UNIVAC®
490
REAL-TIME SYSTEM

TECHNICAL BULLETIN

SYSTEM CONVENTIONS

Programmer's Reference

First Edition

August, 1962

PREFACE

This document is a description of conventions for programs operating under the control of standard routines within the UNIVAC® 490 Real-Time System. Sections I through V assume the use of standard software packages such as the Real-Time Executive Routine (REX). Section VI describes flow charting and coding as it will appear in the technical documentation for standard software packages.

TABLE OF CONTENTS

I. TAPE CONVENTIONS

Under UNIVAC 490 Real-Time System conventions, two types of input/output tapes are recognized:

Tapes containing instructions or programs that will process data. For the purpose of these conventions, such tapes will be called Master Instruction Tapes (MIT's).

Tapes containing data to be processed, or that have resulted from processing, by a program.

In some ways this distinction is artificial (for example, while they are being placed on tapes or being loaded from tape, the instructions on the MIT are data) but is is retained for reference purposes.

A. MASTER INSTRUCTION TAPES

It is assumed that all programs have been originally coded in SPURT or COBOL language and hâve gone through compilation, resulting in object programs that are to be loaded by and will run under the control of the Real-Time Executive Routine (REX).[1]

The MIT is created by a Utility Run from SPURT output. The resultant tape is in a form acceptable for REX loading and control.

B. DATA TAPE CONVENTIONS

Basically, a file may be regarded as a collection of data that conforms to desired limits. The data arrangement within the file is in units called records or items. These units consist of one or more fixed length (30-bit) computer words. The length of the item is always expressed in multiples of computer words. When recorded on magnetic tape, the items comprising a file are written in groups called blocks. These consist of a number of computer words.

[1] SPURT processing is the final stage of COBOL compilation. Thus the object program for REX control is SPURT output.

For example, given an item length of five words, a block of twenty-five word length could contain five items (disregarding, for the moment any block identifying words).

Besides the blocks of data, there are certain conditions that must be signaled for proper handling of a file:

1. The beginning of a file or tape.

2. The end of a file.

3. The end of a tape when a file extends over more than one reel.

4. The presence of blocks on the tape that are not, properly, a part of the data.

Each of these conditions is signaled by the use of a special type of block at particular points in the magnetic tape file.

In the UNIVAC 490 Real-Time System, the size of blocks to be written on or read from magnetic tapes is a function of a program or routine. Input and Output in the UNIVAC 490 System is handled by Input/Output Functional Subroutines under REX control. The following standards have been established for the Magnetic Tape Input/Output File Control Subroutines:

1. The maximum data block size is 4096 computer words.*

2. Data blocks within a file must be of the same size except the last data block which may be a short block.

3. Items within a file must be of the same size.

4. An item cannot be partially in one block and partially in another.

*This does not apply to rerun dumps on data tapes.

5. Each data block is identified by the presence of block descriptors as the first and last words of the block. (The content of the data block descriptor words is detailed below)

6. Aside from the blocks of data, each file must contain certain standard sentinel blocks as necessary:

   Label Block
   Bypass Sentinel Block
   End-of-Reel Sentinel Block
   End-of-File Sentinel Block

7. Any Input/Output close-out subroutine will type out the following information:

   $\left.\begin{array}{l}\mathrm{IN}\Delta\Delta\\ \mathrm{OUT}\Delta\end{array}\right\}$ CnnΔSnΔ Label Δ yydddΔrr

   where IN or OUT is used depending on whether the file is an input or an output.

   Cnn is channel number in octal.

   Sn is servo number in octal.

   Label is the file identifier as contained in words 2 through 4 of the Label block (in Fieldata code; see Label Block, below).

   yyddd is the date as contained in word 5 of the Label block (in Fieldata code; see Label Block, below).

   rr is the reel number expressed in octal. This number is contained,right justified,in one computer word. Before printing, leading zeros are suppressed.

   This type out is preceded by the number assigned by REX to the program processing the file and is provided by REX.

II. DATA TAPE BLOCK CONVENTIONS AND FORMATS

A. Data Blocks

Aside from the restriction of 4096 computer words maximum length, and the use of the first and last block words, the content of data blocks is a function of the program. Any program or file control routine used is responsible for arranging the data block and delivering it to REX for processing by the functional subroutine.

The first and last words of a data block are called block descriptors and contain:

```
Bit position 29 28 27 26                      15 14                                  0
            +--+--+--+--------------------------+-------------------------------------+
            |0 |0 |0 | Number of items in block | Number of Computer words in block   |
            +--+--+--+--------------------------+-------------------------------------+
```

where:

bit positions 0 through 14 contain the octal expression of the number of computer words in the block, including the block descriptors. This number is right-justified.

bit positions 15 through 26 contain the octal expression of the number of items in the block between the block descriptors. This number is right-justified.

