed in consecutive order and consists of two or more file areas, this card specifies the ending address of the last pack of the first file area. The ending address of each additional file area is specified by a RDLIN card when the EOR (end-of-records) condition occurs.

### 18. Seekeoc (1311 Only)

When only *one* logical file is stored in a 1311 disk pack and that file is processed in consecutive or control-sequential order, this card may be included to reduce access time. *YES* is specified in the operand field.

This entry automatically eliminates all seek operations for the file except at the end of a cylinder (determined by an unequal-address-compare condition) or when an out-of-order record is to be read in a control-sequential file (see *Control-Sequential Processing*). Whenever SEEKEOC is specified for a file, it takes precedence over the functions of the Direct Seek special feature or the DIOCS SEEKCHECK entry if either of these is included in the program.

### 19. Modepar

This card must be included if tape records are to be read or written in the load mode. *LOAD* is entered in the operand. If this card is omitted, IOCS automatically reads or writes tape records in the move mode, but this card may be included with *MOVE* in the operand, if desired.

### 20. Outlin

This entry applies to blocked records. The time required by IOCS to block or deblock records *in-line* is less than the time required to block or deblock *out-of-line*. However, more core storage is required for in-line blocking or deblocking than for out-of-line whenever more than one GET or PUT instruction is issued for a file.

With this entry, the user can select the operation that best suits his program. When this entry is included, with YES in the operand field, blocking and deblocking will be performed out-of-line. Only a linkage to these routines will be placed in-line. If this entry is omitted, however, the blocking and deblocking routines will be placed in-line.

### 21. Typelabel

Whenever an input or output tape file contains header and trailer labels, this card must be included to specify the type of label:

STANDARD, whenever standard header and trailer labels are used. The standard label may be followed by additional (nonstandard) labels.

NONSTANDARD, when nonstandard labels are used.

Whenever STANDARD is specified, the size of the label must also be specified by entering *one* of these as a second operand:

A, for a 120-character label.
B, for an 80-character label.
C, for an 84-character label.

This entry is also used to indicate that IOCS should pass a tape mark (if any) between a header label and the first data record, when an input tape with Type B or Type C labels is processed. For this, *TM* is entered in the operand. (IOCS automatically passes a tape mark when Type A labels are used.)

### 22. Checklabel

With this entry, the programmer specifies the checking procedure he desires for a header label in a 1311 *disk* file, or for standard header and trailer labels in a *tape input* file. He enters *one* of these two specifications in the operand:

ALL

*Disk* – to completely check a header label for an input file, or to write a header label for *any* output file. The nine fields handled by IOCS are processed.

*Tape* – to completely check the standard header and trailer labels. The fields checked by IOCS in each type of label are:

| *Type* | *Header/Trailer* | *Field Numbers* |
|---|---|---|
| A | Header | 1-7 if no field 9-19 is DTF-specified<br>1-19 if any field 9-19 is DTF-specified |
| | Trailer | 1-2 |
| B | Header | 1-7 |
| | Trailer | 1-4 |
| C | Header | 1-8 |
| | Trailer | 1-3 |

IDENT

*Disk* – to partially check the header label in an *input file*. Only the ten-position *File Identification* field is checked. This specification is not used for an output file.

*Tape* – to partially check the standard header, and completely check the standard trailer. Only the *File Identification* field is checked in the header label.

### 23. Filseq (1311 Only)

When a multi-pack logical disk file (1311) with a header label is processed, the *File Sequence Number* field in the header label of the first pack is automatically numbered 0001 by IOCS, if no other number is specified. If the programmer wants a different number on the first pack, he includes this FILSEQ entry and specifies a four-digit number. This is used to check the header label in the first pack or on input, or to write a header label on output.

After the number of the first pack is determined (0001 or some other number), it is automatically increased by one for each succeeding pack.

This entry may be omitted if a RDLIN macro instruction is issued prior to opening the file.

### 24. Serialnum

This card specifies the five-digit *File Serial Number* for a standard 1311 disk or tape header label and must be included for:

- an input file when complete header-label checking is DTF-specified.
- an output file if the *File Serial Number* is to differ from the *Pack Serial Number* or from the *Reel* (or *Tape*) *Serial Number*.

