BASIC CONTROL SYSTEM

HEWLETT hp PACKARD

11000 Wolfe Road
Cupertino, California 95014

HP 02116-9017

December 1971

First edition, February, 1968
Revised, December, 1971

Printed in the U.S.A.

Second Edition

PREFACE

This publication is a reference manual for the programmer using the HP 2100 series Basic Control System (BCS).

It is assumed that the reader is familiar with the HP 2100 family of computers and the Hewlett-Packard standard software systems. Completion of the Hewlett-Packard programming training courses or equivalent experience in programming is a prerequisite to using this manual.

Other software manuals of interest include:

HP ASSEMBLER (02116-9014)
HP ALGOL (02116-9072)
HP FORTRAN (02116-9015)
FORTRAN IV REFERENCE MANUAL (5951-1321)
RELOCATABLE SUBROUTINES (02116-91780)
MAGNETIC TAPE SYSTEM (02116-91752)

This manual contains programming information for BCS; the necessary operating information can be found in the "BASIC CONTROL SYSTEM" module (5951-1391) of the *SOFTWARE OPERATING PROCEDURES*.

NEW AND CHANGED INFORMATION

All known errors have been corrected in this edition. Also, operating procedures have been removed and can now be found in the "BASIC CONTROL SYSTEM" (5951-1391) module of the *SOFTWARE OPERATING PROCEDURES*. Operating procedures for BCS within the Magnetic Tape System are covered in *MAGNETIC TAPE SYSTEM* (02116-91752).

CONTENTS

SECTION II (cont.)

INPUT/OUTPUT REQUESTS

SECTION II (cont.)

INPUT/OUTPUT REQUESTS

SECTION II (cont.)

INPUT/OUTPUT REQUESTS

RELOCATING LOADER

INPUT/OUTPUT DRIVERS

SECTION IV (cont.)

INPUT/OUTPUT DRIVERS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION I
GENERAL DESCRIPTION

The Basic Control System (BCS) provides an efficient loading and input/output control capability for relocatable programs produced by the HP Assembler, HP FORTRAN, FORTRAN IV, or HP ALGOL. BCS is modular in design and is constructed to fit each user's hardware configuration.

The Basic Control System performs the following functions:

- Loads and links relocatable programs
- Creates indirect and base page addressing when necessary
- Selects and loads referenced library routines
- Processes I/O requests and services I/O interrupts

The Basic Control System is comprised of two distinct parts: input/output subroutines and the Relocating Loader. Associated with the Basic Control System are two other systems: Prepare Control System and the Debugging System.

The Relocating Loader loads and links relocatable object programs generated by the Assembler, FORTRAN, and ALGOL. It also links the object programs with the input/output subroutines and any library subroutines referred to in the programs. The Prepare Control System adapts the Basic Control System program to a particular hardware configuration. The Debugging System is a relocatable program that BCS loads after the object program(s); with the debugging program, the programmer can find errors in his program.

The minimum equipment configuration required for the Basic Control System (and Prepare Control System) is as follows:

2100 family computer with 4K memory

Teleprinter

INPUT/OUTPUT SUBROUTINES

The input/output package consists of an Input/Output Control subroutine and driver subroutines for the peripheral devices. Input/output operations are specified as symbolic calling sequences in assembly language. These requests are translated into object code calls to the I/O Control subroutine. The subroutine interprets the call and directs the request to the proper driver. The driver initiates the operation and returns control to the calling program. Whenever an interrupt occurs, the driver temporarily resumes control to transfer the next element of data. When the operation is completed, the I/O Control subroutine makes the status of the operation available for checking by the program.

The input/output package allows device-independent programming; a device is specified in terms of a unit-reference number rather than a channel number or select code. Furthermore, the user need not be concerned about how data is transmitted (by bit, by character, etc.), he need only specify the number of words or characters and the location in memory where the data is stored.

RELOCATING LOADER

The Relocating Loader loads object code programs produced by the Assembler, FORTRAN and ALGOL. The linking capability of the loader allows the user to divide a program into several subprograms, to assemble and test each separately, and finally to execute all as one program. Object subprograms produced by the Assembler may be combined with object subprograms produced by FORTRAN and ALGOL. The subprograms are linked through symbolic entry points and external references.

The loader also provides indirect addressing whenever an operand of an instruction does not fall in the same page as that into which the instruction is being loaded. This allows a program to be designed without concern for page boundaries.

An optional feature of the loader allows the user to obtain an absolute dump of a relocatable program plus the Basic Control System and those library subroutines that were referenced by the program. The process of generating the absolute program is such that instructions (not just common storage) may be allocated to the area normally occupied by the loader. This feature may also be utilized for a program which has reached "production" status; absolute format requires less loading time because an absolute program is loaded by the Basic Binary Loader.

The following information is relevant to the Relocating Loader used in core memory greater than 4K:

a. When the Relocating Loader is *not* requested to produce an absolute version of a program, it sets all unused locations in memory to 106055$_8$ (a unique halt instruction) so that a halt will occur if any should be executed. This is useful for detecting errors in programs.

b. A certain portion of the BCS Relocating Loader must always be resident in core while the BCS is in use. This portion of the Relocating Loader contains a segment labled HALT, which is used by the new version of the .STOP routine in the Relocatable Library. The final halt instruction for the BCS is directly associated with this entry point for use in one of two ways. The final halt instruction remains unchanged if paper tape operation is used, but it *is* changed to JSB 00106$_8$, I (a call to the Inter-Pass Loader of the Magnetic Tape System) if the BCS is run using MTS.

For further information on the BCS and its relation to the Magnetic Tape System see the *Magnetic Tape System* manual, HP 02116-91752.

PREPARE CONTROL SYSTEM

Prepare Control System is a special purpose program which produces an absolute version of the Basic Control System from relocatable BCS subprograms. During the construction of the absolute BCS, the user also establishes the relationships among I/O channel numbers, drivers, interrupt entry points in the drivers, and unit-reference numbers. Prepare Control System is used when the configuration of the hardware is defined initially or whenever there is a modification or expansion to the configuration.

DEBUGGING SYSTEM

The debugging routine provides aids in program testing. Options provided by the routine will print selected areas of memory, trace portions of the program during execution, modify the contents of selected areas in memory, modify simulated computer registers, halt execution of the program at specified breakpoints, and initiate execution at any point in the program.

SECTION II
INPUT/OUTPUT REQUESTS

The Basic Control System provides the facility to request input/output operations in the form of five-word calling sequences in assembly language. BCS interprets the call, initiates the operation, and returns control to the calling program. When the data transfer is complete, the system provides status information which may be checked by the program. Interrupts that occur during or on termination of the transfer are processed entirely by the system; interrupt handling subroutines are not required in the user's program.

GENERAL CALLING SEQUENCE

The general form of the input/output request is:

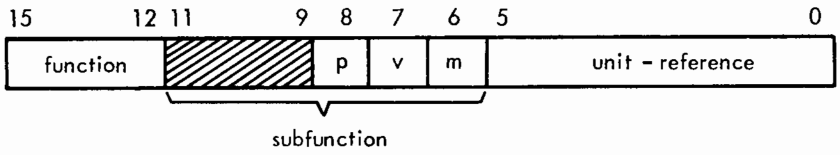
```
        EXT         .IOC.
         .
         .
         .
        JSB         .IOC.

        OCT         <function><subfunction><unit-reference>

        JSB  }
        JMP  }      reject address <error return>

        DEF         buffer address

        DEC  }
        OCT  }      buffer length

        <normal return>
```

Input/Output Subroutine (.IOC.)

.IOC. is the symbolic entry point name of the Input/Output Control subroutine within the Basic Control System. All input/output operations are requested by performing a jump subroutine (JSB) to this entry point. The input/output control subroutine returns control to the calling program at the first location following the last word of the calling sequence. Programs referring to .IOC. must declare it as an external symbol.

Function, Subfunction, and Unit-Reference

The second word of the request determines the function to be performed and the unit of equipment for which the action is to be taken. In assembly language, this information may be supplied in the form of an octal constant. The bit combinations of the constant are as follows:



FUNCTION

The function (bits 15-12) indicates the basic read/write operation:

Function Name	Code (octal)
Read	01
Write	02

SUBFUNCTION

The subfunction (bits 11-6) defines the options for certain read/write operations:

p = 1 Print input: The ASCII data read from the teleprinter is printed as it is received.

v = 1 Variable length binary input: The value in bits 15-8 of the first word on an input paper tape indicates the length of the record (including the first word). If the value exceeds the length of the buffer, only the number of words specified as the buffer length are read. If v = 0, the buffer length field always determines the length of record to be transmitted. If the device does not read paper tape, this parameter is ignored.

m = 1	Mode: The data is transmitted in binary form exactly as it appears in memory or on the external device. If m = 0, the data is transmitted in ASCII or BCD format.

UNIT-REFERENCE

The value specified for the unit-reference field indicates the unit of equipment on which the operation is to be performed. The number may represent a standard unit assignment or an installation unit assignment. Standard unit numbers are as follows:

Number	Name	Usual Equipment Type
1	Keyboard Input	Teleprinter
2	Teleprinter Output	Teleprinter
3	Relocatable Library	Punched Tape Reader
4	Punch Output	Tape Punch
5	Input	Punched Tape Reader
6	List Output	Teleprinter

Installation unit numbers may be in the range 7_8 - 74_8 with the largest value being determined by the number of units of equipment available at the installation. The particular physical unit referenced depends on the manner in which equipment is defined within the Basic Control System by the installation.

When the Basic Control System configuration is established, an equipment table (EQT) is created. This table defines the type of equipment (teleprinter, magnetic tape, etc.), the channel on which each unit is connected, and other related details. The ordinal of the unit's entry in this table is the value specified as a unit-reference number for an installation unit. Since numbers 1-6 are reserved as standard unit numbers, the first unit described in the table is referred to by the number 7_8; the second, 10_8; the third, 11_8; and so forth. The entries for one possible equipment table configuration might establish the following relationships:

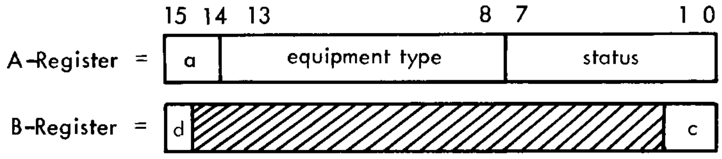
Installation unit number (ordinal)	Device	I/O Channel
7	Teleprinter	12 or 12 and 13
10	Punched Tape Reader	10
11	Tape Punch	11

The standard unit numbers are associated with physical equipment via a standard equipment table (SQT) and EQT. The SQT is a list of references to the EQT. SQT is also created by the installation when the BCS configuration is established. Each standard unit may be a separate device, or a single device may be accessed by several standard unit numbers as well as an installation unit number. (For complete details on the SQT and EQT, see Appendices B and C.)

Reject Address

The content of the third word of the calling sequence is normally a JSB or a JMP to a reject address which is the start of a user subroutine designed to determine the cause of a reject and take appropriate action.

The Basic Control System transfers control to this address if the input/output operation can not be performed. On transfer, the system provides status information that may be checked by the user's program.



The contents of the A-Register indicate the physical status of the equipment. (See STATUS REQUEST.)
The contents of the B-Register indicate the cause of the reject (bits 14-1 are zeros):

d = 1 The device or driver subroutine is busy and therefore unavailable, or, for the Kennedy 1406 Tape unit, a broken tape condition encountered.

c = 1 A Direct Memory Access channel is not available to operate the device.

d = c = 0 The function or subfunction selected is not legal for the device.

Buffer Storage Area

The buffer address is the location of the first word of data to be written on an output device or the first word of a block reserved for storage of data read from an input device. The length of the buffer area may be specified in terms of words or characters. If the length is given as words, the value in the buffer length field must be a positive integer; if it is given as characters, it must be a negative integer.

In addition to describing the buffer area in the calling sequence, the area must also be specifically defined in the assembly language program, usually with a BSS or COM pseudo instruction.

ERROR CONDITIONS DURING EXECUTION

(HLT 102076 WITH IOC 24173A)

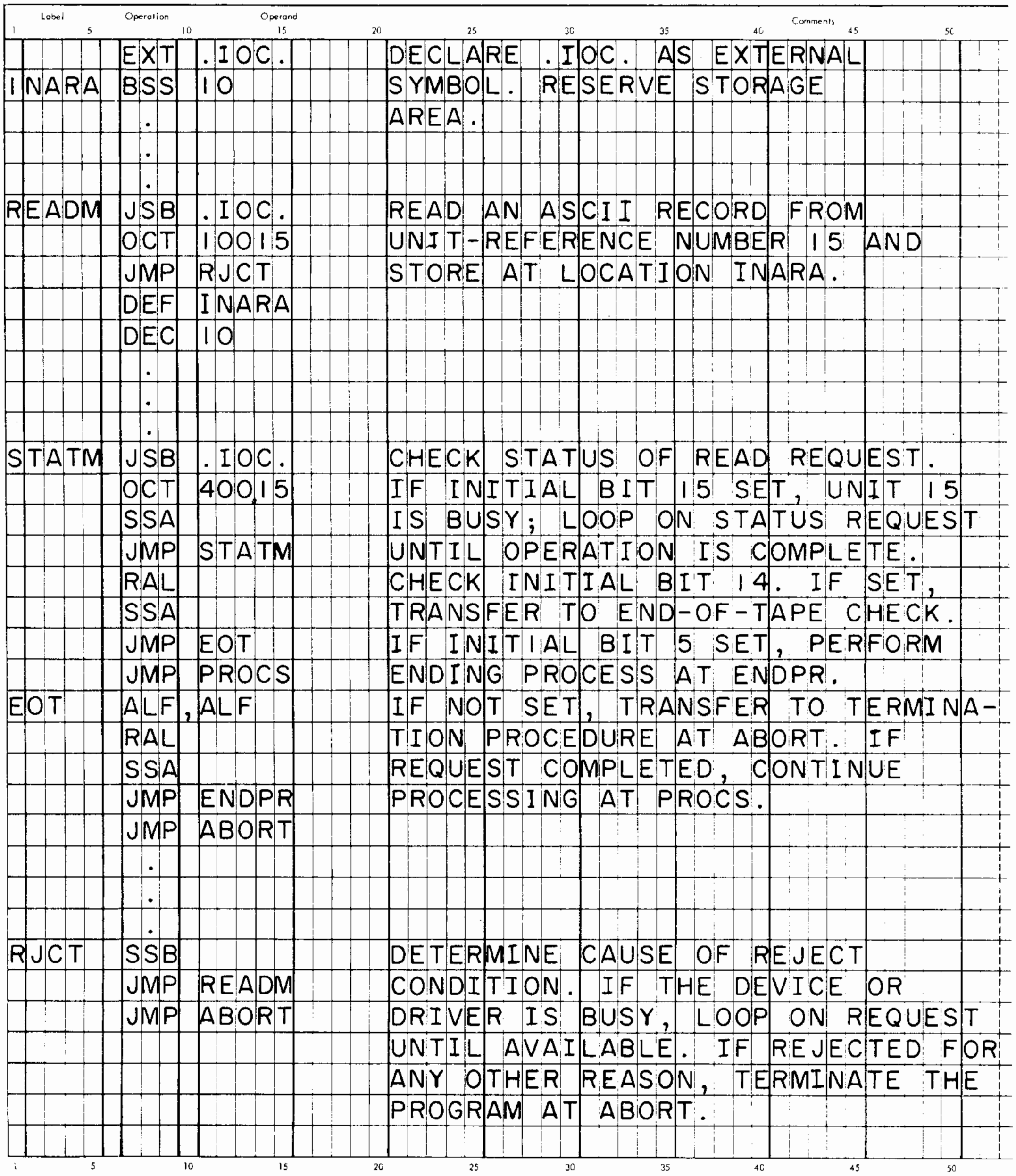
Illegal conditions encountered during .IOC. request processing are termed irrecoverable and cause a halt. (The halt is at the absolute location assigned to the symbol IOERR during Prepare Control System processing.) Diagnostic information is displayed in the A- and B-Registers at the time of the halt.

The B-Register contains the absolute location of the JSB instruction of the request call containing the illegal condition.

The A-Register contains a code defining the illegal condition:

A-Register	Explanation
000000	Illegal request code.
000001	Illegal unit-reference number in request.
000002	The Standard unit requested is not defined as a particular device in the Equipment Table.

Examples:


Label
Operation
Operand
Comments

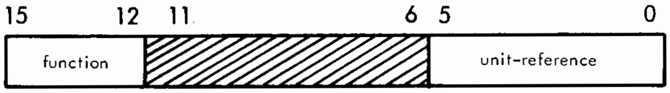
```
      EXT  .IOC.     DECLARE .IOC. AS EXTERNAL
INARA BSS  10        SYMBOL. RESERVE STORAGE
      .              AREA.
      .
      .
READM JSB  .IOC.     READ AN ASCII RECORD FROM
      OCT  10015     UNIT-REFERENCE NUMBER 15 AND
      JMP  RJCT      STORE AT LOCATION INARA.
      DEF  INARA
      DEC  10
      .
      .
      .
STATM JSB  .IOC.     CHECK STATUS OF READ REQUEST.
      OCT  40015     IF INITIAL BIT 15 SET, UNIT 15
      SSA            IS BUSY; LOOP ON STATUS REQUEST
      JMP  STATM     UNTIL OPERATION IS COMPLETE.
      RAL            CHECK INITIAL BIT 14. IF SET,
      SSA            TRANSFER TO END-OF-TAPE CHECK.
      JMP  EOT       IF INITIAL BIT 5 SET, PERFORM
      JMP  PROCS     ENDING PROCESS AT ENDPR.
EOT   ALF,ALF        IF NOT SET, TRANSFER TO TERMINA-
      RAL            TION PROCEDURE AT ABORT. IF
      SSA            REQUEST COMPLETED, CONTINUE
      JMP  ENDPR     PROCESSING AT PROCS.
      JMP  ABORT
      .
      .
      .
RJCT  SSB            DETERMINE CAUSE OF REJECT
      JMP  READM     CONDITION. IF THE DEVICE OR
      JMP  ABORT     DRIVER IS BUSY, LOOP ON REQUEST
                     UNTIL AVAILABLE. IF REJECTED FOR
                     ANY OTHER REASON, TERMINATE THE
                     PROGRAM AT ABORT.
```

CLEAR REQUEST

The CLEAR request can be used to terminate a previously issued input or output operation before all data is transmitted. It has the following form:

```
        EXT       .IOC.
         .
         .
         .
        JSB       .IOC.
        OCT       <function><unit-reference>
        <normal return>
```

The second word consists of the following:

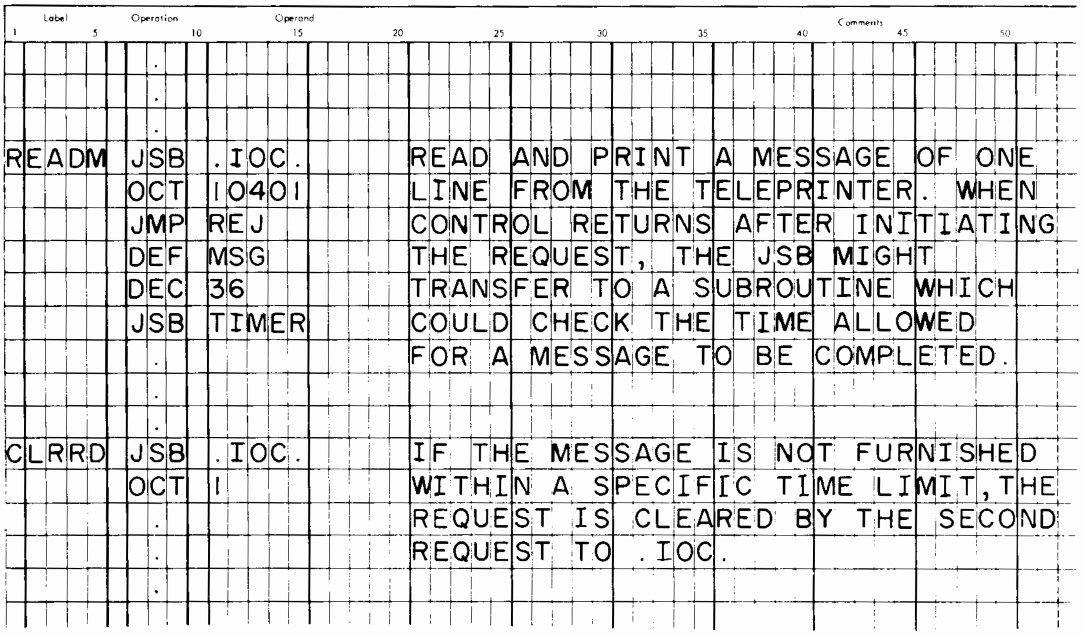


The function has the following value:

Function Name	Code (octal)
CLEAR	00

The only other parameter required is the unit-reference number. If the unit-reference number is specified as 00 (i.e., the second word of the calling sequence is OCT 0), all previous input and output operations are terminated. This request, the system CLEAR request, makes all devices available for the initiation of a new operation. On return from a system CLEAR request, the contents of the A- and B-Registers are meaningless.

Example:


Label
Operation
Operand
Comments
1
5
10
15
20
25
30
35
40
45
50

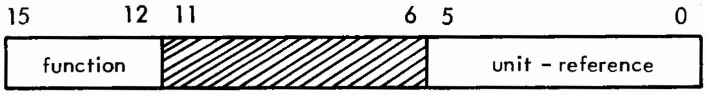
```
      .
      .
      .
READM JSB .IOC.     READ AND PRINT A MESSAGE OF ONE
      OCT 10401     LINE FROM THE TELEPRINTER. WHEN
      JMP REJ       CONTROL RETURNS AFTER INITIATING
      DEF MSG       THE REQUEST, THE JSB MIGHT
      DEC 36        TRANSFER TO A SUBROUTINE WHICH
      JSB TIMER     COULD CHECK THE TIME ALLOWED
      .             FOR A MESSAGE TO BE COMPLETED.
      .
      .
CLRRD JSB .IOC.     IF THE MESSAGE IS NOT FURNISHED
      OCT 1         WITHIN A SPECIFIC TIME LIMIT,THE
      .             REQUEST IS CLEARED BY THE SECOND
      .             REQUEST TO .IOC.
      .
```

STATUS REQUEST

A request can be directed to .IOC. to determine the status of a previous input/output request or to determine the physical status of one or all units of equipment. The request has the following form:

```
JSB       .IOC.
OCT       <function><unit-reference>
<normal return>
```

The second word of the request has the following form:

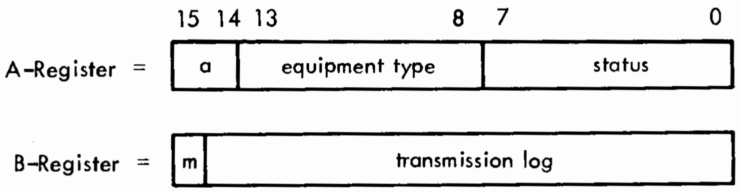


The function has the following value:

Function Name	Code (octal)
STATUS	04

The calling sequence requires no other parameters. A reject location is not necessary since the status information is always available. If the unit-reference number is specified as 00 (i.e., the second word on the calling sequence is OCT 40000), the request is interpreted as a system request.