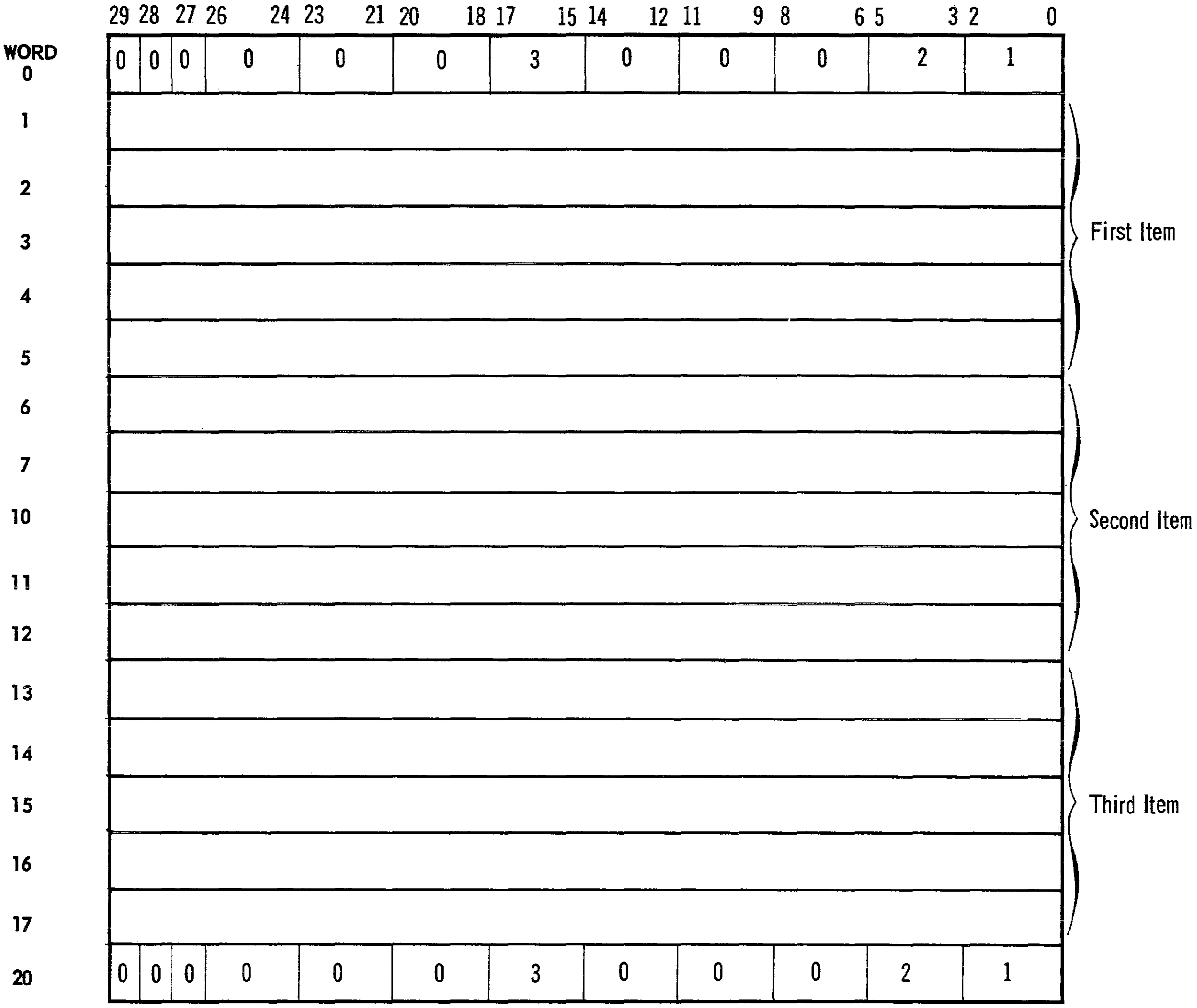
bit position 29 contains a binary zero to identify this block as a data block.[2]

For example, the content of the first and last words of a data block seventeen words long containing three five-word items would be:

```
Bit positions 29 28 27 26                      15 14                            0
             +--+--+--+------------------------+------------------------------+
             |0 |0 |0 |0 0 0 0 0 0 0 0 0 0 1 1|0 0 0 0 0 0 0 0 0 0 1 0 0 0 1|
             +--+--+--+------------------------+------------------------------+
             Octal                           3                             21
```

[2] A binary zero in the high order bit position will cause the block descriptor word to test positive.

The complete block would be:



B. Data Standard Sentinel Blocks

The standard sentinel blocks associated with a data file are of twenty-four words each. The first two words and the last two words of such blocks (words 0, 1, 22 and 23) contain specific codes to uniquely identify each type of block. The codes associated with each are the octal expression of Fieldata characters. Five Fieldata characters make up each word. The standard sentinel blocks and their associated characters and code are:

Block Type	Fieldata Character	Octal	Binary Expression
Label Block	;	73	111011
Bypass Sentinel Block	/	74	111100
End-of-Reel Block	.	75	111101
End-of-File Block	(Special Character)	76	111110

Thus, the block descriptor words for a Bypass Sentinel block, for example, would contain:

Binary	111100	111100	111100	111100	111100[3]
Octal	7 4	7 4	7 4	7 4	7 4
Fieldata	/	/	/	/	/

1. Standard Sentinel Block Use

a. Each tape file and each reel of a tape file must begin with a Label block.

b. If within a file there is information that is not part of the data as such (for example, a memory dump for rerun), the block or blocks containing such information must be preceded by a Bypass sentinel block and followed by a Bypass sentinel block.