If this card is omitted, the *file-serial* number is assumed to be the same as the *pack* or *reel-serial* number. With a multi-pack disk file, the pack-serial number of the first pack becomes the file-serial number in all packs. With a multi-reel tape file, the reel-serial number of the first reel becomes the file-serial number in all reels.

This entry may be omitted if a RDLIN macro instruction is issued prior to opening the file.

### 25. Packserial (1311 Only)

This card specifies the five-digit *pack serial number* for checking a header label on a 1311 disk pack. It must be included for:

- an input file when complete checking is specified.
- an output file. IOCS checks that records will be written on the correct pack.

If a logical file is written on two or more packs, the number of the *first* pack must be specified. If a logical file is processed in consecutive order and consists of two or more file areas, this card specifies the pack serial number of the first pack of the first file area.The number of the first pack of each additional file area is specified by a RDLIN card.

This entry may be omitted if a RDLIN macro instruction is issued prior to opening the file.

### 26. Reelseq

When a multi-reel tape file with standard labels is processed, the *Reel Sequence Number* field in the header label of the first reel is automatically numbered 001 (Type B Label) or 0001 (Types A and C labels) by IOCS, if no other number is specified. If the programmer wants a different number in the first reel, he includes this REELSEQ entry. He must specify a three-digit number for Type B labels, and a four-digit number for Type A or Type C labels. This is used to check the header in the first reel on input, or to write the header on output.

After the number of the first reel is determined (001, 0001, or some other number) it is automatically increased by one for each succeeding reel, or it may be increased or decreased by some other factor by using Exit 6.

### 27. Header

This card applies to both tape and 1311 disk files and must be included for:

- an input file with standard header labels when checking is DTF-specified.
- an output file with standard header labels.

It specifies the information for checking three fields in an input header label, or for writing two fields in an output header label:

1. *File Identification:* This is included for input and output files.

   *Disk* – Ten alphamerical characters (see *Header Label, Item 4*).

   *Tape* – Ten alphamerical characters or blanks for Type A or Type B labels. Eighteen alphamerical characters or blanks for Type C labels.

2. *Creation Date:*
   *Disk and Tape* – Five digits – two for year, followed by three for day. This is included only for an input file that is to be completely checked. It is not included for an output file, because the date is taken from today's date in storage.
3. *Retention Period:* This is included as the third operand for an input file that is to be completely checked, or as the second operand for an output file.
   *Disk* – Four digits.
   *Tape* – Three digits for Type B or Type C labels. Four digits for Type A labels.

When two (output file) or three (input file) of these operands apply, they *must* be written in the sequence listed and separated by a comma. This entry may be omitted if a RDLIN macro instruction is issued prior to opening the file.

### 28. Totals

This card specifies the totals (record or hash) required for tape files. To accumulate a record count, *RECORD* is entered in the operand. To accumulate a hash total, the low-order position of the chosen field, within the record, is specified. For example, if the field is located in record positions 16-21, "21" is entered in the operand. This field must be defined by a word mark in the storage area referred to by IOCS. This is the work area if it is specified in the DTF entries; otherwise, it is the input (or output) area.

These totals may be specified for tape files with standard or nonstandard labels, but not for unlabelled files.

When Type B standard labels are used. IOCS checks or writes these totals in their corresponding trailer-label fields. When Type A or Type C standard labels are used and these totals are required, the programmer must provide for checking or writing them in the *User's Information* area of the trailer label.

The totals accumulated throughout the run are available at these symbolic addresses:

Record Count – IOCnRC (10-position field).
Hash Total – IOCnHS (10-position field).

The letter "n" represents the number of original tape unit assigned to the file in the DTF entry CHANDRIVE.

### 29. EX1Addr through EX8Addr

These eight cards are used in conjunction with tape input/output files. They provide the means of branching from the IOCS routines to a user's routine for processing header or trailer labels. The symbolic address of the user's routine is entered in the operand of the EXIT card selected. Each exit (1-8) occurs at a specific time in the IOCS routines and is used for a specific function, as indicated briefly in Figure 37 and described under *Label Operation.* The programmer includes the appropriate EXIT cards according to the operations he wishes to perform.

### 30. Padding

If blocked fixed-length records in an output tape are to be padded with some character other than blanks, this card is included. It can specify any one character except: asterisk (*), substitute blank (ƀ), group mark (‡), record mark (‡), tape mark (√‾), word separator (v), or *at* sign (@).