If information is requested for a single unit, the Basic Control System returns to the location immediately following the request with the status information in the A- and B-Registers:



a = Availability of device:

0 The device is available; the previous operation is complete.

1 The device is available; the previous operation is complete; but a transmission error has been detected.

2 The device is not available for another request; the operation is in progress.

Equipment Type = This field contains a 6-bit code that identifies the device referenced:

00-07	Paper Tape devices	
	00	2752A Teleprinter
	01	2737A Punched Tape Reader
	02	2753A Tape Punch
10-17	Unit Record devices	
	15	Mark Sense Reader
20-37	Magnetic Tape and Mass Storage devices	
	20	Kennedy 1406 Incremental Tape Transport
	21	HP 2020 Magnetic Tape Unit
	22	HP 3030 Magnetic Tape Unit
40-77	Instrumentation devices	
	40	Data Source Interface
	41	DVM Programmer
	42	Scanner Programmer
	43	Time Base Generator

Status = The status field indicates the actual status of the device when the data transmission is complete. The contents depend on the type of device referenced:

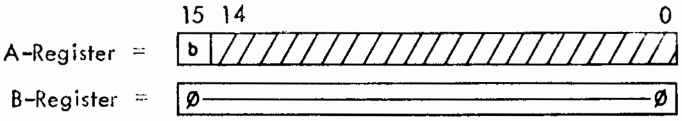
Device	Bits 7-0	Condition
Teleprinter reader or Punched Tape Reader	*xx*1*xxxxx*	End-of-Tape (10 Feed Frames)
Tape Punch	*xx*1*xxxxx*	Tape supply low
Kennedy 1406 Incremental Tape Transport	*xx*1*xxxxx*	End-of-Tape mark sensed
	*xxxx*1*xxx*	Broken tape; no tape on write head
	*xxxxxxx*1	Device busy
HP 2020 and 3030 Magnetic Tape Units =	1*xxxxxxx*	End-of-file record (17_8 for 2020, 23_8 for 3030) is detected or written.
	*x*1*xxxxxx*	Start-of-tape marker sensed
	*xx*1*xxxxx*	End-of-tape marker sensed
	*xxx*1*xxxx*	Timing error on read/write
	*xxxx*1*xxx*	I/O request rejected: a. tape motion required but controller busy b. backward tape motion required but tape at load point c. write request given but reel does not have write enable ring.
	*xxxxx*1*xx*	Reel does not have write enable ring or tape unit is rewinding.
	*xxxxxx*1*x*	Parity error on read/write
	*xxxxxxx*1	Unit busy or in LOCAL mode.

m = This bit defines the mode of the data transmission:

0	ASCII or BCD
1	Binary

Transmission Log = This field is a log of the number of characters or words transmitted. The value is given as a positive integer and indicates characters or words as specified in the calling sequence. The value is stored in this field only when the request is completed; that is, when all data is transmitted or when a transmission error is detected.

If a system status request is made, the information in the A- and B-Registers is as follows:



b =	System Status
0	No device is busy
1	At least one device is busy

PAPER TAPE SYSTEM

RECORD FORMATS

The Paper Tape System operates on ASCII and binary records.

ASCII Records

An ASCII record is a group of characters terminated by an end-of-record mark which consists of a carriage return, CR, and a line feed, LF. If an odd number of characters is input, the last word transmitted to the buffer is padded with an ASCII blank.

For an input operation, the length of the record transmitted to the buffer is the number of characters or words designated in the request, or less if an end-of-record mark is encountered before the character or word count is exhausted. The codes for CR and LF are not transmitted to the buffer. An end-of-record mark preceding the first data character is ignored.

For an output operation, the length of the record is determined by the number of characters or words designated in the request. An end-of-record mark is supplied at the end of each output record by the input/output system.

If the last character of an ASCII output record to the teleprinter or punch is ←, however, the end-of-record mark is omitted. This allows control of teleprinter line spacing. For example, the user may write a message (the ← is not printed) and expect the reply to be typed on the same line. The reply must be terminated with the CR and LF.

If a "Rub Out" code followed by a CR, LF is encountered on input from the teleprinter or Paper Tape Reader, the current record is ignored (deleted) and the next record transmitted. ("Rub out" appears on the teleprinter keyboard and is synonymous with the ASCII symbol DEL.)

If less than ten feed frames (all zeros) are encountered before the first data character from a paper tape input device, they are ignored. Ten feed frames are interpreted as an end-of-tape condition.

Binary Records

A binary record is transmitted exactly as it appears in memory or on an 8-level paper tape. The record length is specified by the number of characters or words designated in the request. The first character of a binary record must be nonzero. On input operations, less than ten feed frames preceding the first data character are ignored. Ten feed frames are interpreted as an end-of-tape condition. On output, the system writes four feed frames to serve as a physical record separator.

Binary input records may vary in length (up to 255 words). The first word of the record contains a number in bits 15-8 specifying the length of the record (including the first word). The entire record including the word count is transmitted to the buffer. If the actual length exceeds the size of the buffer, only the number of words equivalent to the buffer length is transmitted.

> *NOTE: Although binary transmission is normally stated in words as opposed to characters, if an odd number of characters is requested on input the last word transmitted to the buffer is padded with binary zeros.*

CALLING SEQUENCE

```
EXT       .IOC.
 .
 .
 .
JSB       .IOC.
OCT       <function><subfunction><unit-reference>
JSB }
JMP }    <reject address><error return>
DEF      <buffer address>
DEC }
OCT }    <buffer length>
<normal return>
```

Function and Subfunction Codes

Allowable combinations of function and subfunction codes are as follows:

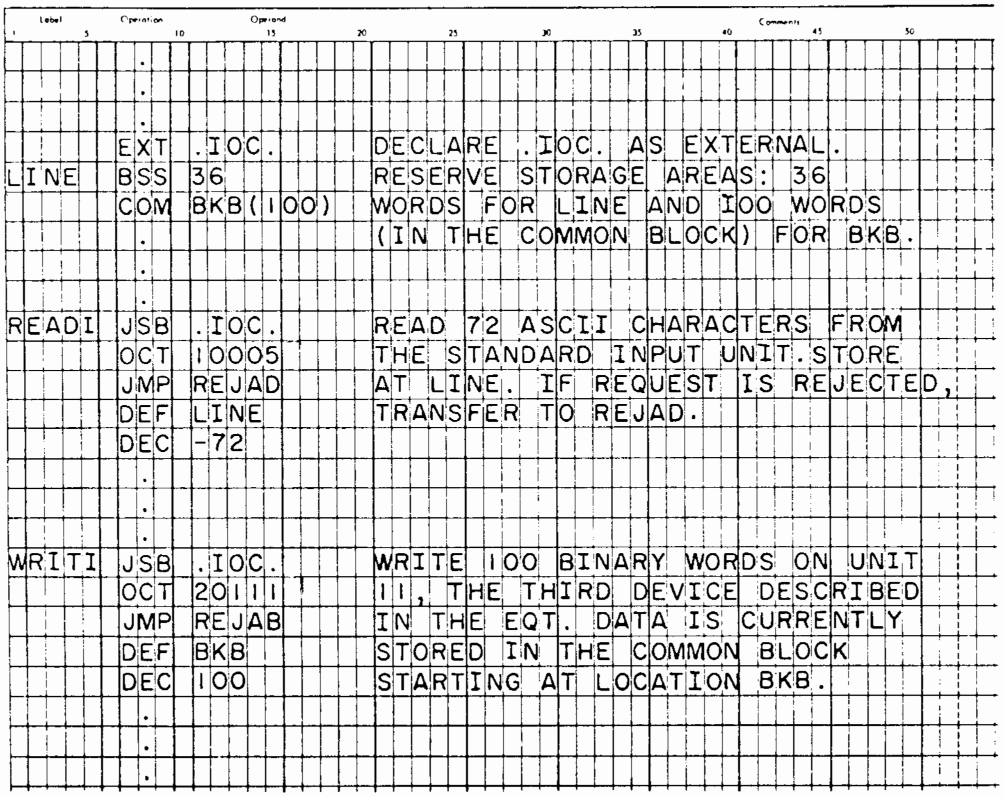
Operation	Octal value of Bits 15-6
Read ASCII record	0100
Read ASCII record and print	0104
Read binary record	0101
Read variable length binary record	0103
Write ASCII or BCD record	0200
Write binary record	0201

All illegal combination of codes is rejected.

Buffer Length

Character or word transmission may be specified for any paper tape device. The buffer length for data that may be printed on the teleprinter should be no more than 72 characters (36 words) or else the teleprinter will overprint at the end of line.

Examples:


Label
Operation
Operand
Comments

```
       .
       .
       .
       EXT .IOC.    DECLARE .IOC. AS EXTERNAL.
LINE   BSS 36       RESERVE STORAGE AREAS: 36
       COM BKB(100) WORDS FOR LINE AND 100 WORDS
       .            (IN THE COMMON BLOCK) FOR BKB.
       .
       .
READI  JSB .IOC.    READ 72 ASCII CHARACTERS FROM
       OCT 10005    THE STANDARD INPUT UNIT.STORE
       JMP REJAD    AT LINE. IF REQUEST IS REJECTED,
       DEF LINE     TRANSFER TO REJAD.
       DEC -72
       .
       .
       .
WRITI  JSB .IOC.    WRITE 100 BINARY WORDS ON UNIT
       OCT 20111    11, THE THIRD DEVICE DESCRIBED
       JMP REJAB    IN THE EQT. DATA IS CURRENTLY
       DEF BKB      STORED IN THE COMMON BLOCK
       DEC 100      STARTING AT LOCATION BKB.
       .
       .
       .
```

NOTE: In READI and WRITI, the leading 0 of the second word of the calling sequence need not be written in the source language since it is supplied in the object code as a result of using the OCT pseudo instruction.

HP 2891A CARD READER

DATA FORMATS

The HP 2891A Card Reader driver (D.11) provides three card reading functions to read any type of punched card, as described in the following paragraphs.

Hollerith to ASCII (Octal Equivalents) Conversion

Hollerith characters are converted to ASCII octal equivalents which are then placed into a buffer word according to the character's column number. All characters in odd-numbered columns are placed into the left byte (bits 15-8), and those in even-numbered columns are placed into the right byte (bits 7-0). Table 2-1 shows how the octal equivalent of each character appears in the two possible positions within a buffer word. Consecutive characters (including blanks) are placed into consecutive buffer characters. (See Figure 2-1.)

The function code 0100 (READ HOLLERITH TO ASCII CONVERSION) reads a card containing "ASMB" in columns 1 through 4:

"A" = 101_8 which appears as 404_8 in "offset-octal" bits 15 - 8 of the first buffer word:

```
 15      12       9       6       3       0
[0  ■0 0  0 0 0  ■                         ]
```

"S" = 123_8 in "true-octal" bits 7 - 0 of the first buffer word:

```
 15      12       9       6       3       0
[                        0■  0■0  0■■]
```

Thus the first packed word of the buffer is:

```
 15      12       9       6       3       0
[0  ■0 0  0 0 0  ■0■  0■0  0■■]
```

"M" = 115_8 which appears as 464_8 in "offset-octal" bits 15 - 8 of the second buffer word, and "B" = 102_8 in "true-octal" bits 7 - 0 of the second buffer word:

```
 15      12       9       6       3       0
[0  ■0 0  ■■0  ■0■  000  0■0]
```

NOTE: Bits 8, 7, and 6 contain the octal sum of the least significant digit in the "offset-octal" value in bits 15 - 8 and the most significant digit in the "true-octal" value in bits 7 - 0.

Figure 2-1. The READ HOLLERITH TO ASCII Function.

Table 2-1

Hollerith to ASCII Octal Equivalents

Character		ASCII Octal Equivalent	
HOLLERITH	ASCII	Bits 15-8 (offset)	Bits 7-0 (true)
A	A	404	101
B	B	410	102
C	C	414	103
D	D	420	104
E	E	424	105
F	F	430	106
G	G	434	107
H	H	440	110
I	I	444	111
J	J	450	112
K	K	454	113
L	L	460	114
M	M	464	115
N	N	470	116
O	O	474	117
P	P	500	120
Q	Q	504	121
R	R	510	122
S	S	514	123
T	T	520	124
U	U	524	125
V	V	530	126
W	W	534	127
X	X	540	130
Y	Y	544	131
Z	Z	550	132
0	0	300	060
1	1	304	061
2	2	310	062
3	3	314	063
4	4	320	064
5	5	324	065

Character		ASCII Octal Equivalent	
HOLLERITH	ASCII	Bits 15-8 (offset)	Bits 7-0 (true)
6	6	330	066
7	7	334	067
8	8	340	070
9	9	344	071
(space)		200	040
!	!	204	041
" *(quote)*	"	210	042
#	#	214	043
$	$	220	044
%	%	224	045
&	&	230	046
' *(apostrophe)*	'	234	047
(	(	240	050
)	)	244	051
*	*	250	052
+	+	254	053
, *(comma)*	,	260	054
- *(hyphen or minus)*	-	264	055
. *(period)*	.	270	056
/	/	274	057
:	:	350	072
;	;	354	073
<	<	360	074
=	=	364	075
>	>	370	076
?	?	374	077
@	@	400	100
¢ *(cent)*	[	554	133
¬ *(not mark)*	]	564	135
¦ *(vertical bar*)*	↑	570	136
_ *(underscore**)*	←	574	137
0-8-2	\	560	134

*NUMERIC Y

**NUMERIC W

Packed Binary

The Read Packed Binary function is used for cards punched in relocatable binary format by either an assembler or a compiler. Figure 2-2 shows how data is packed four card-columns into three buffer-words. One 80-column card fills 60 words of the user's buffer. Column 1 rows 12-5 in each card contain the Record Length octal value *x*, where $0 < x \leq 74_8$. See *Relocatable Tape Format* in the *BASIC CONTROL SYSTEM* manual, HP 02116-9017.

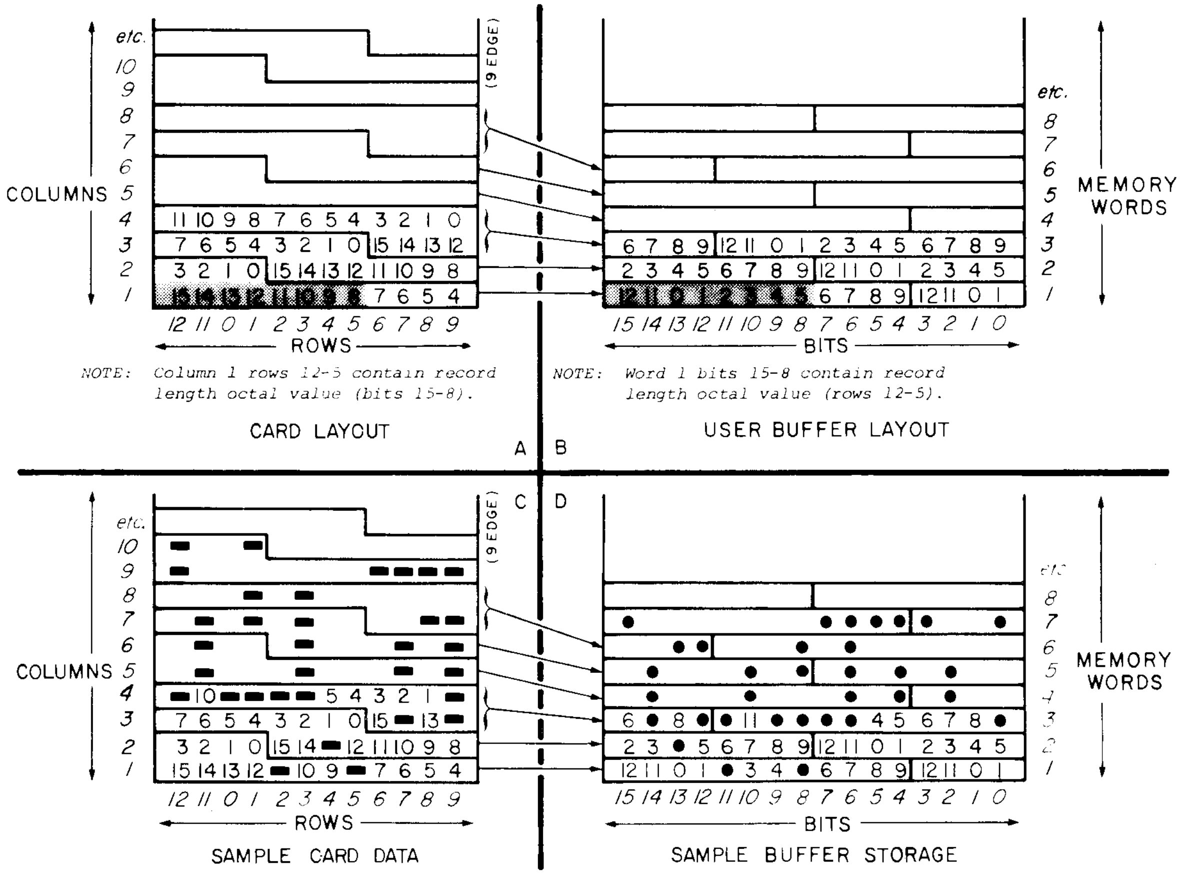
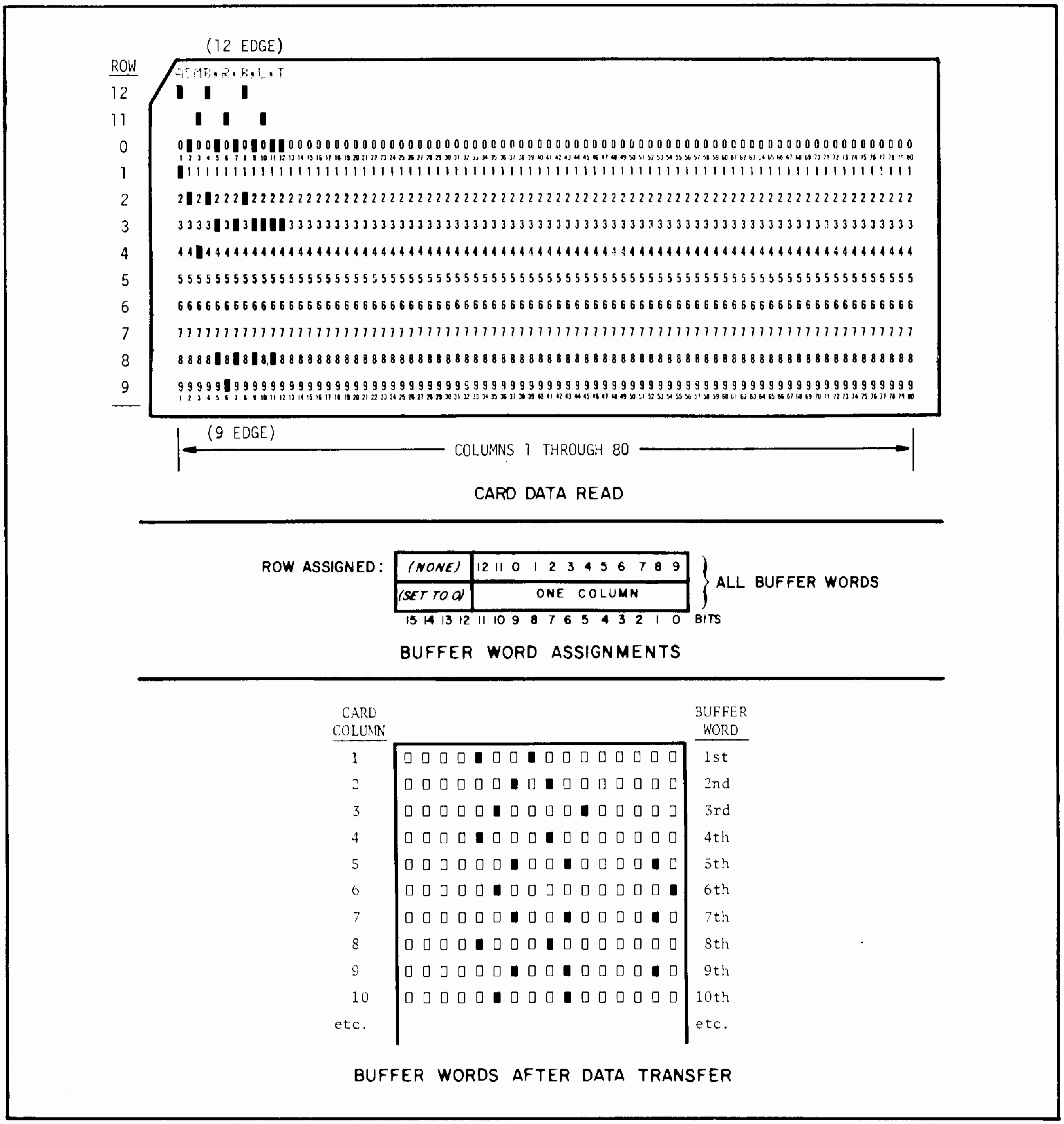


Figure 2-2. The READ PACKED BINARY Function

Column Image Binary

The Read Column Image Binary function places each 12-row card column into one 16-bit word of the user's buffer, right-justified. The four left bits (15-12) are set to 0, as shown in Figure 2-3.



CARD COLUMN																	BUFFER WORD
1	▯	▯	▯	▯	▮	▯	▯	▮	▯	▯	▯	▯	▯	▯	▯	▯	1st
2	▯	▯	▯	▯	▯	▯	▮	▯	▮	▯	▯	▯	▯	▯	▯	▯	2nd
3	▯	▯	▯	▯	▯	▮	▯	▯	▯	▯	▮	▯	▯	▯	▯	▯	3rd
4	▯	▯	▯	▯	▮	▯	▯	▯	▮	▯	▯	▯	▯	▯	▯	▯	4th
5	▯	▯	▯	▯	▯	▯	▮	▯	▯	▮	▯	▯	▯	▯	▮	▯	5th
6	▯	▯	▯	▯	▯	▮	▯	▯	▯	▯	▯	▯	▯	▯	▯	▮	6th
7	▯	▯	▯	▯	▯	▯	▮	▯	▯	▮	▯	▯	▯	▯	▮	▯	7th
8	▯	▯	▯	▯	▮	▯	▯	▯	▮	▯	▯	▯	▯	▯	▯	▯	8th
9	▯	▯	▯	▯	▯	▯	▮	▯	▯	▮	▯	▯	▯	▯	▮	▯	9th
10	▯	▯	▯	▯	▯	▮	▯	▯	▯	▮	▯	▯	▯	▯	▯	▯	10th
etc.																	etc.

Figure 2-3. The READ COLUMN IMAGE BINARY Function

CALLING SEQUENCE

A calling sequence must be executed for each card read.

```
        EXT       .IOC.
         .
         .
         .
        JSB       .IOC.
        OCT       <function><subfunction><unit reference number>
        JMP       <reject address>   \
        DEF       <buffer address>    }    [Omit for Clear or Status requests.]
        DEC \                        /
        OCT /     <buffer length>   /
        <normal return>
```

where:

```
       function - in bits 15-12       \
                                       }          See Table 2-2.
       subfunction - in bits 11-6     /

       unit-reference number - in bits 5-0
```

reject address - .IOC. returns control to the user at this location if the function or subfunction request is rejected by the initiator section. The A-Register contains 1 and B-Register contains a cause-of-reject code:

a. If the card reader is busy or inoperable, or if the driver is busy, the B-Register contains 100000_8.

b. If the subfunction requested is invalid for the card reader, the B-Register contains 0; or if DMA is required but a DMA channel is currently not available, the B-Register contains 1.

buffer address - address of the first word of the user's buffer.

buffer length - a positive integer for 16-bit buffer words, or a negative integer for 8-bit buffer characters (half words). An odd number of characters specified is incremented by one (i.e., "-3" sets two buffer words, [3+1] /2 = 2). A 0 buffer length for either Read binary function feeds a card but ignores the data. A 0 buffer length for Read Hollerith to ASCII causes an immediate normal return with no action performed.