Bypass information with the associated Bypass sentinel blocks can only appear:

[3] Since each Standard sentinel block must contain a binary 1 in the high order bit position, these block descriptor words will test negative.

(1) After the Label block of a file or tape

and

(2) Before the other standard sentinel blocks (End-of-Reel or End-of-File sentinel blocks)

c. When a file extends over more than one tape reel (a multi-reel file), each reel except the last will have two End-of-Reel sentinel blocks after the last data block.

d. The last data block of a file must be followed by two End-of-File sentinel blocks.

e. Thus, the standard sentinel blocks associated with a magnetic tape data file are:

(1) One Label block at the beginning of the file (or each tape of the file for a multi-reel file).

(2) One Bypass sentinel block preceding and one Bypass sentinel block following information to be bypassed in a file.

(3) Two End-of-Reel sentinel blocks for each intermediate tape in a multi-reel file.

(4) Two End-of-File sentinel blocks after the last data block of a file.

f. Unused words, fields, or portions of fields are filled with binary zeros.

C. Standard Block Format and Content

1. Label Block

WORD										
0	7	3	7	3	7	3	7	3	7	3
1	7	3	7	3	7	3	7	3	7	3
2	FILE									
3	IDENTIFIER									
4										
5	DATE OF CYCLE									
6	REEL NUMBER									
7	TIME									
8	BLOCK SIZE									
9	ITEM SIZE									
10	UNUSED									
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										
22	7	3	7	3	7	3	7	3	7	3
23	7	3	7	3	7	3	7	3	7	3

A Label block is twenty-four words in length. It must be the first block of each tape file and the first block of each tape reel of a multi-reel file.

Word 0 is a block descriptor word and contains five Fieldata ";"'s (octal 73's).

Word 1 Same as word 0.

Word 2
Word 3
Word 4 } File identification written as fifteen Fieldata characters. File identification is left-justified. Unused bit positions contain binary zeros.

Word 5 Date of cycle-the date the current file is created. It is expressed in five (5) Fieldata numerics in the format:

yyddd

where yy = year
ddd = day of year and is in the range 001 through 366

For example the 29th of February 1964 would be:

Fieldata	6		4		Ø		6		0	
Octal	6	6	6	4	6	Ø	6	6	6	0
Binary	110	110	110	100	111	001	110	110	110	100.

Each reel of a multi-reel file must contain the same date.

Word 6 Reel number is expressed in octal code right justified. Each reel of a multi-reel file must contain the number of the reel within the file.

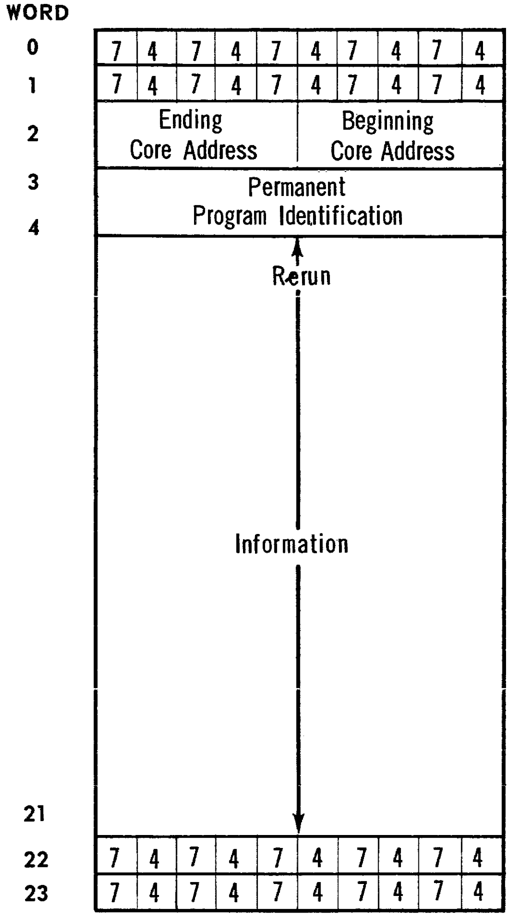
Word 7 Time of file creation. This is the reading of the DAY clock as provided by REX when the first Label block is created. Each label block of a multi-reel file must contain the same time entry.

Word 8 Block size is the octal expression of the size of the data blocks of the file. This octal number is right-justified. Unused bit positions contain binary zeros.

Word 9 Item size is the octal expression of the number of computer words which contain an item. It will never be less than 1 nor more than block size minus 2. This octal number is right-justified. Unused bit positions contain binary zeros.