### 31. Rewind

If no specifications are given by the programmer, tape files are automatically rewound, but not unloaded, on an OPEN or CLOSE instruction and on an end-of-reel condition. If other operations are desired for a tape input or output file, this card may specify:

UNLOAD, to rewind the tape on OPEN, and to rewind and unload on CLOSE or an end-of-reel condition.

NORWD, to prevent rewinding the tape at any time.

### 32. WLRAddr

This entry causes IOCS to check the length of records read from tape. If a wrong-length record (unblocked record or block of records) is read, programming branches to the user's routine specified by the symbolic address in the operand.

At the end of his routine the user must return to IOCS by branching to either IOCDMP or IOCRMV. Returning to IOCDMP causes IOCS to *dump* the wrong-length record on another tape and read the next record. The tape on which the wrong-length records are to be written must be specified by *TAPE,n* in the DIOCS READERROR entry. Thus, that tape will contain both wrong-length and parity-error records. Returning to IOCRMV causes the next record to be read, but the wrong-length record is *not* written.

This card may be included for input tape files that contain blocked records (fixed- or variable-length) or unblocked fixed-length records. Unblocked variable-length records cannot be checked for correct record-length.

Whenever this entry is included for a tape file, one extra core storage position must be provided in the input area for that file. This is specified by the DA statement(s).

### 33. Cardpoc

When all the cards in a file are to be selected into the same pocket, the number of that pocket (1, 2, 4, or 8)

is specified in the operand of this entry, if it is *not* specified in the GET or PUT macro instructions. If the Read Release special feature is used, any selection of card input files *must* be specified in this entry, not in the GET instruction. Cards from an input file *cannot* be selected when records are processed in the *overlap mode.*

### 34. Formcntl

Spacing or skipping of forms is specified in this card if it is to be the same for every line written on a form and if it is *not* included in either the PUT or SPACE macro instructions. The standard IBM 1401/1460 d-character for the desired control is entered in the operand.

### 35. Overflow

This card defines the operation to be performed if an overflow condition occurs in the printer carriage. The number of the carriage-tape channel used to indicate the overflow is entered in the operand. This may be either channel 9 or channel 12. When an overflow occurs, processing of data is interrupted after the detail line is printed, the carriage is restored to channel 1, and processing resumes.

If he prefers, the programmer may branch to his own routine on an overflow condition (the detail line has already been printed). For this he enters the symbolic address of his routine as the second operand in this card. This allows him to print an overflow-page heading. In his routine he may perform any carriage-control operation, such as a skip to channel 1. Then he can move the page-heading data to the printer output area, and return to IOCS by branching to IOCRET (IOCS Return). IOCS prints the one header line, and processing continues.

If a job requires multiple heading lines, the programmer should write all but the last line in his routine, using *Autocoder* instructions to perform the write operations. Then, IOCS writes the *last* line (set up in the printer output area) when the branch to IOCRET occurs. In this type of operation, the programmer must *not* issue any IOCS macro instructions in his print overflow routine.

An alternate method of writing multiple heading lines allows the programmer to issue a PUT instruction in his routine, after he has set up the data for each heading line. To use PUT, the programmer must save the 3-position IOCS address stored at IOCUXT+3 at the beginning of his routine, and restore it immediately before branching to IOCRET. In this type of operation, PUT functions as specified for the detail lines of the report. That is, data is written either from the output area, or from a work area if one is specified for this output file. Therefore the programmer, in his routine, must set up the heading-line data in the corresponding area. However, the *last* heading line is written upon return to IOCRET, and therefore *must always* be set up in the printer output area.

### 36. Labinfo

This card may be included to specify five codes for checking or writing the *Label Information* field (No. 7) of a Type C tape header label. Starting with the first position of the operand field (column 21), one of the specified codes must be punched for each of the five factors:

| *Factor* | *Code* |
|---|---|
| Density | 0 for low density<br>1 for high density |
| Character Coding | 0 for BCD coding<br>1 for binary coding |
| Checksum | 0 if checksum is not used<br>1 if checksum is present |
| Block-Sequence | 0 if block sequence is not used<br>1 if block sequence is present |
| Checkpoint Record | 0 if checkpoint record is not used<br>1 if checkpoint record is present |

Whenever this card is included, all five columns *must* be punched in the operand field.