NOTE: Buffer characters (a negative integer) should be specified only with the READ HOLLERITH to ASCII function.

Function and Subfunction Codes

The allowable function and subfunction codes for the .IOC. calling sequence are shown in Table 2-2.

Table 2-2

Allowable Function and Subfunction Codes

Function and subfunction	Octal code (bits 15-6)
Read Hollerith to ASCII octal equivalent conversion. Two characters per buffer word; see Table 2-1 and Figure 2-1. Trailing blanks are suppressed.	0100
Read packed binary. Four 12-row card columns packed into three 16-bit buffer words; see Figure 2-2. One 80-column card fills 60 sixteen-bit buffer words.	0103
Read column image binary. Each card column stored in one 16-bit buffer word right-justified; see Figure 2-3. Bits 15-12 are set to 0.	0101
Clear request.	0000
Dynamic Status request. See *Status*.	0300
Status request. See *Status*.	0400

STATUS REQUESTS

Two types of status requests can be made: normal, which returns the status of the Card Reader for the last time it was referenced, and dynamic, which returns the actual status of the card reader.

A normal status request returns the current contents of EQT entry words 2 and 3 for the Card Reader in the A- and B-Registers, respectively. Table 2-3 shows the meanings of status bits 7-0 in the A-Register of EQT entry word 2. This driver returns an equipment type code of 11 in bits 13-8.

Table 2-3

Status Bit Meanings

Bit (when set to 1)	Status Indicated
0	Reader not ready, or in TEST mode.
1	Illegal ASCII character(s), or hardware read trouble.
2	Card Reader in TEST mode.
3	Timing error, last column.
4	Pick failure.
5	Hopper empty.
6	Stacker is full.
7	End of file scratch is set and the feed hopper is empty.

When the user's calling sequence requests dynamic status, the driver returns only the status word in the A-Register (B-Register is unspecified). The status information is shown in Table 2-3.

TRANSMISSION LOG

The transmission log in EQT entry word 3 for the Card Reader is a positive integer. It reports the number of buffer words or characters transmitted and the data transfer mode. (When bit 15 = 1, the mode is a Read Binary function; when bit 15 = 0, the mode is Read Hollerith to ASCII.) According to the Read function requested, the transmission log count has one of three maximum values:

Function	Maximum Transmission Log Count
Read Hollerith to ASCII	80 characters or 40 words.
Read Packed Binary	60 words.
Read Column Image Binary	80 words.

The number of words or characters transmitted is determined by:

Function	Transmission Log Method
Read Hollerith to ASCII	The number of buffer characters requested, or the number of columns on the card, whichever is less, minus trailing blanks. (A totally blank card returns a zero.)
Read Packed Binary	The number of buffer words requested, or the octal number recorded in rows 12-5 of the first card column, whichever is less.
Read Column Image Binary	The number of words requested or the number of columns on the card, whichever is less.

HP 2778A, 2778A-001 LINE PRINTER

MODES OF OPERATION

This driver has three modes of operation: Plus, Normal, and TTY. The modes are selected by issuing the proper control subfunction or by selecting one of the following unit numbers at BCS configuration (PCS) time:

Unit No.	Mode
0 (Default Unit #)	Normal
2	Plus
4	TTY

For example, the Plus mode may be set at PCS time by supplying the following Equipment Table entry:

```
nn, D.12,U2
```

where *nn* is the channel number (select code) for the device.

In the Normal and Plus modes, the first character of the print buffer is used as control and is not printed. Instead, the second character of the buffer is printed in column one of the line printer paper.

In the Normal mode, if the first character is a "+", the driver interprets it as a blank (i.e., single space). In the Normal mode an attempt is made to drive the printer as a "space then print" device. Thus, if the command character says space 3 lines, the driver subtracts one and spaces 2 lines (one space was sent to terminate the last line, so the total is 3).

The Plus mode interprets a "+" in column one and overprints the current line on top of the last line. The driver sends a hold command at the end of each line and a single space before each line without a "+" in column one. The net effect is that the printer runs as a "space then print" device at approximately half-speed.

The TTY mode makes the Line Printer act like a teleprinter and prints the first character (in column one) of the buffer. Line space control for the TTY mode may be executed by using the print or control subfunction field. The TTY mode, if set, overrides the Plus and Normal modes and drives the printer as a "print then space" device. Two methods of spacing are permitted by using the print subfunction field.

The driver, in all modes, handles a line ending with a left arrow (←) by printing the first character in the buffer of the next request where the left arrow would have appeared had it been printed.

CALLING SEQUENCE

The general form of the input/output request is:

```
       EXT          .IOC.
        .
        .
        .
       JSB          .IOC.
       OCT          <function><subfunction><unit-reference>
       JMP          <reject address><error return>
       DEF          <buffer address>
       DEC \
            }       <buffer length>
       OCT /
       <normal return>
```

Input/Output Control (.IOC.)

All line printer input/output operations are requested by performing a JSB to entry point .IOC. The input/output control subroutine returns control to the calling program at the first location following the last word of the I/O request.

Function and Subfunction Codes

The second word of the I/O request determines the function to be performed and the line printer unit-reference for which the action is to be taken. The bit combinations that comprise the control word are as follows:

15 12	11 6	5 0
function	subfunction	unit-reference

The function (bits 15-12) is the basic input/output operation; it may be any of the following:

Function Name	Code (Octal)
Clear	00
Write	02
Control	03
Status	04

WRITE FUNCTION (02)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
00*x*	*xxx*	Normal and Plus mode - first character is carriage control, the ASCII character in Table 2-4. The second character is printed in column one of the line printer.
01*x*	*xxx*	TTY mode - first character is data. The carriage control character is the low 6 bits of the status word (second word of equipment table). The status word is set with an extended carriage control explained on the next page.

WRITE FUNCTION (02) (cont.)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
11*x*	*ddd*	TTY mode - first character is data. Carriage control is tape level corresponding to *ddd* in Table 2-4.
10*x*	*xx*0	Extended carriage control - first word in the buffer is sent as a carriage control command to the line printer. The first word is an octal code in bits 5-0, as defined in Table 2-5. The buffer length (I/O request fifth word) should be set to 1.
10*x*	*xx*1	Extended carriage control - first word of buffer is set into status word to be used as TTY carriage control. The first word is an octal code in bits 5-0, as defined in Table 2-5.

CONTROL FUNCTION (03)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
00*x*	000	Dynamic Status Request
00*x*	111	Clear TTY mode and Plus Mode (and set Normal mode)
00*x*	110	Set TTY mode
00*x*	010	Set Plus mode
*CC*0	*CCC*	If not one of the above codes, *CC*1*CCC* will be sent to the line printer. (See Table 2-4.)

Reject Address

Control is transferred to the third word of the I/O request if the input/output operation cannot be initiated. On transfer, the system provides status information which may be checked by the user's program. The A-Register is set to 0 to indicate that the operation is initiated, or is set to 1 to indicate that the operation is rejected. The B-Register contains the cause-of-reject code:

a. If the printer is busy or inoperable, or if the driver is busy, the B-Register contains 100000_8.

b. If the subfunction requested for the printer is invalid, the B-Register contains 0; or if DMA is required but a DMA channel is currently not available, the B-Register contains 1.

Buffer Storage Area

The buffer address is the location of the first word of data to be printed. The length of the buffer area may be specified in terms of words or characters. If the length is given as words, the value in the buffer length field must be a positive integer; if given as characters, a negative integer. A length of zero causes a blank line to be printed.

Table 2-4

Allowable Motion Requests (HP 2778A, 2778A-001)

Print Subfunction *ddd* code (octal)		Control Subfunction* *CC*0 *CCC* code (octal)	ASCII Character in Column One	Action
			0	Double space †
Printer Carriage Controls	7	67	1	Top of form †
	6	66	2	Bottom of form †
	5	65	3	Next sixth page †
	4	64	4	Next quarter page †
	3	63	5	Next half page †
	2	62	6	Next triple space line †
	1	61	7	Next double space line †
	0	60	8	Next single space line †
			9	Advance 55 lines
			:	Advance 54 lines
			;	Advance 53 lines
			<	Advance 52 lines
			=	Advance 51 lines
			>	Advance 50 lines
			?	Advance 49 lines
			@	Advance 48 lines
		47	A	Advance 47 lines
		46	B	Advance 46 lines
		45	C	Advance 45 lines
		44	D	Advance 44 lines
		43	E	Advance 43 lines
		42	F	Advance 42 lines
		41	G	Advance 41 lines
		40	H	Advance 40 lines
			I	Advance 39 lines
			J	Advance 38 lines
			K	Advance 37 lines

Table 2-4 (continued)

Print Subfunction *ddd* code (octal)	Control Subfunction* *CC*0 *CCC* code (octal)	ASCII Character in Column One	Action
		L	Advance 36 lines
		M	Advance 35 lines
		N	Advance 34 lines
		O	Advance 33 lines
		P	Advance 32 lines
	27	Q	Advance 31 lines
	26	R	Advance 30 lines
	25	S	Advance 29 lines
	24	T	Advance 28 lines
	23	U	Advance 27 lines
	22	V	Advance 26 lines
	21	W	Advance 25 lines
	20	X	Advance 24 lines
		Y	Advance 23 lines
		Z	Advance 22 lines
		[	Advance 21 lines
		\	Advance 20 lines
		]	Advance 19 lines
		↑	Advance 18 lines
		←	Advance 17 lines
		(Blank)	Advance 1 line
		!	Advance 15 lines
		"	Advance 14 lines
		#	Advance 13 lines
	04	$	Advance 12 lines
	03	%	Advance 11 lines
	02	&	Advance 10 lines
	01	' (apostrophe)	Advance 9 lines
		(	Advance 8 lines
		)	Advance 7 lines

Table 2-4 (continued)

Print Subfunction *ddd* code (octal)	Control Subfunction* *CC*0 *CCC* code (octal)	ASCII Character in Column one	Action
		*	Overprint next line
		+	In Plus mode: overprint this line
		+	In Normal mode: Advance 1 line
		, (comma)	Advance 4 lines
		-	Advance 3 lines
		. (period)	Advance 2 lines
		?	Advance 1 line

*The *x* (priority bit 9) has been set = 0 for this table.

†These control requests include an automatic page eject.

Table 2-5

Extended Carriage Control Code (HP 2778A, 2778A-001)

Octal Code (in bits 5-0)	Action
77	Top of Form†
76	Bottom of Form†
75	Next sixth Page†
74	Next quarter Page†
73	Next half Page†
72	Next triple space line†
71	Next double space line†
70	Next single space line†
67	Advance 55 lines
66	Advance 54 lines
65	Advance 53 lines
64	Advance 52 lines
63	Advance 51 lines
62	Advance 50 lines
61	Advance 49 lines
60	Advance 48 lines
57	Advance 47 lines
56	Advance 46 lines
55	Advance 45 lines
54	Advance 44 lines
53	Advance 43 lines
52	Advance 42 lines
51	Advance 41 lines
50	Advance 40 lines
47	Advance 39 lines
46	Advance 38 lines
45	Advance 37 lines
44	Advance 36 lines
43	Advance 35 lines
42	Advance 34 lines
41	Advance 33 lines
40	Advance 32 lines
37	Advance 31 lines
36	Advance 30 lines
35	Advance 29 lines
34	Advance 28 lines
33	Advance 27 lines
32	Advance 26 lines
31	Advance 25 lines
30	Advance 24 lines
27	Advance 23 lines
26	Advance 22 lines
25	Advance 21 lines
24	Advance 20 lines
23	Advance 19 lines
22	Advance 18 lines
21	Advance 17 lines
20	Advance 16 lines
17	Advance 15 lines
16	Advance 14 lines
15	Advance 13 lines
14	Advance 12 lines
13	Advance 11 lines
12	Advance 10 lines
11	Advance 9 lines
10	Advance 8 lines
7	Advance 7 lines
6	Advance 6 lines
5	Advance 5 lines
4	Advance 4 lines
3	Advance 3 lines
2	Advance 2 lines
1	Advance 1 line
0	Advance 0 line

†*These actions include an automatic page eject.*

STATUS REQUESTS

Either of the following types of status requests may be made:

a. Normal status -

```
JSB      .IOC.
OCT      0400 <unit-reference>
<normal return>
```

b. Dynamic status -

```
JSB      .IOC.
OCT      0300 <unit-reference>
<normal return>
```

The dynamic status request is used to obtain the actual status of a line printer unit. The normal status request returns the status of the line printer unit for the last time it was referenced. The dynamic status request goes to the driver for its operation; it returns only the status word in the A-Register with nothing in particular in the B-Register. The EQT status table entry is updated by this request.

Status Return Information

A-Register:

15 14	13 8	7 0
a	Equipment Type	Status

B-Register:

15	14 0
M	Transmission Log

a = Availability (A-Register bits 15 and 14):

0 = The device is available; the previous operation is complete.

1 = The driver is available; the operation could not be initiated because the device is not ready.

2 = The device is not available for another request; an operation is in progress.

Equipment Type (A-Register bits 13 - 8):

12_8 = HP 2778A (or 2778A-001) Line Printer

Status (A-Register bits 7 - 0):

Bits	Meaning
5-0	TTY termination code with bits 3-5 inverted
6	Left arrow (←) last time flag; if true, bit 6 = 1
7	Asterisk (*) last time flag; if true, bit 7 = 1

M = data transmission mode (B-Register bit 15):

Always 0 = ASCII

Transmission Log (B-Register bits 14 - 0):

This field is a log of the number of characters or words transmitted. The value is given as a positive integer and indicates characters or words as specified in the I/O request.

CLEAR REQUEST

The clear request terminates a previously issued input or output operation and sets all busy flags to "not-busy." A clear request has the following form:

```
EXT      .IOC.
 .
 .
 .
JSB      .IOC.
OCT      0000 <unit-reference>
```

On return, the contents of the A- and B-Registers are meaningless. The clear request checks for multi-unit operation based on the device; i.e., the I/O channel number. The driver is cleared only if the clear request is for the current operation I/O channel.

If a clear request is issued while operating the driver in the plus mode, either of the following two events may occur:

1. If the driver is busy, the clear request will print and space one line.
2. If the driver is not busy, the clear request will not print and space one line.

In either case, the next print request following the clear request prints without spacing ("overprint next line" has been set by the driver); i.e., if the line printer paper is resting at Top-Of-Form and the driver is not busy, the first line of the next print request prints on the first line of the paper. However, if the line printer has just printed a line prior to the clear request and the driver is not busy, the first line of the next print request overprints the last line printed. To alleviate this problem, a control request may be issued prior to the print request.

ERROR CONDITIONS

Equipment Table Flags

Word 2 of the equipment table contains no hardware status in bits 7-0. See *Status Return Information* for the meaning of these bits.

Illegal Character

Should an illegal character be encountered, the driver will output an "@" character. A legal character is defined as $\geq 40_8$ and $\leq 137_8$ (all ASCII characters are legal), and all other octal numbers are considered to be illegal characters.

Illegal Buffer Length

Should an illegal buffer length be encountered, the driver will use 132 characters (or 66 words) as a legal length. A legal buffer length is defined as $\leq$132 characters (or $\leq$66 words).

HP 2767 LINE PRINTER

MODES OF OPERATION

The HP 2767 line printer driver has three modes of operation: Plus, Normal, and TTY. The modes are selected by issuing the proper control subfunction or by selecting one of the following unit numbers at BCS configuration (PCS) time:

Unit No.	Mode
0 (Default Unit #)	Normal
2	Plus
4	TTY

For example, the Plus mode may be set at PCS time by supplying the following Equipment Table entry:

```
        nn, D.16,U2
```

where *nn* is the channel number (select code) for the device.

In the Normal and Plus modes, the first character of the print buffer is used as control and is not printed. Instead, the second character of the buffer is printed in column one of the line printer paper.

In the Normal mode, if the first character is a "+", the driver interprets it as a blank (i.e., single space). In the Normal mode an attempt is made to drive the printer as a "space then print" device. Thus, if the command character says space 3 lines, the driver subtracts one and spaces 2 lines (one space was sent to terminate the last line, so the total is 3).

The Plus mode interprets a "+" in column one and overprints the current line on top of the last line. The driver sends a hold command at the end of each line and a single space before each line without a "+" in column one. The net effect is that the printer runs as a "space then print" device.

The TTY mode makes the Line Printer act like a teleprinter and prints the first character (in column one) of the buffer. Line space control for the TTY mode may be executed by using the print or control subfunction field. The TTY mode, if set, overrides the Plus and Normal modes and drives the printer as a "print then space" device. Two methods of spacing are permitted by using the print subfunction field.

The driver, in all modes, handles a line ending with a left arrow (←) by printing the first character in the buffer of the next request where the left arrow would have appeared had it been printed.

CALLING SEQUENCE

The general form of the input/output request is:

```
EXT       .IOC.
 .
 .
 .
JSB       .IOC.
OCT       <function><subfunction><unit-reference>
JMP       <reject address><error return>
DEF       <buffer address>
DEC }
OCT }     <buffer length>
<normal return>
```

Input/Output Control (.IOC.)

All input/output operations are requested by performing a JSB to entry point .IOC. The input/output control subroutine returns control to the calling program at the first location following the last word of the I/O request.

Function and Subfunction Codes

The second word of the I/O request determines the function to be performed and the line printer unit-reference for which the action is to be taken. The bit combinations that comprise the control word as follows:

15 12	11 6	5 0
function	subfunction	unit-reference

The function (bits 15-12) is the basic input/output operation; it may be any of the following:

Function Name	Code (Octal)
Clear	00
Write	02
Control	03
Status	04

WRITE FUNCTION (02)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
00*x*	*xxx*	Normal and Plus mode - first character is carriage control, the ASCII character in Table 2-6. The second character is printed in column one of the line printer.
01*x*	*xxx*	TTY mode - first character is data. The carriage control character is the low 6 bits of the status word (second word of equipment table). The status word is set with an extended carriage control explained below.
11*x*	*ddd*	TTY mode - first character is data. Carriage control is tape level corresponding to *ddd* in Table 2-6.
10*x*	*xx*0	Extended carriage control - first word in the buffer is sent as a carriage control command to the line printer. The first word is an octal code in bits 5-0, as defined in Table 2-7. The buffer length (I/O request fifth word) should be set to 1.

WRITE FUNCTION (02) (cont.)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
10*x*	*xx*1	Extended carriage control - first word of buffer is set into status word to be used as TTY carriage control. The first word is an octal code in bits 5-0, as defined in Table 2-7.

CONTROL FUNCTION (03)

Subfunction Bits (Ignore the *x*'s)		Subfunction Description
00*x*	000	Dynamic Status Request
00*x*	111	Clear TTY mode and Plus mode (and set Normal mode)
00*x*	110	Set TTY mode
00*x*	010	Set Plus mode
CCx	*CCC*	If not one of the above codes, *CC*1*CCC* will be sent to the line printer (See Table 2-6.)

Reject Address

Control is transferred to the third word of the I/O request if the input/output operation cannot be initiated. On transfer, the system provides status information which may be checked by the user's program. The A-Register is set to 0 to indicate that the operation is initiated, or is set to 1 to indicate that the operation is rejected. The B-Register contains the cause-of-reject code:

a. If the printer is busy or inoperable, or if the driver is busy, the B-Register contains 100000_8.

b. If the subfunction requested for the printer is invalid, the B-Register contains 0; or if DMA is required but a DMA channel is currently not available, the B-Register contains 1.

Buffer Storage Area

The buffer address is the location of the first word of data to be printed. The length of the buffer area may be specified in terms of words or characters. If the length is given as words, the value in the buffer length field must be a positive integer; if given as characters, a negative integer. A length of zero causes a blank line to be printed.

Table 2-6

Allowable Motion Requests (HP 2767)

Print Subfunction *ddd* code (octal)		Control Subfunction** *CCx CCC* code (octal)	ASCII Character in Column One	Action
			0	Double space †
Printer Carriage Controls	7	67	1	Top of form †
	6	66	2	Bottom of form †
	5	65	3	Next sixth page †
	4	64	4	Next quarter page †
	3	63	5	Next half page †
	2	62	6	Next triple space line †
	1	61	7	Next double space line †
	0	60	8	Next single space line †
			9	Advance 55 lines *
			:	Advance 54 lines *
			;	Advance 53 lines *
			<	Advance 52 lines *
			=	Advance 51 lines *
			>	Advance 50 lines *
			?	Advance 49 lines *
			@	Advance 48 lines *
		47	A	Advance 47 lines *
		46	B	Advance 46 lines *
		45	C	Advance 45 lines *
		44	D	Advance 44 lines *
		43	E	Advance 43 lines *
		42	F	Advance 42 lines *
		41	G	Advance 41 lines *
		40	H	Advance 40 lines *
			I	Advance 39 lines *
			J	Advance 38 lines *
			K	Advance 37 lines *

Table 2-6 (continued)

Print Subfunction *ddd* code (octal)	Control Subfunction** *CCx CCC* code (octal)	ASCII Character in Column One	Action
		L	Advance 36 lines *
		M	Advance 35 lines *
		N	Advance 34 lines *
		O	Advance 33 lines *
		P	Advance 32 lines *
	27	Q	Advance 31 lines *
	26	R	Advance 30 lines *
	25	S	Advance 29 lines *
	24	T	Advance 28 lines *
	23	U	Advance 27 lines *
	22	V	Advance 26 lines *
	21	W	Advance 25 lines *
	20	X	Advance 24 lines *
		Y	Advance 23 lines *
		Z	Advance 22 lines *
		[	Advance 21 lines *
		\	Advance 20 lines *
		]	Advance 19 lines *
		↑	Advance 18 lines *
		←	Advance 17 lines *
		(Blank)	Advance 1 line *
		!	Advance 15 lines *
		"	Advance 14 lines *
		#	Advance 13 lines *
	04	$	Advance 12 lines *
	03	%	Advance 11 lines *
	02	&	Advance 10 lines *
	01	' (apostrophe)	Advance 9 lines *
		(	Advance 8 lines *
		(	Advance 7 lines *
		*	Overprint next line

Table 2-6 (continued)

Print Subfunction *ddd* code (octal)	Control Subfunction** *CCx CCC* code (octal)	ASCII Character in Column One	Action
		+	In Plus mode: overprint this line
		+	In Normal mode: Advance 1 line *
		, (comma)	Advance 4 lines *
		-	Advance 3 lines *
		. (period)	Advance 2 lines *
		/	Advance 1 line *

*Add six lines for any multiple skips crossing the page perforations. The HP 2767 line printer will not print in three lines before and after the page perforations. Continuous listings are not possible with this printer.

**The *x* (priority bit 9) has been set = 0 for this table.

†These control requests include an automatic page eject.