Word 10	Unused. Available for installation use as desired.
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
Word 22	Same as word 0
Word 23	Same as word 0

2. Bypass Sentinel Blocks



Bypass sentinel blocks are used for rerun purposes only. They are placed before and after information on the tape. They cannot appear before the Label block of a tape or after End-of-Reel or End-of-File sentinel blocks of a tape.

Word 0 is a Block Descriptor and consists of five Fieldata "/"'s.(octal 74's).

Word 1 Same as word 0

Word 2 Number of blocks on the tape, up to and including the sentinel block.

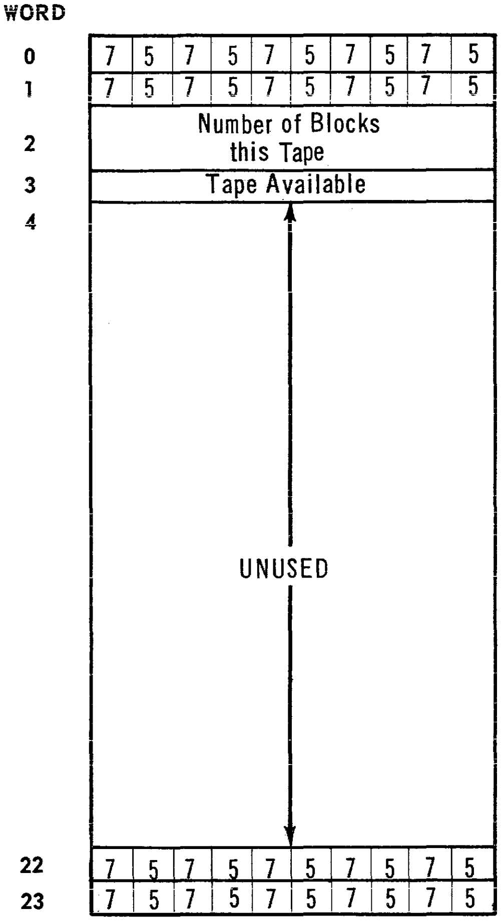
Word 3 Tape availability is the octal number of words available on tape following this block. This octal number, right-justified is placed in this word.

Word 4 . . . 21 For rerun information created by the Rerun routine. Not available for any other use.

Word 22 Block Descriptor - same as word 0.

Word 23 Block Descriptor - same as word 0.

3. End-of-Reel Sentinel Block



WORD										
0	7	5	7	5	7	5	7	5	7	5
1	7	5	7	5	7	5	7	5	7	5
2	Number of Blocks this Tape									
3	Tape Available									
4	UNUSED									
22	7	5	7	5	7	5	7	5	7	5
23	7	5	7	5	7	5	7	5	7	5

End-of-Reel sentinel blocks are of twenty-four words each. Two End-of-Reel sentinel blocks must be placed after the last data block on intermediate reels of a multi-reel file (the last reel of a file will contain two End-of-File sentinel Blocks).

The block count in the first End-of-Reel sentinel block includes that block. The block count in the second End-of-Reel sentinel block includes the second. Thus, if 700 blocks of the file preceded the first End-of-Reel sentinel block, the count in that block would be 701. The count in the second would be 702.

Word 0 is a Block Descriptor word of five Fieldata "."'s (octal 75's).

Word 1 Same as word 0.

Word 2 Block Count is the number of blocks on the tape (in octal) including this block, all preceding standard and data blocks, and any blocks between Bypass sentinel blocks. This octal number is right-justified. All unused bit positions contain binary zeros.

Word 3 Tape availability is the octal number of words available on tape following this block. This octal number, right-justified, is placed in this word.

Word 4
.
. Unused
.
21

Word 22 Block Descriptor (same as word 0)

Word 23 Block Descriptor (same as word 0)

4. End-of-File Sentinel Block

WORD										
0	7	6	7	6	7	6	7	6	7	6
1	7	6	7	6	7	6	7	6	7	6
2	Number of Blocks this Tape									
3	Tape Available									
4	UNUSED									
21										
22	7	6	7	6	7	6	7	6	7	6
23	7	6	7	6	7	6	7	6	7	6

An End-of-File sentinel block is twenty-four words in length. Two End-of-File Sentinel blocks must be the last blocks on the last reel of a data file. The block count includes the End-of-File Sentinel block in which it appears. A file of 1000 blocks not counting the first End-of-File Sentinel block would have a block count of 1001 in the first End-of-File Sentinel block and 1002 in the second.

Word 0 Block Descriptor word consisting of five Fieldata "SPEC"'s (octal 76's)

Word 1 Block Descriptor (same as word 0)

Word 2 Block Count is the number of blocks in the file. This includes all standard and data blocks and any blocks between Bypass sentinel blocks.

Word 3 If tape availability is programmed, the octal number expressing it is placed in word 3, right-justified. (see page 14)

Word 4 . . . 21 Unused.

Word 22 Block Descriptor (same as word 0).

Word 23 Block Descriptor (same as word 0).