If this card is omitted, and a header label is completely checked, or written, IOCS uses blanks for these codes.

This entry does not apply to Type A or Type B standard header labels, and it must be omitted when they are processed.

### 37. Density

This card may be included for checking, or writing, the *Density Indicator* field (No. 9) of a Type A tape header label. It specifies one of these three single-position codes:

0, when density coding is not applicable
1, for low density
2, for high density.

If the 19 header-label fields are processed but this card is omitted, IOCS uses code "0" for checking or writing field 9. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 38. Chksum

This card may be included for checking, or writing, the *Checksum Indicator* field (No. 10) of a Type A tape header label. It specifies one single-position code:

0, when a checksum does not apply or is not present.
1-9, for the type of checksum used (standard single-position codes will be provided as needed).

If the 19 header-label fields are processed but this card is omitted, IOCS uses code "0" for checking or writing field 10. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 39. Blkseqind

This card may be included for checking, or writing, the *Block Sequence Indicator* field (No. 11) of a Type A tape header label. It specifies one single-position code:

0, when block sequence does not apply or is not present.
1-9, for the type of block sequence used (standard single-position codes will be provided as needed).

If the 19 header-label fields are processed but this card is omitted, IOCS uses code "0" for checking or writing field 11. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 40. Tpchkind

This card may be included for checking, or writing, the *Tape Checking/Interpreting Indicator* field. (No. 12) of a Type A tape header label. It specifies one single-position code to indicate the character coding used for this tape file:

1, for binary coding.
2, for BCD coding.
3, when the coding need not be specified.

If the 19 header-label fields are processed but this card is omitted, IOCS uses code "3" for checking or writing field 12. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 41. Tprectch

This card may be included for checking, or writing, the *Tape Data Recording Technique Indicator* field (No. 13) of a Type A tape header label. It indicates the number of bits (not including a check bit) used as a unit, for characters recorded on the tape. Because the IBM 729 and the IBM 7330 process data recorded in 6-bit code, this card specifies a "6" for 1401 systems operation.

If the 19 header-label fields are processed but this card is omitted, IOCS uses a "6" for checking or writing field 13. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 42. Tpprotch

This card may be included for checking, or writing, the *Tape Data Processing Technique Indicator* field (No. 14) of a Type A tape header label. It indicates the number of bits (not including a check bit) out of a byte that are to be treated as a unit in processing.

If the 19 header-label fields are processed but this card is omitted, IOCS uses a blank for checking or writing field 14. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 43. Creatsys

This card may be included for checking, or writing, the *Creating System* field (No. 15) of a Type A tape header label. It indicates the IBM data processing system on which the tape file was written, by specifying a four-digit system number.

If the 19 header-label fields are procesed but this card is omitted, IOCS uses system number "1401" for checking or writing field 15. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 44. Fmrecord

This card may be included for checking, or writing, the *Record Format* field (No. 16) of a Type A tape header label. It specifies one single-position code to indicate the record format for this tape file:

A, for fixed-length, unblocked records.
F, for fixed-length, blocked records.
W, for variable-length, unblocked records.
X, for variable-length, blocked records.

If the 19 header-label fields are processed but this card is omitted, IOCS uses a blank for checking or writing field 16. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 45. Lngrecord

This card may be included for checking, or writing, the *Record Length* field (No. 17) of a Type A tape header label. A five-digit number is punched to specify:

- The number of characters per record, for fixed-length records.
- The number of characters in the largest record, for variable-length records.

If the 19 header-label fields are processed but this card is omitted, IOCS uses blanks for checking or writing field 17. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 46. Blkfactor

This card may be included for checking, or writing, the *Blocking Factor/Size* field (No. 18) of a Type A tape header label. A five-digit number is punched to specify:

- The number of records per block, for fixed-length records.
- The number of characters in the largest block, for variable-length records.

If the 19 header-label fields are processed but this card is omitted, IOCS uses blanks for checking or writing field 18. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

### 47. Chptrcd

This card may be included for checking, or writing, the *Checkpoint Indicator* field (No, 19) of a Type A tape header label. It specifies one single-position code:

0, when checkpoint records do not apply or are not present.
1-9, for the type of checkpoint record used.