Table 2-7

Extended Carriage Control Code (HP 2767)

Octal Code (in bits 5-0)	Action	Octal Code (in bits 5-0)	Action
77	Top of Form†	37	Advance 31 lines
76	Bottom of Form†	36	Advance 30 lines
75	Next sixth Page†	35	Advance 29 lines
74	Next quarter Page†	34	Advance 28 lines
73	Next half Page†	33	Advance 27 lines
72	Next triple space line†	32	Advance 26 lines
71	Next double space line†	31	Advance 25 lines
70	Next single space line†	30	Advance 24 lines
67	Advance 55 lines	27	Advance 23 lines
66	Advance 54 lines	26	Advance 22 lines
65	Advance 53 lines	25	Advance 21 lines
64	Advance 52 lines	24	Advance 20 lines
63	Advance 51 lines	23	Advance 19 lines
62	Advance 50 lines	22	Advance 18 lines
61	Advance 49 lines	21	Advance 17 lines
60	Advance 48 lines	20	Advance 16 lines
57	Advance 47 lines	17	Advance 15 lines
56	Advance 46 lines	16	Advance 14 lines
55	Advance 45 lines	15	Advance 13 lines
54	Advance 44 lines	14	Advance 12 lines
53	Advance 43 lines	13	Advance 11 lines
52	Advance 42 lines	12	Advance 10 lines
51	Advance 41 lines	11	Advance 9 lines
50	Advance 40 lines	10	Advance 8 lines
47	Advance 39 lines	7	Advance 7 lines
46	Advance 38 lines	6	Advance 6 lines
45	Advance 37 lines	5	Advance 5 lines
44	Advance 36 lines	4	Advance 4 lines
43	Advance 35 lines	3	Advance 3 lines
42	Advance 34 lines	2	Advance 2 lines
41	Advance 33 lines	1	Advance 1 line
40	Advance 32 lines	0	Advance 0 line

†*These actions include an automatic page eject.*

STATUS REQUESTS

Either of the following types of status requests may be made:

a. Normal status -

```
JSB      .IOC.
OCT      0400 <unit-reference>
<return>
```

b. Dynamic status -

```
JSB      .IOC.
OCT      0300 <unit-reference>
<return>
```

The dynamic status request is used to obtain the actual status of a line printer unit. The normal status request returns the status of the line printer unit for the last time it was referenced. The dynamic status request goes to the driver for its operation; it returns only the status word in the A-Register with nothing in particular in the B-Register. The EQT status table entry is updated by this request.

Status Return Information

	15 14	13 8	7 0
A-Register:	a	Equipment Type	Status

	15	14 0
B-Register:	M	Transmission Log

a = Availability (A-Register bits 15 and 14):

0 = The device is available; the previous operation is complete.

1 = The driver is available; the operation could not be initiated because the device is not ready.

2 = The device is not available for another request; an operation is in progress.

Equipment Type (A-Register bits 13 - 8):

16_8 = HP 2767 Line Printer

Status (A-Register bits 7 - 0):

Bits	Meaning
5-0	TTY termination code with bits 3-5 inverted.
6	Left arrow (←) last time flag; if true, bit 6 = 1
7	Asterisk (*) last time flag; if true, bit 7 = 1

M = data transmission mode (B-Register bit 15):

Always 0 = ASCII

Transmission Log (B-Register bits 14 - 0):

This field is a log of the number of characters or words transmitted. The value is given as a positive integer and indicates characters or words as specified in the I/O request.

CLEAR REQUEST

The clear request terminates a previously issued input or output operation and sets all busy flags to "not-busy". A clear request has the following form:

```
EXT       .IOC.
 .
 .
 .
JSB       .IOC.
OCT       0000 <unit-reference>
```

On return, the contents of the A- and B-Registers are meaningless. The clear request checks for multi-unit operation based on the device; i.e., the I/O channel number. The driver is cleared only if the clear request is for the current operation I/O channel.

If a clear request is issued while operating the driver in the plus mode, either of the following two events may occur:

1. If the driver is busy, the clear request will print and space one line.

2. If the driver is not busy, the clear request will not print and space one line.

In either case, the next print request following the clear request prints without spacing ("overprint next line" has been set by the driver); i.e., if the line printer paper is resting at Top-Of-Form and the driver is not busy, the first line of the next print request prints on the first line of the paper. However, if the line printer has just printed a line prior to the clear request and the driver is not busy, the first line of the next print request overprints the last line printed. To alleviate this problem, a control request may be issued prior to the print request.

ERROR CONDITIONS

Equipment Table Flags

Bits 14-9 of word one of the equipment table contain the line count of the HP 2767 Line Printer; i.e., if the carriage is resting on line 20, the bits contain 20B. Word 2 contains no hardware status in bits 7-0. See *Status Return Information* for the meaning of these bits.

Illegal Character

Should an illegal character be encountered, the driver will output an "@" character. A legal character is defined as $\geq 40_8$ and $\leq 137_8$ (all ASCII characters are legal), and all other octal numbers are considered to be illegal characters.

Illegal Buffer Length

Should an illegal buffer length be encountered, the driver will use 80 characters (or 40 words) as a legal length. A legal buffer length is defined as ≤80 characters (or ≤40 words).

KENNEDY INCREMENTAL TRANSPORT

RECORD FORMATS

Binary Coded Decimal Records

A BCD record is a group of BCD characters terminated (on magnetic tape) by a record gap. A request to write a BCD record results in the translation of each 7-level ASCII character in the buffer area into a 6-level BCD character on magnetic tape. (See Figure 2-4 and the table on page A-2.) The translation process does not alter the original contents of the buffer.

The length of the record is determined by the number of characters or words designated in the request. A record gap is supplied at the end of each record by the input/output system.

If the last character in the buffer area is ←, however, the record gap is omitted. The ← is not written on tape.

A WRITE request specifying a buffer length of zero causes a record gap only to be written.

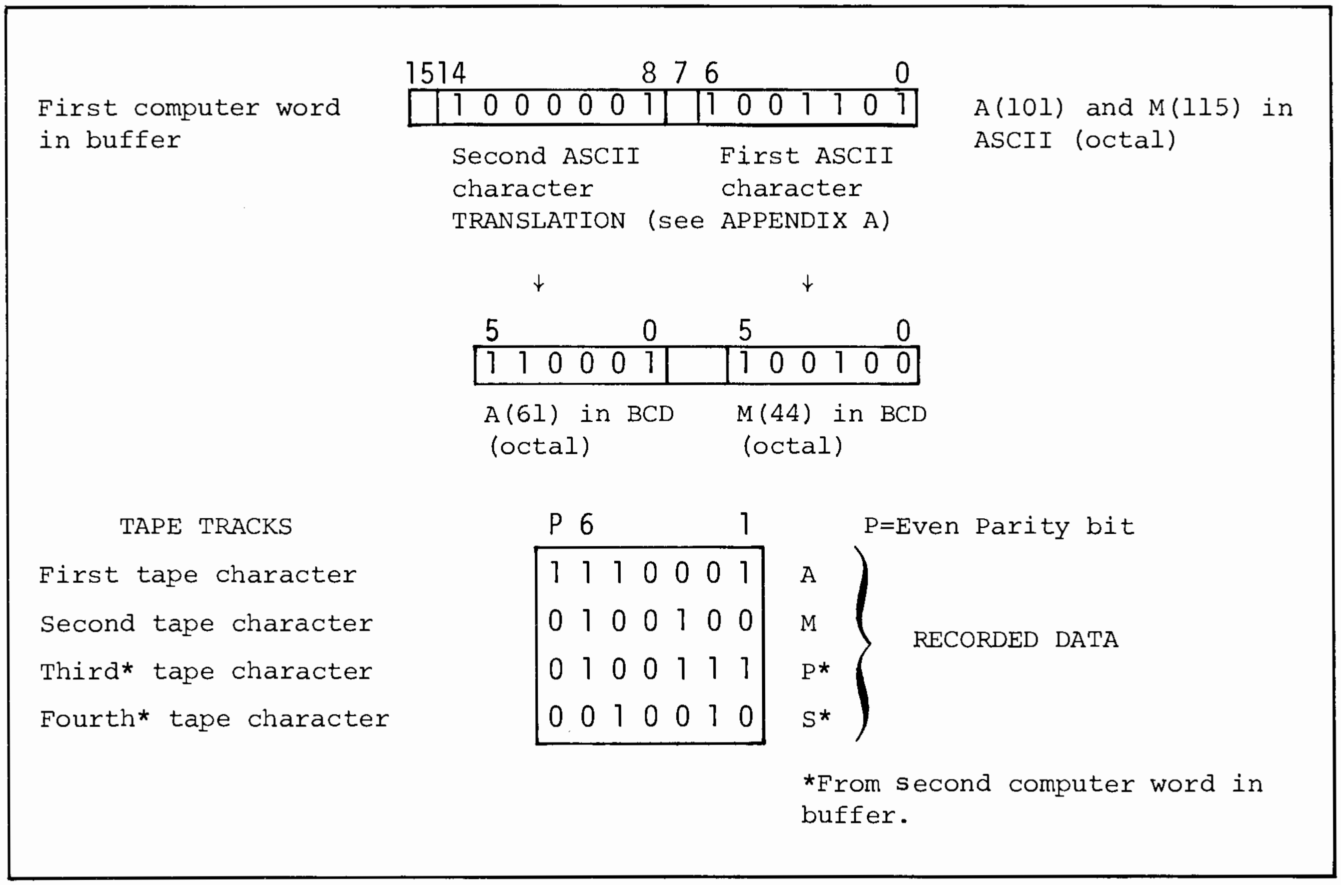


Figure 2-4. Kennedy Incremental Transport BCD Record Format

CALLING SEQUENCE

```
EXT         .IOC.
 :
JSB         .IOC.
OCT         <function><subfunction><unit-reference>
JSB }
JMP }       <reject address><error return>
DEF         <buffer address>
DEC }
OCT }       <buffer length>
<normal return>
```

Function and Subfunction Codes

Allowable function codes for the 1406/1506 Kennedy Incremental Tape Transport are as follows:

WRITE (ASCII Mode only)	0200
WRITE End-of-file	0301
CLEAR	0000

MAGNETIC TAPE SYSTEM
(HP 2020 MAGNETIC TAPE UNIT)

RECORD FORMATS

Binary Records

A binary record on magnetic tape is a group of 6-level tape "characters" recorded in odd parity and terminated by a record gap. The record length is determined by the number of characters or words in the buffer as designated in the request.

NOTE: Odd parity: a seventh bit is recorded on tape if the number of 1 bits in the six levels is an even decimal number (0, 2, 4 or 6).

Even parity: a seventh bit is recorded on tape if the number of 1 bits in the six levels is an odd decimal number (1, 3 or 5).

Each computer word is translated into three tape "characters" (and vice versa) as shown in Figure 2-5.

For output operations, the minimum buffer length is three computer words.

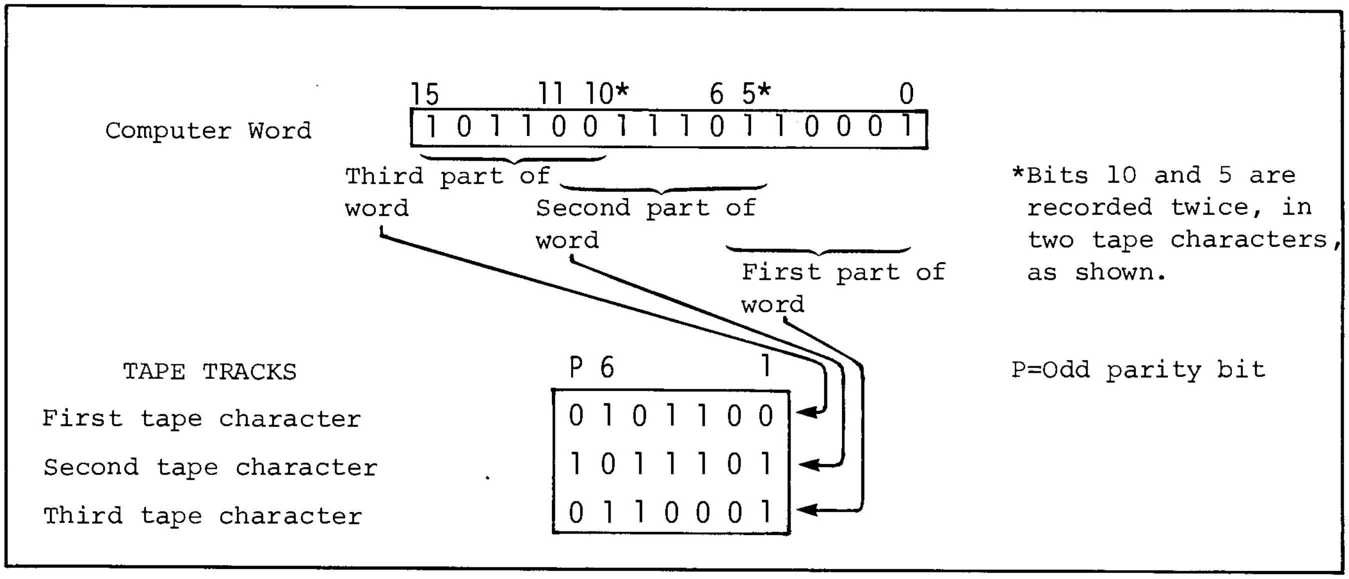


Figure 2-5. HP 2020 Magnetic Tape Unit Binary Record Format

Binary Coded Decimal Records

A BCD record on magnetic tape is a group of BCD characters recorded in even parity and terminated by a record gap. A request to write a BCD record results in the translation of each 7-level ASCII character in the buffer area into a 6-level BCD character on magnetic tape. (Refer to Figure 2-6 and the table on page A-3.) A request to read a BCD record results in the translation of each BCD character into an ASCII character after the block has been read.

The length of the record may not be more than 120 characters. A record gap is supplied at the end of each record.

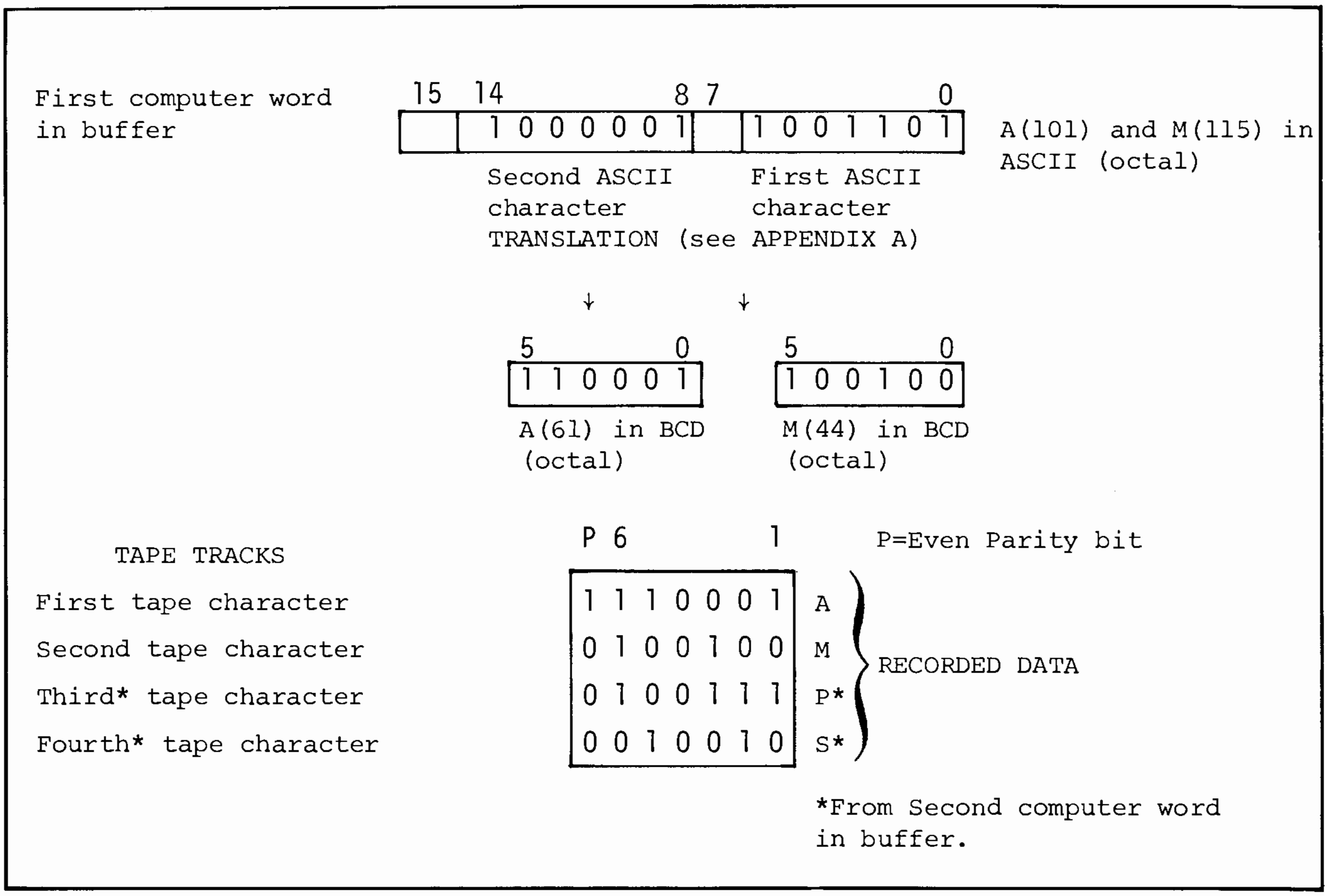


Figure 2-6. HP 2020 Magnetic Tape Unit BCD Record Format

CALLING SEQUENCE

```
       EXT          .IOC.
        .
        .
        .
       JSB          .IOC.
       OCT          <function><subfunction><unit-reference>
       JSB  }
       JMP  }       <reject address><error return>
       DEF          <buffer address>
       DEC  }
       OCT  }       <buffer length>
       <normal return>
```

Function and Subfunction Codes

All allowable combinations of function and subfunction codes are as follows:

Operation	Octal value of Bit 15-6
Read BCD record and convert to ASCII	0100
Read binary record	0101
Write BCD record after converting from ASCII	0200
Write binary record	0201
Write End-of-File (EOF) mark	0301
Forward space one record	0302
Backspace one record	0303
Rewind to start of tape (SOT) the LOAD Point, Ready (AUTO mode)	0304
Rewind to start of tape (SOT) the LOAD Point, Unload (LOCAL mode)	0305

Buffer Length

A WRITE request for the HP 2020 Magnetic Tape Unit must have a minimum buffer length of seven ASCII characters (four words). If less than seven characters are specified, spaces will be added to fill the seven characters.

MAGNETIC TAPE SYSTEM
(HP 3030 MAGNETIC TAPE UNIT)

The 3030 Driver operates the HP 3030 9-channel magnetic tape controller. It initiates, continues and completes any tape operations requested through input/output control. As a module of the Basic Control System, the driver conforms to the general specifications for performing input/output under control of the Input/Output Control (IOC) module.

Two consecutive I/O channels are required with the data channel assigned to the higher priority of the two. The other channel is the command channel. Data is transferred to or from memory by a DMA channel.

The name of the Driver is D.22. The entry points are D.22 (Initiator Section and C.22 (Continuator Section).

When configuring a BCS tape with the 3030 driver using PCS, the only requirement is a link from the command channel interrupt location to the entry point C.22 of the driver Interrupt Processor.

If an error is detected on a WRITE operation, the tape is backspaced over the record; three inches of tape are erased and the record is rewritten. This will continue until end-of-tape is sensed. If an error is detected on a READ operation, the driver will attempt to read ten times before aborting the operation.

RECORD FORMAT

Each computer word is translated into two tape "characters" by repositioning the bits as shown in Figure 2-7.

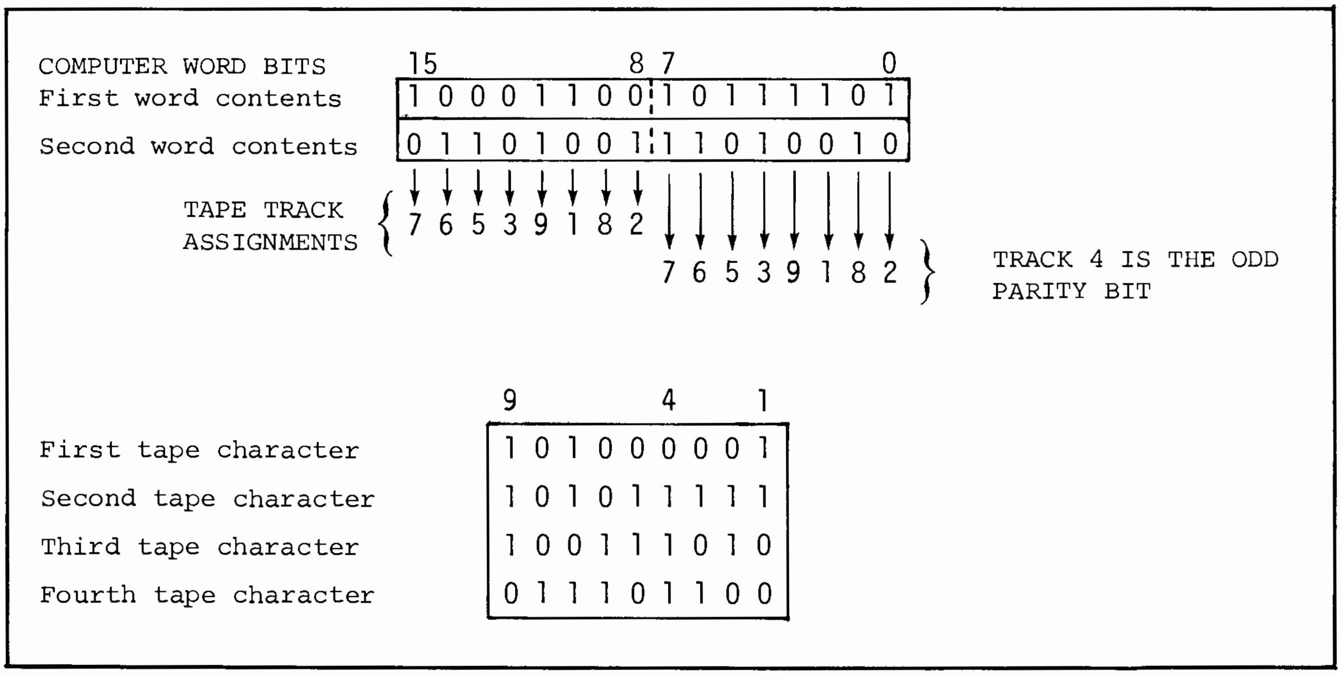


Figure 2-7. HP 3030 Magnetic Tape Unit Record Format

CALLING SEQUENCE

```
EXT      .IOC.
 .
 .
 .
JSB      .IOC.
OCT      <function><subfunction><unit-reference
JSB }
JMP }    <reject address><error return>
DEF      <buffer address>
DEC }
OCT }    <buffer length>
<normal return>
```

Function and Subfunction Codes

All allowable combinations of function and subfunction codes are as follows:

Operation	Code
CLEAR	0000
READ (binary only)	0101 (or 0100)
WRITE (binary only)	0201 (or 0200)
DYNAMIC STATUS	0300
WRITE END-OF-FILE (EOF) MARK	0301
BACKSPACE ONE RECORD	0302
FORWARD SPACE ONE RECORD	0303
REWIND TO START OF TAPE (SOT, or the LOAD POINT), READY (AUTOmode)	0304
REWIND TO START OF TAPE (SOT, or the LOAD POINT), UNLOAD (LOCAL mode)	0305

Buffer Length

Character transmission is not applicable since the transmission is via a DMA channel. The minimum data block is twelve tape characters. Output blocks with a block length less than twelve characters are padded with zeroes.