Tape availability would be of most use with a multi-file reel. Since the Input/Output File Control Subroutines do not accept such files, this would have to be programmed by the installation. For the UNISERVO* IIA, tape availability could be calculated in the following manner:

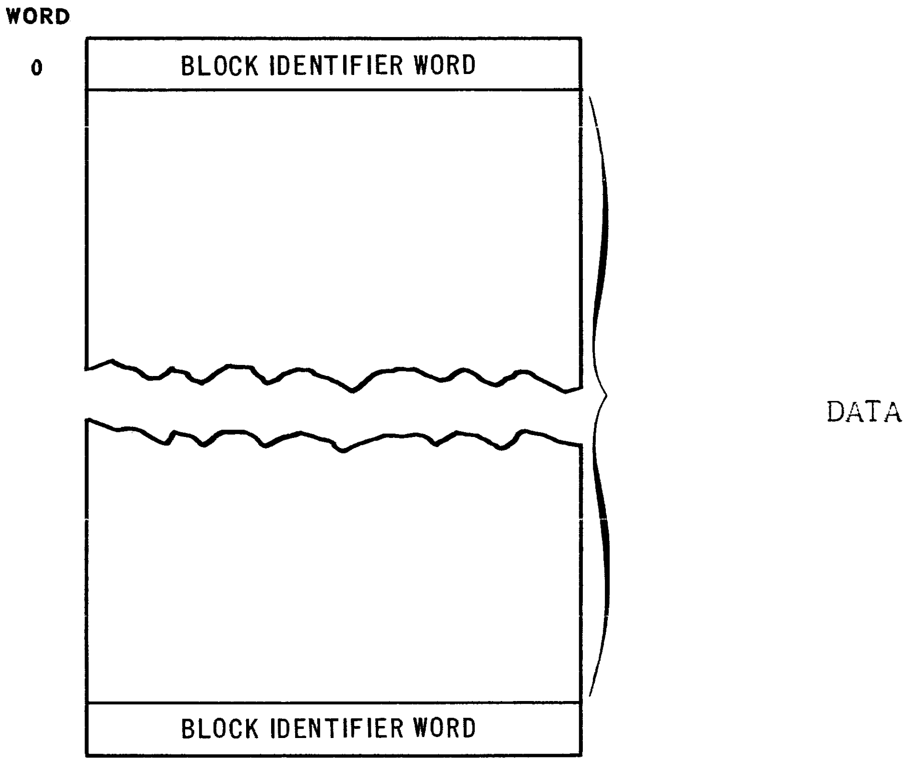
The standard Mylar reel will be assumed to have 2200 feet of usable tape (excluding bad spots). This represents approximately 1,300,000 words of information.

Tape limits can, then, be determined by calculating the block size, adding sixty (60) words for the interblock gap, and subtracting the resultant sum from the set limit of 1,300,000. The result will be the amount of tape remaining for use.

* Trademark of the Sperry Rand Corporation.

D. SEARCHABLE DATA BLOCK FORMAT

The UNISERVO IIA operation repertoire includes forward or backward Search-Read functions. If these functions are to be used, any block to be searched for must conform to the following format:



The first and last words of the block must be the same. This word must be unique to the block being searched for.

The use of a searchable data tape file presupposes the programming of a tape file control routine for this purpose by the installation. The standard tape file control routines assume non-searchable files. The conventions associated with non-searchable files preclude unique first and last words in a data block.

III. FILE CONTROL DESIGN

Any file control routine must include a File Design Block. For the UNIVAC 490 Real-Time System file control routines, such a block is in the following format:

TAPE FILE DESIGN BLOCK

	29	28	27	26	25 24 23 20	19 15	14 8	7 0
0	UPPER BUFFER LIMIT (END)						LOWER BUFFER LIMIT (BEG)	
1	ALTERNATE LOWER BUFFER LIMIT						RECORD AREA	
2	Current item number						Address of next item	
3 4 5	LABEL IDENTIFICATION							
6	y				y	d	d	d
7	REEL NUMBER							
10	TIME							
11	BLOCK SIZE							
12	RECORD SIZE							
13	i	p	f	b	# char. in label	19 Apply 17 | 16 i use 15	Input USE subroutine address	
14	q	m	r	Time	d | l | Design Type	o use	Output USE sub-routine address	
15				First char. position		# char. in data-name	Address of Label if label is data-name	
16	BLOCK LIMIT						Number of Records per Rerun	
17	BLOCK COUNT						Label of file for dump, or servo and channel if special Tape	
20	RECORD COUNT							7 CHANNEL 4 | 3 SERVO 0
21	MEMORY DUMP COUNT							7 ALT-1 CH 4 | 3 ALT-1S 0
22	RUN INDICATOR							7 ALT-2 CH 4 | 3 ALT-2S 0