If the 19 header-label fields are processed but this card is omitted, IOCS uses code "0" for checking or writing field 19. Whenever this card is included and complete checking, or writing, is specified, all 19 header-label fields are processed.

This entry does not apply to Type B or Type C standard header labels, and it must be omitted when they are processed.

# Index

*Whenever one reference has more significance than the others for an item, that page number is listed first.*

*Title:* Technical Bulletin No. 7

**IBM** 1440 Systems

*Dept.:* Programming Systems

*Date:* December 3, 1963

*Interest Area:* IBM 1440 Systems

*Category:* Programming

# 1440/1311 IOCS DTF STRUCTURE

*Classification:* IOCS

The following information concerns the structure of 1311 Disk DTF tables and may be of value to you in implementing 1440 IOCS.

C. R. White

## FILE DTF STRUCTURE

| Referenced Filename | Random | Consec | Ctl-Seq |
|---|---|---|---|
| -1 to 5 | ALT & LABELLING CODE (NOTE 1) | Same | Same |
| FILENAME | DTF CODE (NOTE 2) | Same | Same |
| + 3 | CONTENTS OF INDEX REGISTER | Same | Same |
| 6 | ADDR OF 1ST RECORD (NOTE 3) | Same | Same |
| 12 | DYNAMIC DISK ADDR (NOTE 4) | Same | Same |
| 15 | NSECTORS | Same | Same |
| 18 | OP CODE & 1ST TWO POSITIONS OF XCTRL FLD | Same | Same |
| 19 | MODE | Same | Same |
| 22 | ADDR OF IOAREA | Same | Same |
| 23 | NRECORDS | ZERO | NRECORDS |
| 26 | RCD LENGTH (NOTE 6) | Same | Same |
| 27 | PADDING (1 pos) | | DYNAMIC RCD CODE (NOTE 5) |
| 28 | PADDING (1 pos) | Unblkd- PADDING (2 pos) Blkd- NRECORDS | Not RM |
| 34 | FILESTART | Same | Same |
| 40 | FILEND | Same | Same |
| 41 | PADDING (1 pos) | | PUT INDICATOR (1 pos) (NOTE 7) |
| 42 | | CURRENT RCD IN BLK | |
| 44 | PADDING (3 pos) | Unblkd- PADDING (2 pos) Blkd- PADDING (2 pos) | EOFADDR (3 pos) |
| 48 | RTN PERIOD | Same | Same |
| 53 | CREATION DATE | Same | Same |
| 63 | IDENTIFICATION | Same | Same |
| 68 | SERIAL NO. | Same | Same |
| 73 | PACK SER. NO. | Same | Same |
| 74 | PADDING (1 pos) | Same | Same |
| 78 | FILE SEQUENCE | Same | Same |

Note 1 — ADDR range of drives 0-2-4-6-8 are indicated. In addition, FN-5 has a zone structure indicating type of label checking to be performed.

| | |
|---|---|
| AB | - All |
| B | - Ident |
| No Zone | - No Checking |

Note 2 — File Codes:

| | |
|---|---|
| 3 | Consec Input |
| 4 | Consec Output |
| 5 | Control-Sequential |
| 6 | Consec-Output-Indexed |
| R | Random Input |
| W | Random Output |

Note 3 — Actual address of the record being processed. Reflects high order position. Reflects current record in blocked file.

Note 4 — Random and Ctl-Seq reflect current address of record or block of records being processed.

Consecutive - Address of the next record or block of records.

Note 5 — Code:

| | |
|---|---|
| blank | Very first record in file |
| O | First record of block |
| S | End of file |

Note 6 — This is the last entry of a random non-labelled file.

Note 7 — With Adv Prog:

| | | |
|---|---|---|
| WM | = | Put Given |
| NOWM | = | No Put Given |

Without Adv Prog:

| | | |
|---|---|---|
| 1 | = | Put Given |
| 0 | = | No Put Given |

Note: Filename 48 to 78 are used only when label checking is to be performed.

IBM 1401/1460-30 Printed in U. S. A. C24-1489-3

IBM

International Business Machines Corporation

Data Processing Division

112 East Post Road, White Plains, N. Y. 10601