MAGNETIC TAPE SYSTEM
(HP 7970 MAGNETIC TAPE UNIT)

CALLING SEQUENCE

```
       EXT      .IOC.
        .
        .
        .
       JSB      .IOC.
       OCT      <function><subfunction><unit-reference>
       JMP      <reject address>
       DEF      <buffer address>
       DEC      <buffer length>
       <normal return>
```

where:

function (specified in bits 15-12)
subfunction (specified in bits 11-6) — Specifies the type of input/output operation being requested: Clear, Read, Write, Control, Status.

unit-reference (specified in bits 5-0) — Specifies the unit-reference number of the device used for input/output operations.

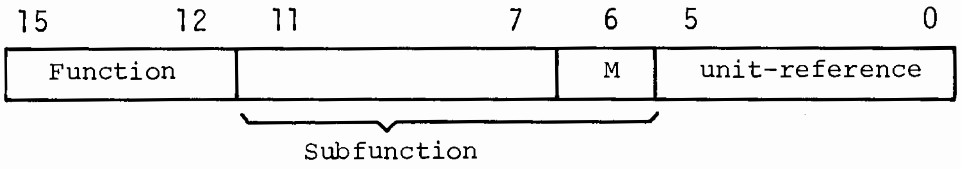
reject address — .IOC. returns control to the user at this location.

buffer address — Address of the first word of the user's buffer.

buffer length — The value in the buffer length field is specified in words (positive integer) or characters (negative integer). A buffer length of zero causes the driver to take no action on a write. A zero buffer length on binary read causes the driver to make a forward skip of one record, while a zero buffer length on ASCII read causes no action to be taken by the driver.

Function, Subfunction, Unit-Reference Codes

The second word of the request determines the function to be performed and the MT unit-reference for which the action is to be taken.



If DMA is being used, the maximum I/O request must be no greater than the equivalent of 16,383 words.

> *NOTE: Setting the mode (m) bit 6 (on) causes the computer to transmit binary data as it appears in memory or on magnetic tape. Clearing the mode bit 6 (off) causes the computer to transmit ASCII data as it appears in memory or on tape.*

Reject Address

If the input/output operation cannot be performed, control is transferred to the third word of the calling sequence. When control is transferred, the computer system provides status information which can be checked by the user's program. The contents of the A-Register indicate the physical status of the equipment, and the contents of the B-Register indicate the cause of reject.

a. If bit 15 is 1, the driver is busy (unavailable).

b. If bit 0 is 1, a DMA channel is not yet available to operate the device.

c. If both bit 15 and bit 0 are 0, then the subfunction selected is illegal.

INPUT/OUTPUT REQUESTS

ALLOWABLE MOTION REQUESTS

Operation	Octal value of bits 15-6
Read ASCII record (RRF)	0100
Read Binary Record (RRF)	0101
Write ASCII record (WCC)	0200
Write Binary record (WCC)	0201
Write End-of-file mark (GFM)	0301
Backspace record (BSR)	0302
Forward space record (FSR)	0303
Rewind (REW)	0304
Rewind/Off Line (RWO)	0305
Erase four inches of tape (GAP)	0306
MTS Relocating Loader Skip record	0307
Forward Space Record (FSF)	0320
Backspace File (BSF)	0321
Status	0400
Clear	0000

READ AND WRITE REQUESTS

Bit 6 is only an indication of the request type; it does not imply two physical modes on the magnetic tape unit.

REWIND OR BACKSPACE RECORD REQUESTS

This request performs no action if the tape unit addressed is at load-point. The status word indicates the SOT condition before and after the request is made.

READ PARITY ERROR CONDITIONS

The driver attempts to read a given record up to three times before declaring an irrecoverable parity error. If there is an irrecoverable parity error, the last try is transmitted to the user buffer and a normal completion return occurs. The status word indicates the parity and/or timing error.

WRITE PARITY ERROR CONDITIONS

The driver tries to rewrite a given record until either the record is successfully written or the end-of-tape is encountered.

ATTEMPTED WRITE REQUEST

If a write request is made to a magnetic tape unit without a write enable ring, the driver makes an immediate completion return to the caller. Status bit 14 is set in the status word, causing the Formatter to print *EQR and halt. To proceed, insert a write enable ring in the magnetic tape.

FORWARD MOTION REQUEST

If forward motion is requested when the tape unit is at end-of-tape, the MT driver ignores the request and makes an immediate completion return. The exceptions to this situation are;

a. Write End-of-file mark request, and

b. Read record request.

Only one of these privileged requests can be made once the EOT has been encountered; after that, they are ignored by the driver.

BACKWARD MOTION REQUEST (REWIND AND BACKSPACE RECORD AND BACKSPACE FILE)

This request restores the privileged nature of the write-end-of-file and read record requests.

FUNCTION/SUBFUNCTION CODE REQUEST 0307XX

Present in BCS MT drivers, the function/subfunction code request enables the Relocating Loader to operate within the Magnetic Tape System. If the request is followed by other I/O requests, they are treated as if the magnetic tape were not file-protected. The file protect feature is turned on again when the tape unit is rewound.

This request is identical to the forward space record request with the additional capability of spacing records within files 1 and 2 (even when the MT unit is in the protected file mode).

BACKSPACE FILE REQUEST AND FORWARD SPACE FILE REQUEST

These two requests cause the tape unit to go forward or backward until a file mark (EOF) is detected. Data is not transferred, and a parity error in any file record sets the parity error status bit.

The backspace file request positions the tape in front of a file mark or at load point, whichever comes first.

If the end-of-tape marker is sensed during execution of a forward space (record) request, the tape stops at the end of the current record. A status request should be used to check for this condition.

STATUS REQUESTS

As soon as tape movement operations for rewind and rewind/standby are initiated, the magnetic tape unit is available. The "A" field of a status reply is set to 00, enabling a system status request to indicate "not busy" for this EQT entry.

The normal status request returns the tape unit to the status when it was last referenced.

```
      JSB         .IOC.
      OCT         0400 <unit-reference>
      <return>
```

The dynamic status request is used to obtain the actual status of a magnetic tape unit. It goes to the driver for operation and returns only the status word in the A-Register. The contents of the B-Register are not significant.

```
      JSB         .IOC.
      OCT         0300
      <return>
```

Status Request Information

A-Register contents:

15 14	13 8	7 0
a	Equipment Type	Status

Bits 15-14 indicate the availability of the device (a):

If 0, the magnetic tape unit is available; previous operation is complete.

If 1, the magnetic tape unit is available; previous operation was ignored because either a write request was made without a write enable ring, or a tape motion request was made when the tape unit was off-line.

If 2, the magnetic tape unit is not available for another request; an operation is now in progress.

Bits 13-8 indicate the equipment type, i.e., specified as 23_8.

Bits 7-0 indicate the status of the device.

Bit	Condition
7	File Mark Sensed (EOF)
6	Load Point Status (BOT)
5	End-of-tape (EOT)
4	Data Timing Error
3	Command Rejected by the Controller
2	File Protected (no write enable ring)
1	Parity and/or Timing Error
0	Tape unit not on-line

NOTE: Bit 3 cannot be set using the driver.

B-Register contents:

15	14 0
m	Transmission Log

Bit 15, m, indicates the mode of data transmission (from the request)

If bit 15 = 0, ASCII code transmission

If bit 15 = 1, binary code transmission

Bit 14-0 indicate the transmission log, a field that is the number of characters or words tranmitted. The value is given as a positive integer and indicates characters or words as specified in the calling sequence of the read or write request. The driver cannot read or write an odd number of characters for this tape because the controller is a word device.

Minimum record length is one word.

An end-of-file mark record returns the user request length in the transmission log after being read, therefore allowing the binary read operation to operate properly through the Formatter. A write end-of-file mark returns one in the transmission log.

Control requests with a subfunction between 02 and 07 set the transmission log to zero.

Function requests of type 03 set m = 1.

CLEAR REQUEST

The clear request terminates a previously issued input or output operation before all data is transmitted. This request checks for multi-unit operation based on the device (i.e., I/O channel number). The driver is cleared only if the clear request is for the current operation I/O channel and physical unit number.

```
        EXT     .IOC.
         .
         .
         .
        JSB     .IOC.
        OCT     0000 <unit-reference>
```

On return, the contents of the A- and B-Registers are not significant.

CONTROL REQUESTS

A request directed to .IOC. may also control the positioning of a reel on a magnetic tape device. The calling sequence is similar to the input/output request, but consists of only three words:

INPUT/OUTPUT REQUESTS

```
EXT         .IOC.
 .
 .
 .
JSB         .IOC.
OCT         <function><subfunction><unit-reference>
JSB }
JMP }       <reject address><error return>
<normal return>
```

The second word of the request has the following composition:

15 12	11 9	8 6	5 0
function	▨	subfunction	unit - reference

The function defines the calling sequence as a tape control request:

Function Name	Code (octal)
Position Tape	03

The subfunction defines the type of positioning:

Subfunction (octal)	Operation
0	dynamic tape status
1	write end-of-file
2	backspace one record
3	forward space one record
4	rewind
5	rewind and standby

As soon as tape movement operations (rewind, and standby) are initiated, the device is considered to be available; the "a" field of a status reply is set to 00 (see STATUS Request). The input/output driver is thus free to process requests for other devices. To obtain the actual status of the device when one of these commands has been issued, the dynamic tape status request is used. If the tape movement operation is still in progress the "a" field is set to 10.

DATA SOURCE INTERFACE

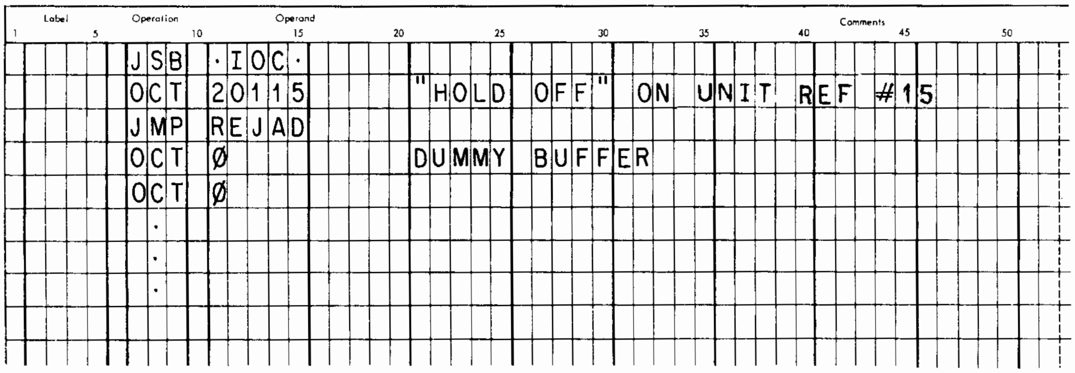
CALLING SEQUENCES

Binary Output Operation

A binary output operation causes the removal of "hold-off." The calling sequence is as below:

```
JSB       .IOC.
OCT       <function><subfunction><unit-reference>
JSB }
JMP }     <reject address><error return>
OCT       0    dummy buffer
OCT       0    buffer length
<normal return>
```

Example:


Label
Operation
Operand
Comments
1
5
10
15
20
25
30
35
40
45
50

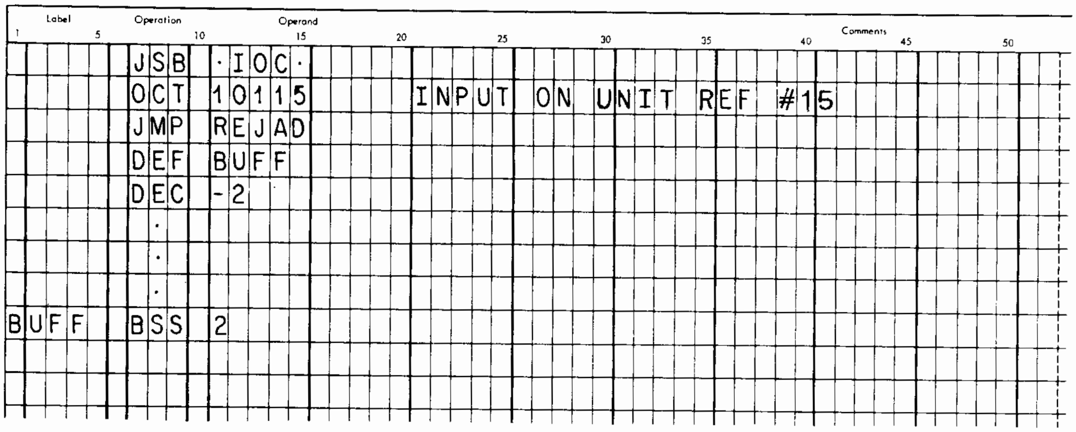
```
      JSB .IOC.
      OCT 20115     "HOLD OFF" ON UNIT REF #15
      JMP REJAD
      OCT Ø         DUMMY BUFFER
      OCT Ø
       .
       .
       .
```

Binary Input Operation

A binary input operation must have a 2-word buffer. Thirty-two bits (4 BCD characters) are read directly into the 2-word buffer.

```
                    JSB      .IOC.
                    OCT      <function><subfunction><unit-reference>
                    JSB }
                    JMB }    <reject address><error return>
                    DEF      <buffer address>
                    DEC      4  (for 4 characters) or DEC-2 (for 2 words)
                     .
                     .
                     .
buffer address      BSS      2
```

Example:



Label
Operation
Operand
Comments
1
5
10
15
20
25
30
35
40
45
50

```
      JSB .IOC.
      OCT 10115    INPUT ON UNIT REF #15
      JMP REJAD
      DEF BUFF
      DEC -2
       .
       .
       .
BUFF  BSS 2
```

ASCII Input Operation

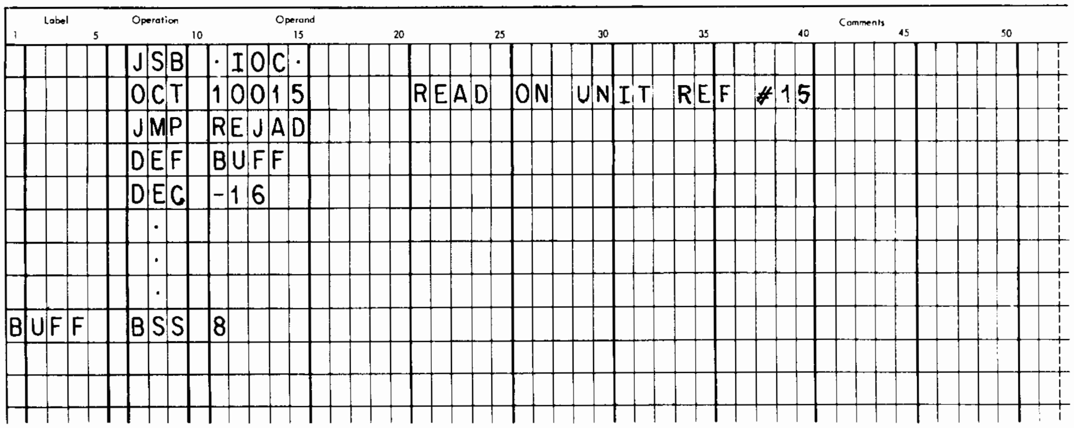
An ASCII input operation must have an 8 word buffer. Eight BCD characters are converted into 16 ASCII characters in the following format:

$$rf.d_5d_4d_3d_2d_1d_\emptyset E\text{-}ss\wedge\wedge gg$$

r	range - a negative power of 10
f	function
d_5-$d_\emptyset$	six digit data value
E-ss	range expressed as an exponent of two digits
^^	two blanks
gg	function expressed as a two-digit number

```
                JSB     .IOC.
                OCT     <function><subfunction><unit-reference>
                JMP }
                JSB }   <reject address>
                DEF     <buffer address>
                DEC     -16
                 .
                 .
buffer address  BSS     8
```

Example:



```
     JSB .IOC.
     OCT 10015          READ ON UNIT REF #15
     JMP REJAD
     DEF BUFF
     DEC -16
      .
      .
      .
BUFF BSS 8
```

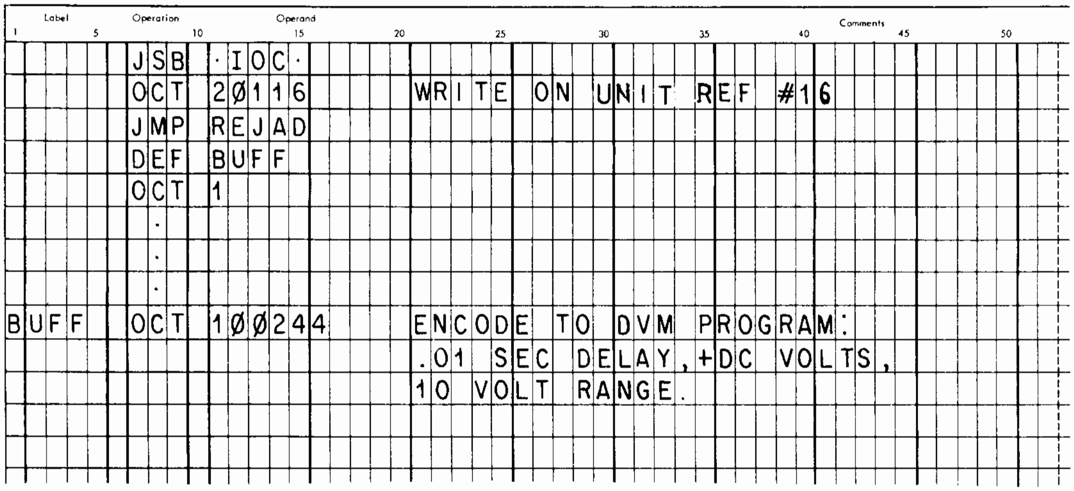
DIGITAL VOLTMETER PROGRAMMER

CALLING SEQUENCE

A WRITE request for the Digital Voltmeter Programmer requires that a one-word buffer be specified. This word contains the voltmeter program: sample period (bit 7-6), function (bits 5-3), and range (bits 2-0). If bit 15 contains a 1, an encode command is sent to the Voltmeter (bit 15 always be 0 if the configuration includes a Scanner).

```
                    JSB      .IOC.
                    OCT      <function><subfunction><unit-reference>
                    JSB  }
                    JMP  }   <reject address><error return>
                    DEF      buffer address
                    OCT      1
                    <normal return>
buffer address      OCT      voltmeter program
```

Example:



```
      JSB  .IOC.
      OCT  2Ø116     WRITE ON UNIT REF #16
      JMP  REJAD
      DEF  BUFF
      OCT  1
       .
       .
       .
BUFF  OCT  1ØØ244    ENCODE TO DVM PROGRAM:
                     .01 SEC DELAY, +DC VOLTS,
                     10 VOLT RANGE.
```

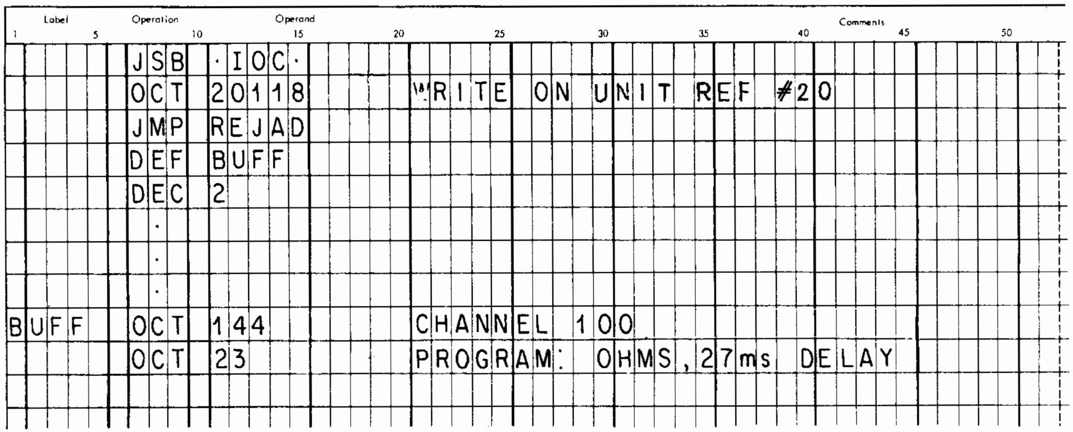
SCANNER PROGRAMMER

CALLING SEQUENCE

A WRITE request for the Scanner Programmer requires a 2-word buffer. The first word contains the channel number for the start of the scan. The second word contains the scanner program: the function (bits 4-3) and the delay (bits 2-0). The driver subroutine converts the binary channel number value produced by the Assembler to the BCD format required by the device.

```
                      JSB      .IOC.
                      OCT      <function><subfunction><unit-reference>
                      JSB  }
                      JMP  }   <reject address><error return>
                      DEF      <buffer address>
                      DEC      2
                      <normal return>
buffer address        OCT      xx     starting channel number
                      OCT      xx     Scanner Program
```

Example:


Label
Operation
Operand
Comments
1
5
10
15
20
25
30
35
40
45
50

```
      JSB .IOC.
      OCT 20118      WRITE ON UNIT REF #20
      JMP REJAD
      DEF BUFF
      DEC 2
       .
       .
       .
BUFF  OCT 144        CHANNEL 100
      OCT 23         PROGRAM: OHMS,27ms DELAY
```

INSTRUMENT CLEAR AND STATUS REQUESTS

INSTRUMENT CLEAR REQUEST

A CLEAR request on one of the instrument drivers follows the standard form:

```
                    JSB     .IOC.
                    OCT     <function><unit-reference>
                    <return>
```

where the function code = 00.

The request will result in the following conditions:

Data Source Interface	A CLEAR request causes no action. It is included for compatibility only.
Digital Voltmeter Programmer	A CLEAR request to this driver will remove the present program from the DVM but the program will not be destroyed.
Crossbar Scanner	A CLEAR request will inhibit the STEP or RESET command on the Scanner programmer driver.

INSTRUMENT STATUS REQUEST

No status information is available from the instrument drivers.

MARK SENSE CARD READER

The BCS Mark Sense Card Reader Driver D.15 operates the HP 2761A-007 Mark Sense Card Reader by initiating, continuing, and completing any operations through the Input/Output Control (.IOC.) subroutine within the Basic Control System (BCS). As a module of the BCS, this driver conforms to the general specifications for performing those controls.

The Initiator section of this driver interprets the function from the calling sequence, stores the buffer address and length, and signals the Mark Sense Card Reader to feed a card. An interrupt occurs for each clock mark printed on the 9-edge of the card to cause a JMP to the Continuator section of the driver. The Continuator then performs these tasks:

1. Saves the previous contents of all registers to be used by the Continuator section.
2. Examines bits 15-12 to check the validity of data. If the data is invalid, it determines why and sets status in the EQT table.
3. Ensures that only one card will be fed.
4. Stores the number of characters or words transmitted in the transmission log.
5. Restores the previously saved contents to the registers.
6. Terminates the transfer when the end-of-card is detected.

CALLING SEQUENCE

```
EXT          .IOC.
 .
 .
 .
JSB          .IOC.
OCT          <function><unit-reference>
JSB \
     }       <reject address><error return>
JMP /
DEF          <buffer address>
DEC \
     }       <buffer length>
OCT /
<normal return>
```

BUFFER LENGTH

The length can be specified for either words (a positive integer) or for characters (a negative integer) for any of the three reading functions. If either of the READ binary functions are requested and the buffer length specified is for an odd number of characters, the length will be effectively incremented by 1. Thus if 3 characters are specified, the buffer will be set for 2 computer words (i.e., (3+1)/2=2). If the buffer length is specified to be zero, a card is fed, but its data is ignored.