CODING (unless stated, 0 = not set, 1 = set)

i - INPUT OPEN
p - OPTIONAL
f - FILE TYPE
b - BLOCK TYPE
Apply - BLOCK ADVANCE
  Ø = DEMAND
  1 = STANDBY
i use -
  Ø = NONE
  1 = Every File
  3 = Every Reel
q - OUTPUT OPEN
m - Recording Mode
r - Rerun Control this File
  Ø = NO
  1 = YES

TIME - Memory Dump at
  Ø = EOR
  1 = # records
  2 = NOW (extrinsic dump bit)
d - Dump to go on
  Ø = File
  1 = Special Tape
l - Label records
  Ø = Standard
  1 = Omitted
Design Type(17-22)
  Ø = U II Tapes
  1 = U III Tapes
  2 = HSP
  3 = Drum
  4 = Card reader
  5 = Card punch

o use (15-16)
  Ø = NONE
  1 = Every File
  3 = Every Reel

If more information is needed for file control, the block may be extended.

IV. CONSOLE PRINTER CONVENTIONS

Console Printer operations are controlled by a Console Printer Functional Subroutine which operates under REX control. The environment controlled by REX is a complex one in which several programs are requesting console printer use. Every message that is printed will be preceded by an identifying number in the form "Pxx", where xx is the program number. Typical formats would appear as:

```
P12          START OF JOB
             NR. 10576

P07          JOB 15625 COMPLETE
```

Where operator action is requested, a delay number will be assigned in the form "Dxx", where xx is the delay table number. The delay number is entered as the first field of the reply. For example, an interlock error may produce the following typeout:

```
REX   ADVISE,   P40,  CH12,   D13
```

After the cause of the error has been removed the operator may request reinitiation by the following REX input:

```
D13  □   F  Ⓢ
```

□ is a special character (octal 76) used to indicate the end of a field.

Ⓢ is a special character (octal 57) used to indicate the end of a message.

↑ is a special character (octal 77) which will erase the character preceding it from the buffer. Three such characters will erase an entire input message.

REX, in most cases, makes provision for the acceptance of 70 characters in a buffer which is provided for each program. Answers to requested information, if provided for by the REX user, should be as concise as possible.

V. ON-LINE HIGH SPEED PRINTER CONVENTIONS

The paper or form in the High Speed Printer must initially be positioned by the operator so that the first line of a page is ready to be printed. This is accomplished by aligning a first page line with the HSP scribe line which indicates the print wheel position.

The High Speed Printer Input/Output Functional Subroutine assumes such positioning as will any High Speed Printer handling routine.

VI. FLOW CHARTING AND CODING CONVENTIONS

A. Flow Charting

Flow charts should be prepared for every routine. In the case of the more complex routines, flow charts should be prepared to represent several different levels. They not only assist the reader in understanding the function of the routine but also serve as a guide when modifying the routine. The degree to which the charts may be understood is dependent upon the degree of standard notation used.

1. Block charts should be prepared for a major routine or system. They show the structure of and the interrelationships between the significant parts. Depending upon the size and complexity of the routine, no block chart or a series of block charts at different levels of detail may be required.

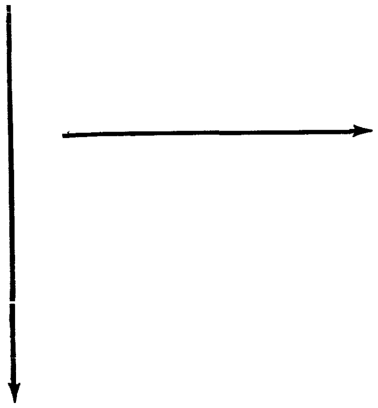
2. Flow charts should be prepared to show the detailed processing sequence within a routine; however, a flow chart should never be so detailed as to relate directly to machine instructions at every step. Operations on a flow chart should be self-explanatory. English statements should be used in decision and in operation boxes in preference to symbolic statements.

3. If a major routine uses a subroutine that has been completely documented, the detailed flow chart of the subroutine should be omitted. Only the name of the subroutine should be indicated.

4. Flow charts should be related to the coding. The machine address or source code tag, label, sequence or operation number should be shown.

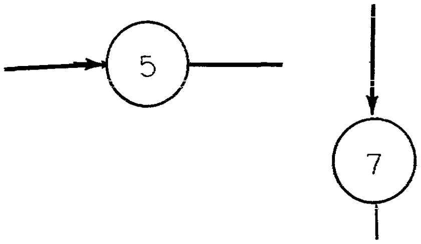
   The following symbols are considered as standard:

   a. Lines of Flow:



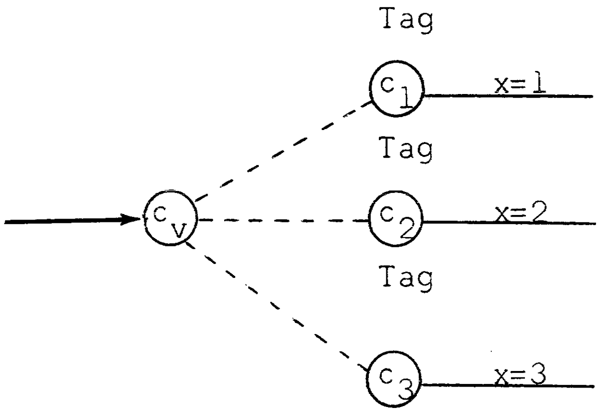
Indicate the direction of flow, Flow paths, in general, should be from top to bottom or from left to right of the page. To avoid the use of too many connectors or to allow for a less complicated chart design, flow may be from bottom to top or right to left of page. Arrows are required in the latter case. The crossing of flow lines and the parallel running of a number of flow lines should be avoided.

b. Connector:



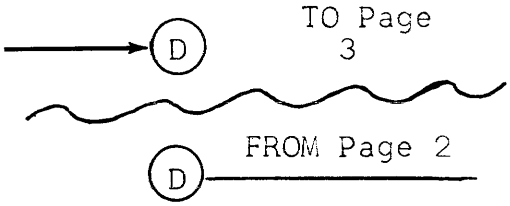
Indicates the juncture of two or more flow paths. If connectors are used for diagram clarity they should be numbered. If they represent actual points in the program the appropriate tag should be shown above the connector.

c. Variable Connector:



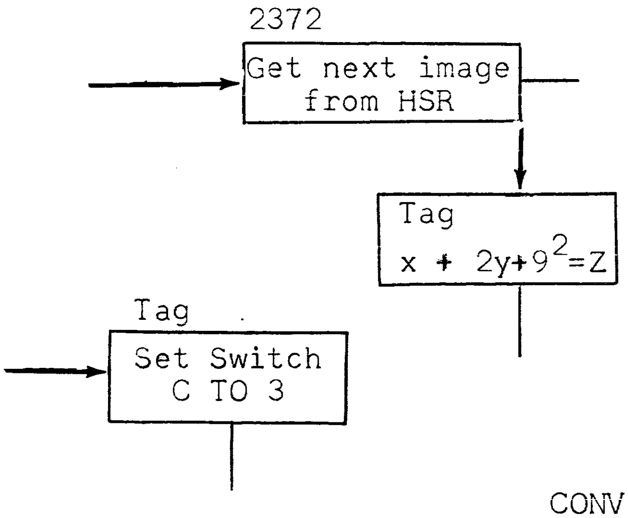
Indicates branching or switching of the flow path. A dotted line must connect the ending connector with each of the variable connectors. Each of the variable connectors must also be labeled to denote the case in which it is used. The notation used within the connectors should be alphabetic; the ending connector containing the variable connectors containing a numeric subscript.

d. Remote Connector:



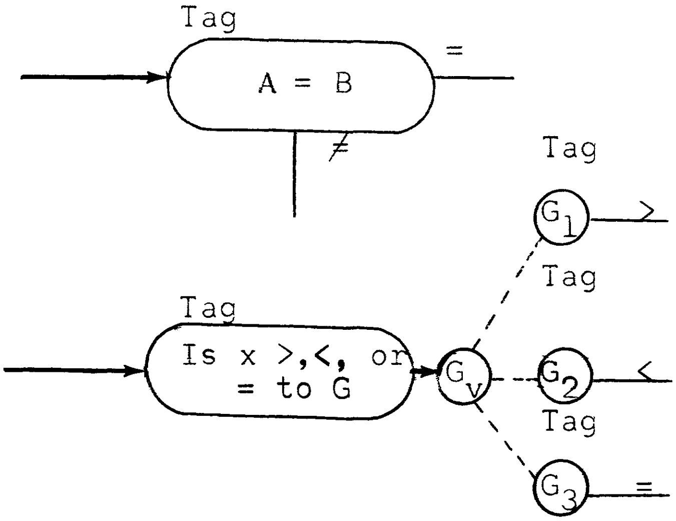
Used when the flow path must be broken because of space (page) limitations. If remote connectors are on different pages of a chart, TO and FROM page numbers should be written beside the connectors.

e. Operation:



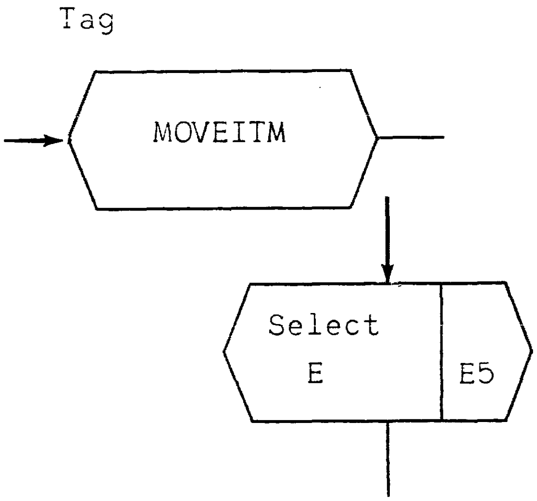
Describes an operation on data, an operation on a program's control mechanism,or a mathematical function. An English statement or mathematical equation should be enclosed in the rectangle. An alphanumeric tag or some representation of a line of coding should be shown above the operation to facilitate reference to the coding.

f. Decision:



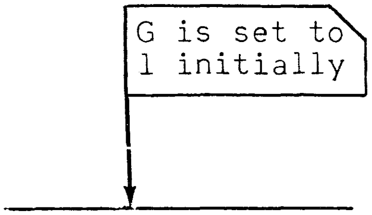
Indicates a point in a program where a choice is made regarding alternative paths of processing or computation. Only one line of flow can enter this symbol. For a two-way decision, two lines must leave this symbol. For a multiple decision, only one line leaves the sumbol and it must lead directly to a variable connector. Each exit path must be labeled to indicate the condition which exists when the path is used. A tag or number must be shown above the oval to facilitate reference to the coding.

g. Subroutine:



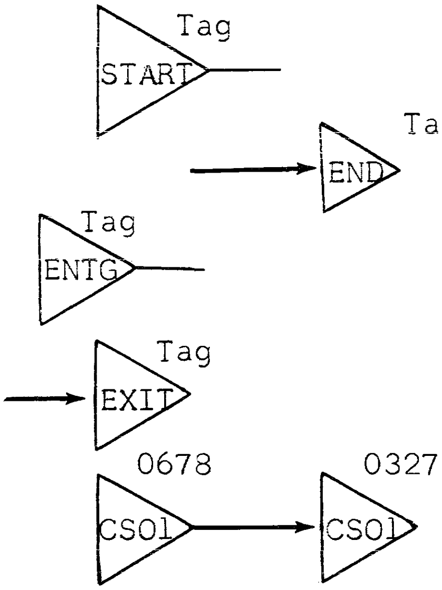
Indicates the execution of a separately defined subroutine at a point in the flow. If the subroutine is part of a library and has its own documentation, only the name of the subroutine is enclosed in the hexagon. If the subroutine is part of this program and is defined elsewhere in the documentation, the name of the subroutine (or the function it performs) and the tag which identifies the subroutine should be enclosed in the hexagon. A vertical or horizontal line should separate the two with the tag at the right or upper portion of the symbol. A tag or number must be shown above the symbol to indicate the location in the coding where entry occurs.

h. Assertion:



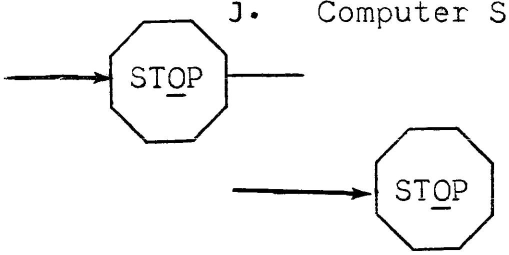
Used to assert or explain the existence of certain conditions at some point in the flow path. Upper right hand corner should be slanted.

i. Terminal Connectors:



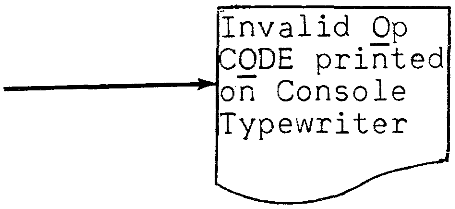
Indicate the start and end (where the end is not a computer stop) of a program or the entry and exit points of a subroutine. For a program or run, the triangle should contain the words "Start" and "End". For a Library Subroutine, the symbol should contain the name of the subroutine. For a subroutine within the program being documented, the symbol should contain the words "Enter" and "Exit" for each entry and exit. In all cases a tag or appropriate notation should be shown above the symbol to facilitate cross reference to the coding. Both symbols should point in the direction of flow.

j. Computer Stop:



Indicates those points in the program where the computer comes to a stop and which require manual intervention for restarting. The word "STOP" should be enclosed in the octagon.

k. Input or Output:



Used to describe input or output information. The specific type of hardware unit should be named within the symbol.

B. Coding

It is a minimal requirement that a listing or edited listing be prepared for every program. These listings serve several purposes: they enable the user to obtain a detailed knowledge of the routine, provide a means of checking the program deck or tape, may be used when modifying the routine, and may be used as a source to prepare program decks or tapes when copying facilities are not available.

Whenever possible, these listings should be printed directly from the media (e.g. cards, tape) on which the coding is stored for computer use. If listings are produced as a byproduct of assembly, this form is preferable. All coding should be annotated. The comments should indicate the tasks performed, not describe the instructions used at each step in the routine.

ADDENDA TO:

"SYSTEMS CONVENTIONS, Programmer's Reference Manual", UP 2572.

Correction to Page 11

Page 11, change word description for words 2 through 21 to read as follows:

Word 2 Contains the beginning and ending core addresses of the dump.

Word 3 and 4 This is the same as the program identification typed out by the REX dump routine

Word 5 . . . 21 } For rerun information created by the rerun routine. Not available for any other use.

UNIVAC®

DIVISION OF SPERRY RAND CORPORATION

PRINTED IN U.S.A. UP 2572