STATUS FIELD

The Status field indications are:

Bits 7-0	Condition
xxxxxxx1	Hopper empty or stacker full.
xxxxxx1x	Reader not READY.
xxxxx1xx	Pick failure.

The equipment type code is 15.

The transmission log has the following maximum values:

Function	Maximum Value
Read, Hollerith to ASCII	80 characters
Read, column image binary	80 words
Read, packed binary	60 words

FUNCTIONS

Function	Contents of bits 15-6
Read, Hollerith to ASCII (octal equivalent) conversion with two characters per computer word, as described in Appendix A. *NOTE: In translating Hollerith to ASCII trailing zeroes are suppressed.*	0100
Read, packed binary; four 12-row card columns packed into three 16-bit computer words. Thus one 80-column card fills 60 words of the user's buffer. The packing format is described in the *Small Programs Manual* "BCS MARK SENSE DRIVERS, D.15" (HP 12602-90021).	0103
Read, column image binary; each card column is placed right justified into one 16-bit word. The four left bits (15-12) are set to zero, as shown in the *Small Programs Manual*.	* 0101
CLEAR request; allows the current card to finish feeding.	0000
STATUS request.	0400

* See p. [illegible]

SECTION III
RELOCATING LOADER

The Loader is the module of the Basic Control System that provides the capability of loading, linking, and initiating the execution of relocatable object programs produced by the Assembler, FORTRAN, and ALGOL. It is available in 4K and non-4K versions. ALGOL programs and the Relocatable Library stored on magnetic tape require the non-4K loader.

EXTERNAL FORM OF LOADER

The Loader, part of the tape titled "Configured BCS," is stored in an absolute record format on an external medium (on magnetic tape or 8-level paper tape) with the Input/Output Control subroutine (.IOC.) and the equipment driver subroutines. It is loaded by the Basic Binary Loader.

INTERNAL FORM OF LOADER

The Loader is located in high-numbered memory along with the Input/Output Control subroutine and the equipment driver subroutines. The Loader uses .IOC. for input/output operations; it refers to the standard input and output units. The binary object program is read from the standard input unit; comments to the user (e.g., loader diagnostics) are written on the teleprinter output unit; and library routines referenced by the object program are assumed to be on the Relocatable Library unit.

RELOCATABLE PROGRAMS

The process of assembling or compiling a set of symbolic source program statements results in the generation of relocatable object code. Relocatable code assumes a starting location of 00000. Location 00000 is termed the relative, or relocatable origin. The absolute origin (also termed the relocation base) of a relocatable program is determined by the Loader. The value of the absolute origin is added to the zero-relative value of each operand address to obtain the absolute operand address. The absolute origin, and thus the values of every operand address, may vary each time the program is loaded.

A relocatable program may be made up of several independently assembled or compiled subprograms. Each subprogram has a relative origin of 00000 and then is assigned a unique absolute origin upon being loaded. Subprograms executed as a single program may be loaded in any order. The absolute origins will differ whenever the order of loading differs.

The operand values produced by the Assembler, FORTRAN, or ALGOL may be program relocatable, base page relocatable, or common relocatable. Each of these segments of the program has a separate relocation base or origin. Operands that reference locations in the main portion of the program are incremented by the program relocation base; those referring to the base page, by the base page relocation base; and those referring to common storage, by the common relocation base.

If the Loader encounters an operand that is a reference to a location in a page other than the "current" page or "base" page, a link is established through the base page. A word in the base page is allocated to contain the full 15-bit address of the referenced location. The address of the word in the base page is then substituted as an indirect address in the instruction in the "current" page. If other similar references are made to the same location, they are linked through the same word in the base page.

RECORD TYPES

The Loader processes three to five record types for a program. These record types are produced by the Assembler, FORTRAN, or ALGOL in the following sequence:

NAM	Name record
ENT	Entry point record
EXT	External name record
DBL	Data block record
END	End record

The NAM, DBL, and END records exist for every object program; ENT and EXT appear only if the corresponding pseudo instructions are used in the source program.

NAM

The NAM record contains the name of the program and the length of the main, base page, and common segments. The NAM record signifies the beginning of the object program.

ENT

The ENT record defines the names of 1 to 14 entry points within this program. Each of the four-word entries in the record contains the name, the relocatable address of the name; and an indicator which specifies whether the address is program or base page relocatable.

EXT

The EXT record contains from 1 to 19 three-word entries which specify the external references defined in the program. The three words allow a maximum of five ASCII characters for the symbol and a number used by the Loader to identify the symbol.

DBL

A DBL record contains 1 to 45 words of the object program. It indicates the relative starting address for the string of words and whether this portion of the object code is part of the main program or base page segment. For each of the words there is also a relocation indicator which defines the relocation base to be applied to each operand value. Possible relocation factors are:

Absolute — Operand is an absolute expression or constant. There is no relocation base.

15-bit Program Relocatable — Operand is a 15-bit value to which is added the program relocation base.

15-bit Base Page Relocatable — Operand is a 15-bit value to which is added the base page relocation base.

15-bit Common Relocatable — Operand is a 15-bit value to which is added the common relocation base.

External Symbol Reference — Operand is a reference to an external symbol. Value is supplied when the Loader determines the absolute location of the linkage word in the Base Page which contains the 15-bit address of the related entry point.

Memory Reference Instruction — A memory reference instruction in the form of a two-word group which consists of the instruction code, a full 15-bit operand address, and a relocation indicator for the operand address. The relocation indicator can define the operand address to be program, base page, or common relocatable.

END

The END record terminates the block of records in an object program. The END record may contain a 15-bit address which is the location to which control is transferred by the Loader to begin program execution.

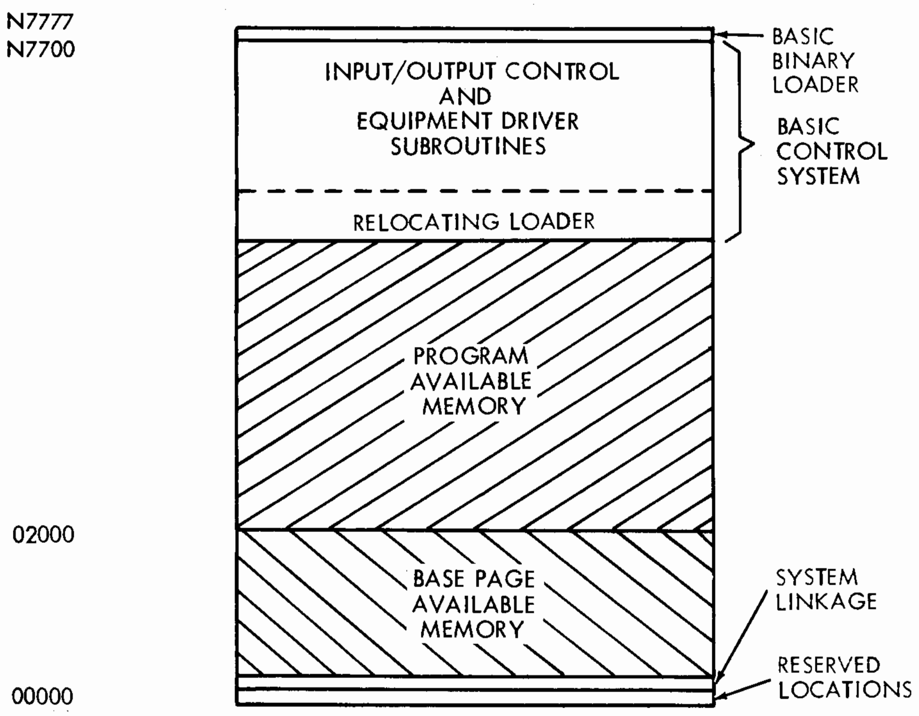
MEMORY ALLOCATION

The Loader loads the object program into available memory. Available memory is defined as that area of memory not allocated for hardware and system usage. Available memory is divided into two segments:

- Available memory in base page -- used for the operand linkage area, program blocks origined into the Base Page by the Assembler pseudo instruction ORB, and for program blocks assigned to the Base Page by the Loader when the amount of program available memory is insufficient.

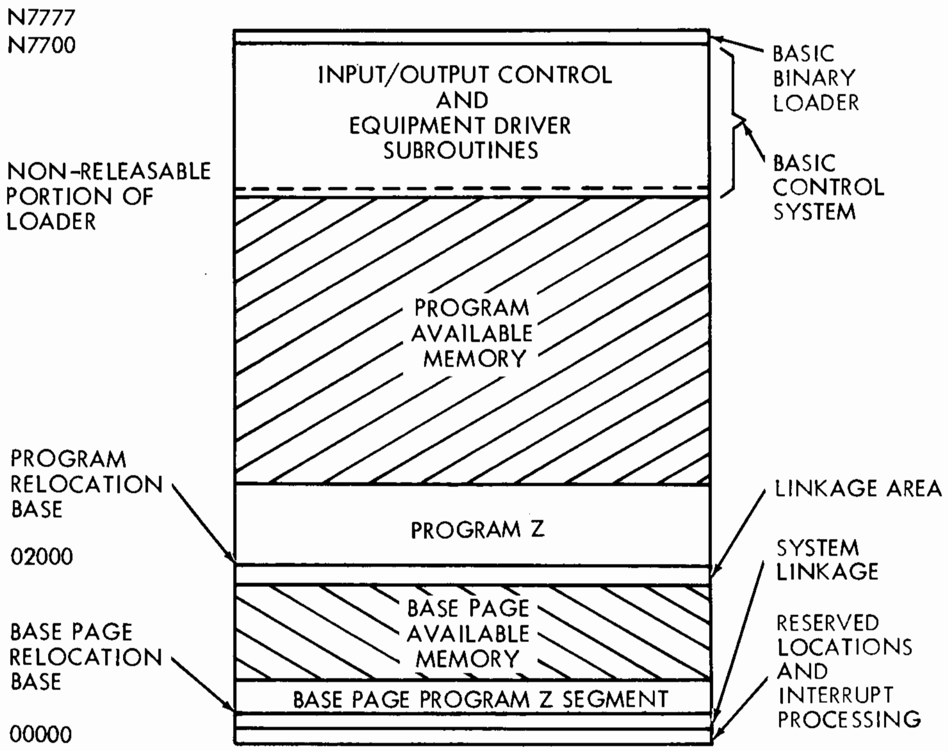
- Program available memory -- used for the main body of the program and may be used by the common block should the area used by the Loader be insufficient.

Prior to loading the object program, memory is allocated as follows:



N = 0(4K), 1(8K), 2(12K), 3(16K), 5(24K), 7(32K)

Assuming Program Z is to be loaded and executed -- after loading, the memory might be allocated as follows:

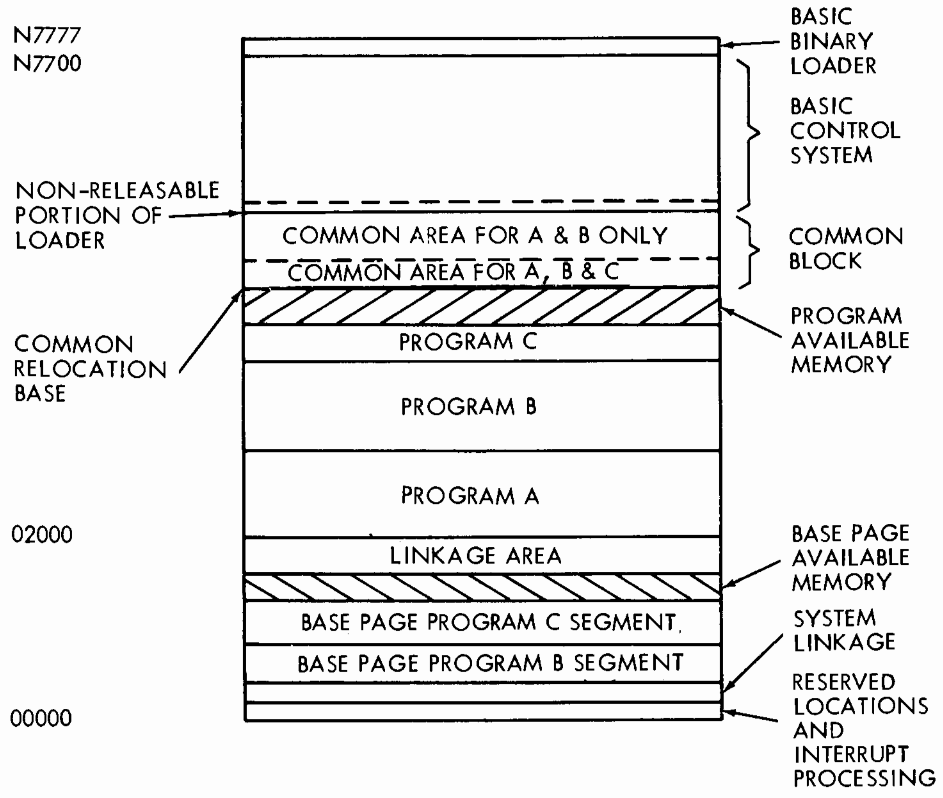


N = 0(4K), 1(8K), 2(12K), 3(16K), 5(24K), 7(32K)

Options selected during PCS processing can define the equipment driver subroutines and other routines as external routines which must be satisfied at run-time. If selected, these routines would be allocated to the available memory areas, and the length of the absolute segment of BCS reduced accordingly.

If several programs are to be loaded and executed together, the following might occur:

> Assume three programs, A, B, and C, comprise a running program. Programs A and B share a common block, a portion of which is also shared by C. Programs B and C contain segments which are designated to be allocated to the Base Page. Allocation is as follows:



N = 0(4K), 1(8K), 2(12K), 3(16K), 5(24K), 7(32K)

Common Block Allocation

The first common length declaration (i.e., the first program containing a common segment) processed by the Loader establishes the total common storage allocation in high memory overlaying the major portion of the area occupied by the Loader. Subsequent programs must contain common length declarations which are less than or equal to the length of the first declaration.

To allocate the common area, the Loader subtracts the total length of the block from the address of the last releasable word in the Loader. The resulting memory address +1 is the origin of the common block. This value is used throughout the entire loading process as the common relocation base.

Program Storage

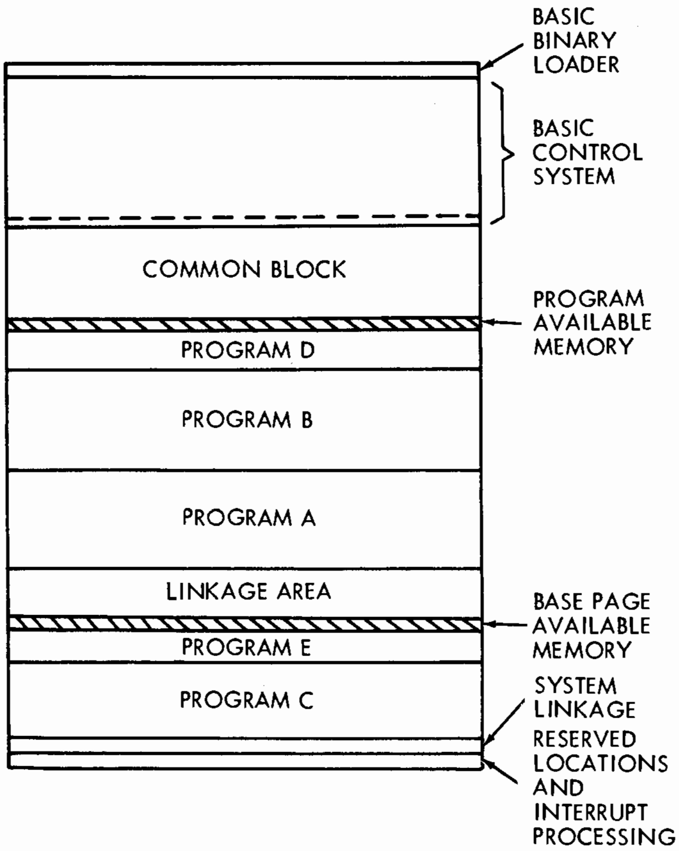
The program length is compared with the amount of available memory. If sufficient space is available, the program is loaded and the upper and lower bounds recorded. If the program has a base page segment, or if the program consists entirely of coding to be stored in the base page, the length of the segment is compared to the amount of available base page memory. If there is enough space in this area, the segment is loaded and the bounds recorded. Whatever is loaded first is usually origined at absolute location 02000 (page 1, module 0). The initial base page segment is usually origined immediately following the area set aside for reserved locations, interrupt processing, and system linkage. Subsequent main program and base page segments are loaded into the next available higher numbered areas contiguous with the previously loaded segments.

Providing the memory allocation list option is selected, the name of each program, its upper and lower bounds, and its base page upper and lower bounds are printed after the program is loaded. The format is as follows:

```
<program name>
lllll uuuuu (main program bounds)
lllll uuuuu (base page bounds)
```

If the Loader finds that the main program segment about to be loaded can not fit in the memory area available for the main segment, it compares the segment's length to the length of available memory in the base page. If there is sufficient space, the main segment will be loaded in the base page. The next segment will be loaded in the main program area if it will fit, or in the base page if not (providing there is sufficient space in the base page). When all available base page space has been used, loading is terminated.

For example, assume that several programs are to be loaded in sequence A, B, C, D, E, and have sizes such that they can not all fit in the main program available memory.



OBJECT PROGRAM RECORD PROCESSING

ENT/EXT Record Processing

The Loader constructs and maintains a Loader Symbol Table containing entry points and external symbols which are declared in the programs and entry point names of any BCS system subroutines that have been defined as relocatable. As each entry point is encountered, its relocated (absolute) address is recorded in the table. As each external reference is processed, a link word is established in the base page. The general processing of the entries in an ENT and EXT record involves searching the loader symbol table to locate a match between the symbols. When a match is found, the absolute entry point address is stored in the base page link word.

The Loader assumes that there is a user program, BCS system routine, or Relocatable Library routine entry point for every external reference. If none exists, the external reference is undefined and considered to be in error. A list of undefined external symbols is printed at the end of the loading operation. If duplicate entry points are detected, a diagnostic is issued. For duplicate entry points, only the first routine is accepted.

Each entry in the loader symbol table occupies five words. The table is positioned before the beginning of the Loader and extends backwards toward low-numbered memory. If sufficient space is not available in the main program portion of memory to store a five-word entry, a diagnostic message is issued and the loading operation is terminated.

DBL Record Processing

A load address for the data or instruction words in a DBL record is relocated by adding either the program relocation base or the base page relocation base. The resulting value is the absolute address for storing the first word. The second word is stored at address +1, the third at address +2 and so forth. A relocation base is added to each operand address as specified by the relocation indicator.

Processing an external reference word involves searching the loader symbol table for the related entry. When found, the address of the link location in the base page is extracted and stored as an indirect address in the instruction.

When a memory reference instruction is processed, the Loader first applies the proper relocation base, (program, base page, or common) to the 15-bit operand address. If the resulting absolute operand address references the base page, the address (bits 09-00) is set into the operand field and the instruction is stored in memory at the current load address. When the absolute operand address and the current load address are in the same page, the operand address is truncated to bits 09-00 and set as the instruction operand address. If the operand address is in a page other than the current load address page, the operand address is stored in the linkage area of the base page and a reference to this location set as an indirect address in the operand field of the instruction.

A memory overflow condition can occur when insufficient space is available in the base page to allocate a linkage word. A diagnostic message is issued, and the loading operation is terminated.

END Record Processing

When an END record is encountered, the Loader determines if it contains a transfer of control address. If it does, the address is saved.

If loading is from the Relocatable Library and no undefined external references exist, the end-of-loading operation is performed.

If loading is from the standard input unit or Relocatable Library unit and if undefined external references exist, the Loader requests the next record. If the next record is a NAM record, processing of the next program begins. If the result of the request is an end-of-information indication, an End condition exists.

Relocatable Library Loading

Loading from the Relocatable Library differs from loading user programs. Only those programs in the library that contain entry points matching undefined external symbols in the loader symbol table are loaded. After each library program is loaded, the loader symbol table is checked for undefined symbols. If none exist, the loading operation is complete and the program is ready to be executed.

End Condition

When the Loader requests input and end-of-tape occurs on the input device, an End condition exists. The Loader acknowledges this condition by writing the message "LOAD" on the teleprinter. The user responds to this message by setting switches 2-0 of the Switch Register. (See "Loader Operating Procedures.") Four replies are available:

a. Load next program from standard input unit. External BCS system subroutines are considered to be part of the program and must be loaded from the standard input unit (unless they are made part of Relocatable Library tape).

b. All programs are loaded; proceed to the end-of-loading operation.

c. Terminate loading operation. This forces program execution even though there may be undefined external references.

d. Load from Relocatable Library; all user programs are loaded.

End-of-Loading Operation

The end of loading is signaled by the second or fourth response to an End condition. The Loader then searches the loader symbol table for any undefined external references. Any such undefined external symbols are written on the teleprinter, and the "LOAD" message is repeated.

When the loading operation is completed or when the user has requested termination of the loading process, the Loader produces a memory allocation list. (This list may be omitted; see "Loader Operating Procedures.") The format of the list is as follows:

```
<symbol 1>   aaaaa
<symbol 2>   aaaaa
     .
     .
     .
<symbol n>   aaaaa
```

The symbols are the entry points in the user's program, the Basic Control System, or the Relocatable Library and the a's are their absolute addresses.

If a common block was allocated, the lower and upper bounds of the block are listed as follows:

```
       *COM       lllll       uuuuu
```

The bounds of the linkage area are listed as follows:

```
       *LINKS     lllll       uuuuu
```

The l's are the absolute lower bounds, and the u's are the absolute upper bounds.

PROGRAMMING CONSIDERATIONS

When a program has been completely loaded, its execution is initiated by performing a jump subroutine (JSB) to the transfer address (from the last END record containing an address). The initial contents of the transfer address should be a NOP, OCT 0, etc., not the first executable instruction of the program.

LOADER OPERATING PROCEDURES

The exact operating procedures for the Loader depend on the available hardware configuration and the construction of the Basic Control System through use of the Prepare Control System routine. The user should know the assignment of input/output equipment and memory size before using the Loader.

Loading Options

The BCS Loader loads one or more tapes containing relocatable programs. The message "LOAD" is typed when an end-of-tape condition is encountered. The user then loads the next tape, indicates loading from the Relocatable Library, specifies that loading is complete, etc. When all programs are loaded and no undefined external references remain, the Loader types the message "*LST" allowing the user to bypass part of the memory allocation list. Following the response, the Loader types the message "*RUN." The user then initiates program execution.

MEMORY ALLOCATION LIST

A memory allocation list can be obtained for the programs being loaded. The list includes the name, main program bounds, and base page bounds for each program. At the completion of the loading operation, this portion of the list may be followed by a list of all entry points and their absolute addresses, the bounds of the common block, and the bounds of the linkage area. The Switch 15 setting determines the contents of the list.

To obtain the bounds for each program on a tape, Switch 15 must be set to 0 before the tape is loaded (in response to the "LOAD" message). To bypass the program bounds listing, set Switch 15 to 1 before loading the tape. The switch setting may be altered whenever the "LOAD" message is typed.

To obtain the entry point list, the common bounds, and the linkage area bounds, set Switch 15 to 0 in response to the message "*LST", which is printed after all programs are loaded. To bypass this portion of the list, set Switch 15 to 1.

ABSOLUTE BINARY OUTPUT

When it is necessary to utilize the area occupied by the Loader for program storage or when an absolute version is desired for "production stage" programs, the user may specify that an absolute binary tape be punched. The process involves a simulated loading operation; however, the absolute program is punched on tape rather than being loaded.

The absolute records produced consist of the relocated programs (including all relocatable subroutines), the linkage area and all referenced segments of the Basic Control System. These include:

Input/output control subroutine (.IOC.)

All input/output equipment drivers

Memory Table (.MEM.)

System linkage area

Interrupt processing area

Absolute locations 2 and 3

In addition, the loader symbol table and the common and linkage area bounds are punched in ASCII format on the end of the binary tape. Ten inches of feed frames separate the binary instructions and the ASCII data. This feature provides a record of the memory allocation.

At the completion of all Loader processes, the Loader types the message "END."

To execute the program, it must be loaded using the Basic Binary Loader. To initiate execution, set 000002 into the P-Register and press RUN. The Loader has stored the transfer address of the program in locations 2 and 3 as follows:

2 contains JMP 3,I

3 contains <transfer address>

SEPARATION OF LIST AND BINARY OUTPUT

If the absolute binary output option is selected and the teleprinter is used as both a list and punch device, the Loader halts before and after each line is printed to avoid punching the line and altering the binary output.

The halts and related procedures are as follows:

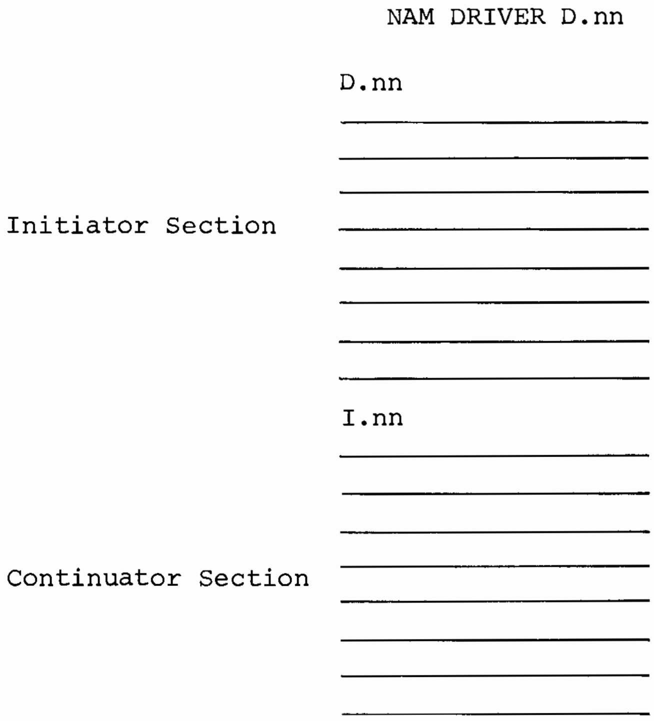
T-Register Contents	Explanation	Action
102055	A line is about to be printed.	Turn punch unit OFF. Press RUN.
102056	A line has been printed.	Turn punch unit ON. Press RUN.

SECTION IV
INPUT/OUTPUT DRIVERS

An I/O driver, operating in the BCS environment, is responsible for controlling all data transfer between an I/O device and the cpu. It operates under control from the program IOC. Its operating parameters are the user I/O request and the information contained in the device associated equipment table entry.

STRUCTURE

An I/O driver is a relocatable program segmented into two closed subroutines, termed the "initiator" and "continuator" sections. The entry point names for these two sections must be "D.nn" and "I.nn", respectively. The numeric value "nn" in the names is the equipment type code assigned to the device. For example, D.00 and I.00 are the entry points for the teleprinter driver; "00" is the equipment type code assigned to a teleprinter.



Initiator Section

This section is called directly from IOC with calling parameters including the address of the second word of the user I/O request and the address of the EQT entry for the referenced device. IOC sets these parameters in A and B and performs a JSB to the entry point "D.nn". Return to IOC from this section must be indirectly through D.nn.

On entry to D.nn,

(A) = Address of word 1 of 4-word EQT entry

(B) = Address of word 2 of I/O request

The initiator section of any driver must perform the functions described below.

1. Reject the IOC request and return to IOC (see step 6) if any of the following conditions exist:

   a. The driver is busy operating another device.

   b. The referenced device is busy or inoperable.

   c. The user request code (or other parameters) is illegal for the device.

   d. A DMA channel is not available and DMA is required for data transfer.

2. Extract the parameters from the user I/O request and save them within the driver storage.

3. Configure all I/O instructions in the driver to include the channel number for the referenced device.

4. Indicate equipment in operation:

   a. Set the "a" field in the EQT entry to 2 (busy) for the device called.

   b. Set an internal driver "busy" flag for the driver.

   c. Set a "busy" flag in IOC if a DMA channel is used.

NOTE: To set a DMA flag in IOC, do the following:

Within the IOC program the two entry points DMAC1, DMAC2 contain the DMA channel locations (6 and 7 or 7 and 6). The sign bit of the channel used must be set to 1 to indicate that the channel is busy.)

5. Initialize operating conditions and activate the device.

6. Return to IOC with the A- and B-Registers set to indicate initiation or rejection and the cause of the reject:

   (A) = 0, operation initiated
   = 1, operation rejected - reason in B-Register

   (B) = 100000, the device is busy or inoperable, or the driver is busy
   = 000001, a DMA channel is required but no channel is available
   = 000000, the request code or subfunction is not legal for the device

Continuator Section

This section is entered by device interrupt to continue or complete an operation. It may also be called from the Initiator Section to begin an operation. The entry point to this section is I.nn. There are no parameters on entry.

The continuator section of any driver must perform the functions described below.

1. Save all registers which will be used by the continuator section.

2. Perform the input or output of the next data item. If the transfer is not completed, restore the "saved" register and return control to the program.

NOTE: A driver for a device which inputs or outputs data independent of program control such as DMA would not include step 2. The device is activated by the initiator section (step 5), and the data transfer is immediately accomplished. The continuator section for such drivers merely completes the input or output operation.

3. When data transfer is completed (end-of-operation) or if a device malfunction is detected, set the following information in the EQT entry:

   The number of words or characters transferred (corresponding to the request) is set as a positive value in word 3. Bit 15 of word 3 is set to 0 or 1 to indicate the mode of transfer.

   The device status, actual or simulated, is set in bits 07-00 of word 2 and the "a" field (bits 15-14) in word 2 set to:

   0 - device available (not busy)
   1 - device available; the operation is complete
       but an error has been detected

   Bits 13-08 of word 2 must not be altered.

4. Clear all "busy" indicators. Clear the driver busy flag. If a DMA channel was used, clear the flag in IOC.

5. Restore all registers saved at the entry.

6. Return indirectly through the entry point I.nn, with the following execption:

   If end-of-operation occurs for an output or function request, the driver returns to the entry point ".BUFR" in IOC. This enables the buffered version of IOC to perform the automatic output buffering function. The standard version of IOC at this entry point just performs a normal return to the point of interruption. The calling sequence to .BUFR is:

```
           EXT    .BUFR
(P)        JSB    .BUFR
(P+1)      NOP       (holds return address from I.nn)
(P+2)      NOP       (holds EQT entry address)
```

SECTION V
PREPARE CONTROL SYSTEM

The Prepare Control System (PCS) program processes relocatable modules of the Basic Control System and produces an absolute version designed to work on a specific hardware configuration. It creates operating units of the Input/Output Control subroutine (.IOC.), the equipment driver subroutines, and the Relocating Loader. It also establishes the contents of certain locations used in interrupt handling. Options are available to define the equipment driver modules and other BCS system subroutines as relocatable programs to be loaded with the user's object program.

The Prepare Control System is an absolute program which is loaded by the Basic Binary Loader. It operates on a minimum configuration of 4K memory and a 2752A teleprinter. However, if a Paper Tape Reader and a Paper Tape Punch are available, the Prepare Control System will utilize these devices. PCS requests their assignment during the initialization phase.

After the initialization phase is completed, each module of BCS is loaded and processed by PCS. The order in which the modules are processed is not significant except that the BCS Loader must be the last module loaded. Two modules, the Input/Output Control subroutine and the Loader, require that parameters be entered via the keyboard input unit after being loaded.

INITIALIZATION PHASE

During the Initialization phase, the system requests the channel assignments of the Paper Tape Reader and the Tape Punch if available. The operator supplies this information. Next the system requests the first and last words of available memory. The first word is the location in the base page following the locations required for interrupt processing (the interrupt locations and the locations containing the addresses of the Interrupt Processors). This location defines the start of the BCS system linkage area. The last word of available memory is usually the location prior to the protected area (e.g., 7677 for 4K memory, 17677 for 8K memory).

Example:	HS INP?	Message
	1Ø	Reply
	HS PUN?	Message
	11	Reply
	FWA MEM?	Message
	3Ø	Reply
	LWA MEM?	Message
	17677	Reply
	⋮	

LOADING BCS MODULES

After the initialization phase is completed, the system types "LOAD." The BCS modules are loaded using the paper tape reader (if available) or the teleprinter. The modules may include .IOC., the equipment drivers, and the Relocating Loader. They can be loaded in any order provided that the Relocating Loader is last. The message is repeated after each module is loaded until the Loader has been processed. Diagnostics are printed if certain error conditions occur during the loading.

The absolute lower and upper bounds of each program within BCS are listed after the program is loaded. The format is as follows:

```
<program name>
lllll  uuuuu
```

Equipment driver subroutines and interrupt processing sections that are to be used in relocatable form are identified during PCS processing but are not loaded. At the completion of the processing, PCS requests the missing subroutines. The proper response identifies each as external.

INPUT/OUTPUT EQUIPMENT PARAMETERS

After the Input/Output Control module is loaded, PCS requests the information needed to construct the Equipment Table (EQT) and Standard Equipment Table (SQT). (These tables are described in Appendices B and C.)

EQT Statements

PCS first types the messages TABLE ENTRY and EQT. The operator responds by supplying the equipment table entries in the following format:

nn, D.ee [,D] [,Uu]

nn The channel number (select code) for the device. For a device connected to two or more channels, nn is the lower numbered channel.

D.ee The BCS symbolic name for the related equipment driver subroutine. ee is the equipment code. Driver names include:

D.00 - 2752A Teleprinter
D.01 - 2737A Punched Tape Reader
D.02 - 2753A Tape Punch
D.15 - Mark Sense Reader
D.20 - Kennedy 1406 Incremental Tape Transport
D.21 - 2020 Magnetic Tape Unit
D.22 - 3030 Magnetic Tape Unit
D.40 - Data Source Interface
D.41 - Integrating Digital Voltmeter
D.42 - Guarded Crossbar Scanner
D.43 - Time Base Generator

D A Direct Memory Access channel is required to operate the device.

Uu The physical unit number u (0-7) for addressing the device if it is attached to a multi-unit controller.

The same response is used regardless of whether the related subroutine driver is to be relocatable or absolute (part of BCS). If the driver is not encountered during processing, PCS prints the following:

```
I/O DRIVER?
D.EE
```

A response of ! indicates that the driver is to be in relocatable form. (Any other response at this time is an error.) Drivers which use DMA or reference IOERR in IOC cannot be used externally.

The order in which the EQT statements are submitted defines the position of the entry in the Equipment Table. It also establishes the unit-reference number that the programmer uses in writing input/output requests to .IOC. The first statement entered describes the unit which is to be referenced as number 7_8; the second statement, number 10_8; the third statement, number 11_8; etc. Numbers 1 through 6 are reserved for Standard unit definition in the Standard Equipment Table.

The statement "/E" is entered to terminate the EQT input.

Example:

		Unit-Reference Number
*TABLE ENTRY	Message	
EQT?	Message	
1Ø,D.Ø1	Statement	7
11,D.Ø2	Statement	10
12,D.ØØ	Statement	11
/E	Terminator	

SQT Table Statements

In constructing the Standard Equipment Table, PCS types a mnemonic for the standard unit and waits for the reply. The reply consists of the unit-reference number for a device previously described in the Equipment Table.

Example:

SQT?	Message	
-KYBD?	Message to assign keyboard input	
11	Reply:	unit-reference number for teleprinter
-TTY?		message to assign teleprinter output
11	Reply:	unit-reference number for teleprinter
-LIB?		message to assign Relocatable Library
7	Reply:	unit-reference number for paper tape reader
-PUNCH?	Message to assign punch output	
1Ø	Reply:	unit-reference number for tape punch
-INPUT?	Message to assign input	
7	Reply:	unit-reference number for paper tape reader
-LIST?	Message to assign list output	
11	Reply:	unit-reference number for teleprinter

DMA Statement

After the equipment tables are completed, PCS requests information about the availability of DMA channels to be controlled by the Input/Output Control and equipment driver subroutines. PCS types the message DMA? and the operator responds with the available DMA channel numbers. The format of the reply is:

$$c_1 \; [,c_2]$$

c_1 is 6 if one channel is available

c_2 is 7 if the second channel is available

If no DMA channel is available, the reply is 0 (zero).

Example:

DMA?	Message
6,7	Reply for two channels

If the reply contains any characters other than 0, 6 or 7, it is an error and a diagnostic is issued.

INTERRUPT LINKAGE PARAMETERS

After the Relocating Loader is loaded, PCS requests the parameters needed to set the interrupt linkage for input/output processing. The information required for each device includes:

The interrupt location within the reserved location area in low core.

The entry point name of the interrupt processing section in the equipment driver subroutine for the device.

The address of the word in the base page which is to contain the 15-bit absolute address of this entry point name.

The same response is used regardless of whether the subroutine driver is to be relocatable or absolute (part of BCS). If the entry point was not encountered during processing, PCS prints the following:

```
*UN name
```

A response of ! indicates that the driver is to be in relocatable form. (Any other response at this time redefines the linkage.) Drivers which use DMA or reference IOERR in IOC cannot be used externally.

Given this information, PCS sets in the interrupt location a Jump Subroutine (Indirect) to the word holding the absolute address for the entry point of the Interrupt Processor.

Location	Content
10	JSB 20B,I
.	.
.	.
.	.
20	DEF I.01

10 is the interrupt location

20 holds the address of the entry point, I.01, of the Interrupt Processor.

PCS types the message INTERRUPT LINKAGE? The operator responds with a message in the following format:

$$a_1, a_2, \text{I.ee}$$

a_1 The address in low core of the interrupt location for the device (channel).

a_2 The address in the base page of the word to contain the absolute address of the interrupt processor entry point.

I.ee The entry point name of the interrupt processor section of the equipment driver subroutine. ee is the equipment type code used by BCS. Entry point names are as follows:

I.00 - 2752A Teleprinter

I.01 - 2737A Punched Tape Reader

I.02 - 2753A Tape Punch

I.15 - Mark Sense Reader

I.20 - Kennedy 1406 Incremental Tape Transport

I.21 and C.21 - 2020 Magnetic Tape Unit

I.22 and C.22 - 3030 Magnetic Tape Unit

I.43 - Time Base Generator

Both the magnetic tape systems are connected to two channels; the lower numbered channel transfers data (I.21, I.22); the higher numbered channel transfers commands (C.21, C.22).

The statement "/E" is entered to terminate the interrupt linkage parameter input.

Example:

INTERRUPT LINKAGE?	Message	
1Ø,2Ø,I.Ø1	Reply:	The paper tape reader uses interrupt location 10. The absolute address for entry point I.01 is location 20 in the base page.
11,21,I.Ø2	Reply:	The tape punch uses interrupt location 11. The address of I.02 is at location 21.
12,22,I.ØØ	The teleprinter uses interrupt location 12. The address of I.00 is at location 22.	
/E	Terminates linkage parameters.	

The response to the INTERRUPT LINKAGE? message may have the following form if a constant (for example, a halt) is to be set in the interrupt location.

a, c

a The address in low core of the interrupt location for the device (channel).

c The constant in octal form that is to be stored at loacation a.

Example:

INTERRUPT LINKAGE?	Message	
27,1Ø2Ø27	Reply:	A halt executed when interrupt occurs on channel 27.
26,Ø	Reply:	A NOP is executed when interrupt occurs on channel 26; the program resumes normal execution.

PROCESSING COMPLETION

When the interrupt linkage parameters have been supplied, PCS performs the following functions:

1. Prints the message *UNDEFINED SYMBOL followed by the entry point names of all system subroutines which have been referenced as externals but not loaded. At this point, PCS may continue and the missing subroutines loaded or, the symbols may be added to the Relocating Loader's Loader Symbol Table. Undefined symbols are assigned as value of 77777 for an absolute address.
2. Completes the construction of the Loader Symbol Table.
3. Sets the memory table (symbolic location .MEM.) in the Relocating Loader to reflect the final bounds of available memory.

Following this, PCS prints a list of all Basic Control System entry points and the bounds of the system linkage area in the base page.

Example:

```
.SQT.    17472
.EQT.    175ØØ
.IOC.    17515
DMAC1    17676
DMAC2    17677
IOERR    17656
XSQT     17674
XEQT     17675
D.ØØ     16745
I.ØØ     171Ø7
D.Ø1     164Ø6
I.Ø1     16521
D.Ø2     16115
I.Ø2     16226
```

```
.LDR.      15413
HALT       1611Ø
.MEM.      1611Ø
LST        141Ø2


*SYSTEM LINK
ØØØ3Ø      ØØØ71
```

The final step in PCS processing is the punching of an absolute binary tape of the configured Basic Control System. This tape can be loaded by the Basic Binary Loader. When the tape is to be punched, BCS types the message *BCS ABSOLUTE OUTPUT. At the completion of the PCS run, the message *END is typed. The tape is punched using the tape punch unit, if available, or the teleprinter.

SECTION VI
DEBUGGING SYSTEM

The debugging routine for BCS provides the following programming aids:

- Print (dump) selected areas of memory in octal or ASCII format
- Trace portions of the program during execution
- Modify the contents of selected areas in memory
- Modify simulated computer registers
- Instruction and operand breakpoint halts
- Initiate execution at any point in program
- Debugging routine restart
- Specifying relocatable program base

The debugging routine supervises the operation of a program in the check-out (debugging) phase by using an interpretive mode of execution with simulated A-, B-, E-, Overflow and P-Registers.

The debugging routine is a relocatable program. The user loads it into memory after his relocatable programs and before the library subroutines. The debugging routine uses the input/output control subroutine, IOC.

OPERATOR COMMUNICATION

All communication between the debugging routine and the user is done through the standard keyboard input and standard teleprinter output units normally assigned to a teleprinter.

After the program is loaded, the debugging routine pauses for the first type-in. The operator types one or more control statements to direct the operation of the debugging routine. Each statement must be terminated by a carriage return and a line feed. The last statement of the set must be a RUN statement.

When an operation requested by a control statement is completed, a pause occurs (except for the Trace operation). The operator can continue by typing RUN or he can enter new control statements. To regain control at any other time, the operator must use Switch 15.

> CAUTION: *Setting Switch 15 when input/output operations are in progress could disrupt any incomplete I/O operation. If this happens, a message is typed.*

CONTROL STATEMENTS

The basic format of the control statement is a single alphabetic character representing the requested operation, followed by a parameter list containing the arguments for the operation. The parameters are separated by commas. The statement is of variable length and is terminated by a carriage return and a linefeed. The numeric fields in the parameter list must be in octal; leading zeros may be omitted.

Program Relocation Base

The format is:

```
M,a
```

This statement defines the program relocation base, a, as the absolute origin in memory of the user's relocatable program. This address may be obtained from the listing produced by the Relocating Loader during loading. If not specified, a value of zero is assumed. The value is added to all address parameters entered by the operator.

DEBUG does not check for memory address greater than the core size; therefore, locations in the base page may be altered if the program relocation base is too high.

Specification of this value allows subsequent reference in the control statements to addresses as shown on the program listing produced by the Assembler or the FORTRAN compiler. If this control statement is not used, program address parameters for other control statements must be absolute.

Example:

```
        M, 2ØØØ
```

Set Memory

The format is:

$S, a, v_1, v_2, \ldots, v_n$

The above statement allows the user to set one or more values into locations defined by the first address, a. The value specified for v_1 is stored in location a; the value for v_2, in location a + 1; and so forth. To specify that an existing value in memory is to remain unchanged, two consecutive commas are used in the control statement. Any number of values may be entered via one control statement provided the length of the statement does not exceed 72 characters.

Example:

```
        S,5,062006
        S,30,136100,026040
        S,40,136101,026050
```

Set Register

The format is:

```
        W,r,v
```

This statement sets the value, v, into register, r, where the register is defined as follows:

```
r = A,A-Register
  = B,B-Register
  = E,E-Register
  = O,Overflow
```

Since the debugging routine simulates the register, the results of a set register operation are not reflected on the computer front panel.

Examples:

```
W,B,2
W,A,102000
W,E,1
```

Dump Memory

The format is:

D,A,a_1,a_2

D,B,a_1,a_2

The second parameter indicates the format of the print-out: A specifies ASCII, and B specifies octal. The address a_1 designates the location of the word or the first of a series of words that is to be dumped. If the second address, a_2, is greater tha a_1, a block of memory, a_1 through a_2, is printed. If a_2 is the same as a_1, only one location is printed.

After the data is printed, the debugging routine waits for the operator to enter another control statement.

Example:

```
D,A,430,477
```

Breakpoint Halt

The format is:

```
B,I,a
B,O,a
```

The first form specifies the address, a, of an instruction breakpoint. Before the instruction at address a is executed, the debugging routine writes a standard breakpoint message. (See "Output Formats.")

The second form specifies the address, a, of an operand breakpoint. When the debugging routine detects an effective operand address equal to the value of a, it writes a standard breakpoint message. The operand breakpoint occurs before the memory reference is completed, and the register contents in the message are the contents during the instruction execution and not at completion.

After the breakpoint message is transmitted, the debugging routine waits for the user to enter another control statement.

One or both types of breakpoint halts may be selected. Once selected, a breakpoint address remains in effect until a new address is selected, until a Restart statement is entered, or until the selection is terminated by the statements:

B,I,Ø or B,O,Ø

Trace

The format is:

$T,a_1[,a_2]$

When the Trace operation is specified, the execution of the instruction located at address a_1, or the execution of every instruction within the area a_1 through a_2, causes the printing of a standard breakpoint message. (See "Output Formats.")

The printing occurs before each instruction is executed. Each time the $a_1 - a_2$ area is reached, the printing resumes; no pause occurs on completion as in the other debugging routine operations.

The area to be traced must not contain calls to the input/output control routine, IOC. The Trace operation uses IOC to print the breakpoint message. An attempt to trace I/O operations will result in I/O errors.

The trace of the area remains in effect until a new area is selected or until the selection is terminated by the statement:

```
T,Ø
```

To enter a new Trace control statement while the program is in operation, Switch 15 must be used.

Run

The format is:

```
R [,a]
```

This statement is used to initiate the execution of the program being debugged. It can also be used to continue execution after a pause in execution (caused by setting switch register bit 15 to 1 or by breakpoint halt). If the letter R only is entered, execution starts with the next sequential instruction in the user's program. To start at another location, the operator enters the address, a.

Restart

The format is:

```
A
```

This statement, consisting of the letter A is used to abort the current operation and restart. This results in all debugging routine and input/output operations in progress being cleared.

CONTROL STATEMENT ERROR

If an incorrect control statement is entered, the following message is typed:

```
        ENTRY ERROR
```

This indicates that the character representing the operation is invalid, or that an illegal parameter has been typed. To recover, type in the correct control statement.

HALT

Any halt operations coded within the user's program result in a typeout consisting of the letter H followed by the standard breakpoint message. The operator can then type in one or more control statements or can reinitiate program execution (with the R control statement).

INDIRECT LOOP

The debugging routine counts levels when indirect addressing is detected. When ten consecutive levels of indirect addressing have occurred, an indirect address loop is assumed and the following is typed out:

```
        INDIRECT LOOP
        L <standard breakpoint message>
```

OUTPUT FORMATS

The debugging routine operations produce either the standard breakpoint message or the memory dump as printed output.

Standard Breakpoint Message

Each standard breakpoint message has the following format:

<id> P = v_1 I = v_2 A = v_3 B = v_4 E = v_5 O = v_6 MA = v_7 MC = v_8

The <id> is a letter identifying the operation producing the output:

id = I, Instruction breakpoint
= O, Operand breakpoint
= T, Trace
= S, Switch 15 set up
= L, Indirect Loop
= H, Halt in object program

The v's are octal values of registers and memory locations as follows:

P - P-Register (instruction address)
I - Instruction (contents)
A - A-Register
B - B-Register
E - E-Register
O - Overflow
MA - Effective operand address of a memory reference instruction
MC - Contents of effective address of a memory reference instruction

Dump

The Dump output record format consists of the contents up to eight consecutive words preceded by the address of the first word:

	addr.	word$_1$	word$_2$...	word$_8$
Octal:	aaaaa	000000	000000...	000000
ASCII:	aaaaa	cc	cc	cc

Octal words consist of six octal digits; ASCII words are listed as two ASCII characters. The contents of eight or more consecutive words are not written if they are the same as the last word of the previous record. Instead, a record containing only an asterisk is produced.

OPERATING PROCEDURES

The following procedures indicate the sequence of steps for use of the debugging routine.

a. Set the teleprinter to LINE, and check that all equipment to be used is operable.

b. Load BCS using the Basic Binary Loader.

c. Set a starting address of 2 and zero the Switch Register.

d. Establish Relocating Loader parameters. (If relocation base is to be entered during operation of the debugging routine, the address must be obtained during loading by setting Switch 15 to 0.)

e. Load user relocatable object programs.

f. Load debugging program (treated as a relocatable program). The debugging routine need not be loaded as the last relocatable program. If loaded in any other order, however, the absolute address assigned to the symbolic location DEBUG must be entered manually as the starting address for the program.

g. Load Relocatable Library routines.

h. Press RUN.

i. The program pauses for the operator-typed control statements.

j. The program can be restarted at any point by entering the absolute address assigned to the symbolic location DEBRS into the P-Register, and pressing RUN.

EXAMPLE

The routine employed in this example is a simple loop which totals the contents of a block of data. Assume that program TOTAL computes personal expenses for a 31-day month. Data (each day's expenses) is read in from the punched tape reader. The sum is printed on the teleprinter.

The program is written and assembled as below. To check it out, a data tape, consisting of a series of 10's is prepared.

```
10, carriage return, linefeed
10, carriage return, linefeed
10, carriage return, linefeed
 .
 .
 .

 PAGE ØØØ2 #Ø1

ØØØ1  ØØØØØ                NAM TOTAL
ØØØ2  ØØØØØ ØØØØØØ  START  NOP
ØØØ3  ØØØØ1 Ø62162R        LDA =D-31
ØØØ4  ØØØØ2 Ø72156R        STA CTR
ØØØ5  ØØØØ3 Ø62163R        LDA =B5
ØØØ6  ØØØØ4 ØØ64Ø4         CLB,INB
ØØØ7  ØØØØ5 Ø16ØØ4X        JSB .DIO.
ØØØ8  ØØØØ6 ØØØØØØ         ABS Ø
ØØØ9  ØØØØ7 ØØØØ14R        DEF *+5
ØØ1Ø  ØØØ1Ø Ø16ØØ6X        JSB .IOR.
ØØ11  ØØØ11 Ø16ØØ1X        DST INPUT,I
      ØØØ12 1ØØØ55R
ØØ12  ØØØ13 Ø16ØØ5X        JSB .RAR.
ØØ13  ØØØ14 Ø66Ø55R        LDB INPUT          INPUT THE DATA
ØØ14  ØØØ15 Ø46164R        ADB =B2
ØØ15  ØØØ16 Ø76Ø55R        STB INPUT
ØØ16  ØØØ17 Ø36156R        ISZ CTR
ØØ17  ØØØ2Ø Ø26ØØ3R        JMP START+3
```

```
ØØ19  ØØØ21 Ø62162R        LDA =D-31
ØØ2Ø  ØØØ22 Ø72156R        STA CTR             INITIALIZE
ØØ21  ØØØ23 Ø16ØØ2X        DLD =FØ.Ø
      ØØØ24 ØØØ165R
ØØ22  ØØØ25 Ø16ØØ1X        DST ANSW
      ØØØ26 ØØØ154R

ØØ24  ØØØ27 Ø16ØØ2X        DLD .MON,I
      ØØØ3Ø 1ØØØ54R
ØØ25  ØØØ31 Ø16ØØ3X        FAD ANSW
      ØØØ32 ØØØ154R
ØØ26  ØØØ33 Ø16ØØ1X        DST ANSW
      ØØØ34 ØØØ154R
ØØ27  ØØØ35 Ø66Ø54R SUM    LDB .MON
ØØ28  ØØØ36 Ø46164R        ADB =B2             ADDITION LOOP
ØØ29  ØØØ37 Ø76Ø54R        STB .MON
ØØ3Ø  ØØØ4Ø Ø36156R        ISZ CTR
ØØ31  ØØØ41 Ø26Ø35R        JMP SUM

ØØ33  ØØØ42 Ø62164R        LDA =B2
ØØ34  ØØØ43 ØØ64ØØ         CLB
ØØ35  ØØØ44 Ø16ØØ4X        JSB .DIO.
ØØ36  ØØØ45 ØØØ157R        DEF OUTPT
ØØ37  ØØØ46 ØØØØ53R        DEF *+5
ØØ38  ØØØ47 Ø16ØØ2X        DLD ANSW            OUTPUT THE RESULT
      ØØØ5Ø ØØØ154R
ØØ39  ØØØ51 Ø16ØØ6X        JSB .IOR.
ØØ4Ø  ØØØ52 Ø16ØØ7X        JSB .DTA.
ØØ41  ØØØ53 1Ø2Ø77         HLT 77B
```

```
ØØ43  ØØØ54 ØØØØ56R .MON   DEF MONTH
ØØ44  ØØØ55 ØØØØ56R INPUT  DEF MONTH
ØØ45  ØØØ56 ØØØØØØ  MONTH  BSS 62
ØØ46  ØØ154 ØØØØØØ  ANSW   BSS 2
ØØ47  ØØ156 ØØØØØØ  CTR    BSS 1
ØØ48                       EXT .DIO.,.RAR.,.IOR.,.DTA.
ØØ49  ØØ157 Ø241Ø6  OUTPT  ASC 3,(F8.2)
      ØØ16Ø Ø34Ø56
      ØØ161 Ø31Ø51

      ØØ162 177741
      ØØ163 ØØØØØ5
      ØØ164 ØØØØØ2
      ØØ165 ØØØØØØ
      ØØ166 ØØØØØØ
ØØ5Ø                       END START
**  NO ERRORS*
```

The TOTAL object tape is loaded by the Basic Control System. The debugging system is loaded next and then the library tape. The program is executed using the debugging system by the following instructions:

`M,2ØØØ` Set program relocation base.

`B,I,53` Breakpoint instruction is 53, the location of the terminating halt in the program.

`R,1` Initiate execution at location 1.

```
1Ø.ØØ
H P=ØØØ53    I=1Ø2Ø77    A=177777    B=ØØ6115    E=Ø    O=1
```

The correct answer for the test data would be 31.00 not the 10.00 that was output.

The following procedure illustrates one method for detecting errors in the program.

```
M,2ØØØ       Set program relocation base.
```

Dump a portion of the storage area MONTH:

```
D,B,56,7Ø
DUMP--BASE = Ø2ØØØ

ØØØ56                                           Ø5Ø5Ø3 ØØØ333
ØØØ6Ø ØØ1253 ØØØØØØ ØØØØØØ ØØ4267 Ø177ØØ ØØØØØØ Ø53Ø7Ø Ø1177Ø
ØØØ7Ø ØØ2256
```

Read in the data:

```
B,I,21
R,1
I P= ØØØ21  I=Ø62162  A=ØØØØØØ  B=ØØ2154  E=Ø O=Ø  MA=ØØ162  MC=177741
```

Check to see that the data has been stored in memory:

```
D,B,56,7Ø
DUMP--BASE = Ø2ØØØ

ØØØ56                                                  Ø5ØØØØ  ØØØØ1Ø
ØØØ6Ø  Ø5ØØØØ  ØØØØ1Ø  Ø5ØØØØ  ØØØØ1Ø  Ø5ØØØØ  ØØØØ1Ø  Ø5ØØØØ  ØØØØ1Ø
ØØØ7Ø  Ø5ØØØØ
```

Knowing that the data has been stored in MONTH, perform the first addition:

```
B,I,35
R,21
I P= ØØØ35  I=Ø66Ø54  A=Ø5ØØØØ  B=ØØØØ1Ø  E=Ø  O=Ø  MA=ØØØ54  MC=ØØ2Ø56
```

Check to see that the first day's expenses have been stored at ANSW:

```
D,B,154,155
DUMP--BASE = Ø2ØØØ
ØØ154                                  Ø5ØØØØ  ØØØØ1Ø
```

The first addition was executed. Perform the remaining additions by looping:

```
B,I,42
R,35
I P= ØØØ42  I=Ø62164  A=Ø5ØØØØ  B=ØØ2154  E=Ø  O=Ø  MA=ØØ164  MC=ØØØØØ2
```

Check final total in ANSW.

```
D,B,154,155
DUMP--BASE = Ø2ØØØ
 ØØ154                                Ø5ØØØØ  ØØØØ1Ø
```

Here, if not previously, the error should be detected; the program does not perform more than the first addition. The label sum has been placed in the wrong instruction. It should be in location 27 preceding the DLD .MON,I instruction.

APPENDIX A
HP CHARACTER SET

			b_7	0	0	0	0	1	1	1	1
			b_6	0	0	1	1	0	0	1	1
			b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC$_0$	ƀ	0	@	P	UNASSIGNED	UNASSIGNED
0	0	0	1	SOM	DC$_1$	!	1	A	Q		
0	0	1	0	EOA	DC$_2$	"	2	B	R		
0	0	1	1	EOM	DC$_3$	#	3	C	S		
0	1	0	0	EOT	DC4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	' (APOS)	7	G	W		
1	0	0	0	FE$_0$	S$_0$	(	8	H	X		
1	0	0	1	HT / SK	S$_1$	)	9	I	Y		
1	0	1	0	LF	S$_2$	*	:	J	Z		
1	0	1	1	V$_{TAB}$	S$_3$	+	;	K	[		
1	1	0	0	FF	S$_4$	, (COMMA)	<	L	\		ACK
1	1	0	1	CR	S$_5$	−	=	M	]		①
1	1	1	0	SO	S$_6$	.	>	N	↑		ESC
1	1	1	1	SI	S$_7$	/	?	O	←		DEL

Standard 7-bit set code positional order and notation are shown below with b_7 the high-order and b_1 the low-order, bit position.

	b_7	b_6	b_5	b_4	b_3	b_2	b_1
EXAMPLE: The code for "R" is:	1	C	1	0	0	1	0

LEGEND

NULL	Null/Idle
SOM	Start of message
EOA	End of address
EOM	End of message
EOT	End of transmission
WRU	"Who are you?"
RU	"Are you...?"
BELL	Audible signal
FE$_0$	Format effector
HT	Horizontal tabulation
SK	Skip (punched card)
LF	Line feed
V$_{TAB}$	Vertical tabulation
FF	Form feed
CR	Carriage return
SO	Shift out
SI	Shift in
DC$_0$	Device control reserved for data link escape
DC$_1$-DC$_3$	Device Control
DC$_4$(Stop)	Device control (stop)
ERR	Error
SYNC	Synchronous idle
LEM	Logical end of media
S$_0$-S$_7$	Separator (information)
ƀ	Word separator (space, normally non-printing)
<	Less than
>	Greater than
↑	Up arrow (Exponentiation)
←	Left arrow (Implies/Replaced by)
\	Reverse slant
ACK	Acknowledge
①	Unassigned control
ESC	Escape
DEL	Delete/Idle

APPENDIX A

BINARY CODED DECIMAL FORMAT

Kennedy 1406/1506 ASCII-BCD Conversion

Symbol	BCD (octal code)	ASCII Equivalent (octal code)	Symbol	BCD (octal code)	ASCII Equivalent (octal code)
(Space)	20	040	A	61	101
!	52	041	B	62	102
#	13	043	C	63	103
$	53	044	D	64	104
%	34	045	E	65	105
&	60	046	F	66	106
'	14	047	G	67	107
(	34	050	H	70	110
)	74	051	I	71	111
*	54	052	J	41	112
+	60	053	K	42	113
,	33	054	L	43	114
-	40	055	M	44	115
.	73	056	N	45	116
/	21	057	O	46	117
			P	47	120
0	12	060	Q	50	121
1	01	061	R	51	122
2	02	062	S	22	123
3	03	063	T	23	124
4	04	064	U	24	125
5	05	065	V	25	126
6	06	066	W	26	127
7	07	067	X	27	130
8	10	070	Y	30	131
9	11	071	Z	31	132
:	15	072	[	75	133
;	56	073	\	36	134
<	76	074	]	55	135
=	13	075			
>	16	076			
?	72	077			
@	14	100			

Other symbols which may be represented in ASCII are converted to spaces in BCD (20)

APPENDIX A

HP 2020 ASCII - BCD Conversion

Symbol	ASCII (octal code)	BCD (octal code)
(Space)	40	20
!	41	52
"	42	37
#	43	13
$	44	53
%	45	34
&	46	60*
'	47	36
(	50	75
)	51	55
*	52	54
+	53	60
,	54	33
-	55	40
.	56	73
/	57	21
0	60	12
1	61	01
2	62	02
3	63	03
4	64	04
5	65	05
6	66	06
7	67	07
8	70	10
9	71	11
:	72	15
;	73	56
<	74	76
=	75	35
>	76	16
?	77	72
@	100	14
A	101	61
B	102	62
C	103	63
D	104	64
E	105	65
F	106	66
G	107	67
H	110	70
I	111	71
J	112	41
K	113	42
L	114	43
M	115	44
N	116	45
O	117	46
P	120	47
Q	121	50
R	112	51
S	123	22
T	124	23
U	125	24
V	126	25
W	127	26
X	130	27
Y	121	30
Z	132	31
[	133	75**
]	135	55**
↑	136	77
←	137	32

*BCD code of 60 always converted to ASCII code 53 (+).

**BCD code of 75 always converted to ASCII code 50 (() and BCD code of 55 always converted to ASCII code 51 ()).

HP 2761A-007 Mark Sense Card Reader

Data read from Mark Sense Cards is converted from the same Hollerith codes used for punched cards to ASCII codes (octal equivalents) and packed two characters per computer word. The first character and every other character after it are placed in the upper half (bits 15 thru 8) of successive words in the buffer. The second character and every other character after it are placed in the lower half (bits 7 thru 0) of those same successive words. Thus, each character has the potential of either one of two representations in a computer word, depending on its position within the reading sequence. Both of these potentials are listed for each character available from Mark Sense Cards in Table A-3, starting below.

For example, if the word HEMP were being read the ASCII octal equivalent for H as the first character is 044000, which is stored as

```
15          87          0
| 0 100 100 000 000 000 |
```

Next, the ASCII octal equivalent for E as the second character is 000105, which is stored as

```
15          87          0
| 0 100 100 001 000 101 |
```

The first packed computer word then, is

```
15          87          0
| 0 100 100 001 000 101 |
```

Finally, the next two characters M (046400) and P (000120) are stored in the next packed computer word as

```
15          87          0
| 0 100 110 101 010 000 |
```

APPENDIX A

MARK SENSE CARD READER CHARACTER CONVERSIONS

Hollerith or ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
A	040400	000101
B	041000	000102
C	041400	000103
D	042000	000104
E	042400	000105
F	043000	000106
G	043400	000107
H	044000	000100
I	044400	000111
J	045000	000112
K	045400	000113
L	046000	000114
M	046400	000115
N	047000	000116
O	047400	000117
P	050000	000120
Q	050400	000121
R	051000	000122
S	051400	000123
T	052000	000124
U	052400	000125
V	053000	000126
W	053400	000127
X	054000	000130
Y	054400	000131
Z	055000	000132
0	030000	000060
1	030400	000061
2	031000	000062
3	031400	000063
4	032000	000064
5	032400	000065

APPENDIX A

MARK SENSE CARD READER CHARACTER CONVERSIONS (cont.)

Hollerith or ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
6	033000	000066
7	033400	000067
8	034000	000070
9	034400	000071
(space)	020000	000040
!	020400	000041
" *(quote)*	021000	000042
#	021400	000043
$	022000	000044
%	022400	000045
&	023000	000046
' *(apostrophe)*	023400	000047
(	024000	000050
)	024400	000051
*	025000	000052
+	025400	000053
, *(comma)*	026000	000054
- *(hyphen or minus)*	026400	000055
. *(period)*	027000	000056
/	027400	000057
:	035000	000072
;	035400	000073
<	036000	000074
=	036400	000075
>	037000	000076
?	037400	000077
@	040000	000100
¢ *(cent)* or [	055400	000133
(not mark) or]	056400	000135

MARK SENSE CARD READER CHARACTER CONVERSIONS (cont.)

Hollerith or ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
| (vertical bar*) or ↑	057000	000136
_ (underscore**) or ←	057400	000137
0-8-2 or \	056000	000134

*NUMERIC Y

**NUMERIC W

APPENDIX B
EQUIPMENT TABLE

The Equipment Table (EQT) provides information for the input/output control routine, .IOC., and the equipment driver subroutines. The table contains an entry for each peripheral device attached to the computer configuration.

The table is constructed as a block of entries assembled by the Prepare Control System routine. The first word of the table, at the symbolic entry point .EQT., contains the number of entries in the table. An entry in the table is referenced according to its position. The numbers 1 through 6 are reserved for standard units. (See "Standard Unit Equipment Table," Appendix C.) The number 7_8 appearing in a program refers to the first table entry; the number 10_8, the second, and so forth. The numbers may be in the range 7_8-74_8 with the largest value being determined by the number of units of equipment available at the installation.

The four-word entry for each device contains the following information:

- The channel number of the device (10_8-76_8)
- A Direct Memory Access channel indicator, if pertinent
- Absolute address of equipment driver subroutine
- Equipment type identification code.

The above information is static for each installation; it is not altered by .IOC. The entry also contains dynamic information which is supplied by the equipment driver subroutine. This information includes:

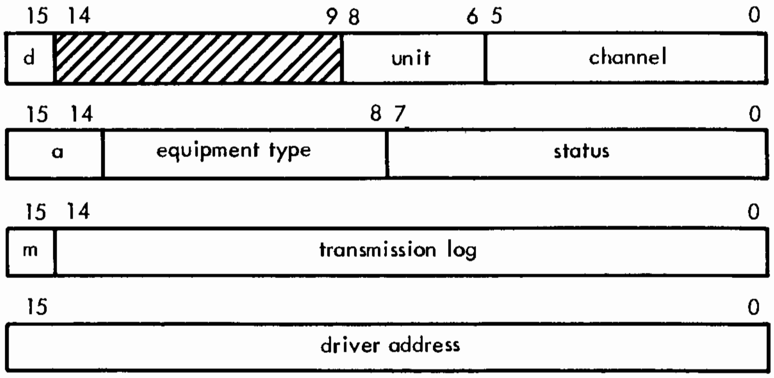
a. Status of operation (i.e., in progress or complete)

b. Status of equipment

c. Number of characters or words transmitted when the operation is completed.

EQUIPMENT TABLE

The format of the entry is as follows:



d = Direct Memory Access channel indicator

1 DMA channel is to be used for each data transmission operation

0 DMA channel not required

Unit = Physical unit number (0-7) used to address the device if it is attached to a multi-unit controller.

Channel = The channel number (select code) for the physical device (also the low core location containing a JSB to the related interrupt subroutine.)

a = Availability of device:

0 The device is available; the previous operation is complete.

1 The device is available; the previous operation is complete but a transmission error has been detected.

2 The device is not available for another request; the operation is in progress.

Equipment type = This field contains a 6-bit code that identifies the device:

00-07 -- Paper Tape devices

00 Teleprinter

01 Paper Tape Reader

02 Tape Punch

10-17 -- Unit Record devices

15 Mark Sense Reader

EQUIPMENT TABLE

Equipment type = (cont.)

20-37 -- Magnetic Tape and Mass Storage devices

- 20 Kennedy 1406 Incremental Tape Transport
- 21 2020 Magnetic Tape Unit
- 22 3030 Magnetic Tape Unit

40-77 -- Instrumentation devices

- 40 Data Source Interface
- 41 Integrating Digital Voltmeter
- 42 Guarded Crossbar Scanner
- 43 Time Base Generator

Status = The status field indicates the actual status of the device when the data transmission is complete. The contents depend on the type of device. (See Status Table at the end of this appendix.)

m = This bit defines the mode of the data transmission:

0 ASCII or BCD

1 Binary

Transmission log = This field is a log of the number of characters or words transmitted. The value is given as a positive integer and indicates characters or words as specified in the calling sequence. The value is stored in this field only when the input/output request has been completed, therefore, when all data is transmitted or when a transmission error is detected.

Driver address = Absolute address of the entry point for the associated driver subroutine for the device.

STATUS TABLE

Status Bit Device	7	6	5	4	3	2	1	0
Teleprinter			End of Input Tape					
Paper Tape Reader			End of Tape					
Tape Punch			Tape Supply Low					
Kennedy 1406 Incremental Tape Transport			End of Tape		BT			DB
2020 Magnetic Tape Unit	EOF	ST	End of Tape	TE	I/O R	NW	PA	DB
3030 Magnetic Tape Unite	EOF	ST	End of Tape	TE	I/O R	NW	PA	DB
Mark Sense Reader						PF	RNR	HE or SF

BT = Broken Tape

DB = Device Busy

EOF = End of File

ST = Start of Tape

TE = Timing Error

I/OR = I/O Reject

NW = No Write (write enable ring missing or tape unit is rewinding)

PA = Parity Error

HE = Hopper Empty

SF = Stacker Full

RNR = Reader Not Ready

PF = Pick Fail

APPENDIX C
STANDARD UNIT EQUIPMENT TABLE

The Standard Unit Equipment Table (SQT) allows reference to input/output devices designated as standard units. The table contains 6 one-word entries. Each entry corresponds to a particular standard unit and contains a pointer to the Equipment Table. The standard units are as follows:

Number	Name
1	Keyboard Input
2	Teleprinter Output
3	Relocatable Library
4	Punch Output
5	Input
6	List Output

The number defines the position in the SQT at which the device is listed. Each Standard unit may be a different device, or a single physical device may represent several Standard units. The value of the pointer in the SQT is the position of the physical unit's entry in the EQT, with the lowest value being 7_8.

APPENDIX D
IOC WITH OUTPUT BUFFERING

IOC with Output Buffering is an extension of the standard version and provides automatic stacking and buffering of all output and function requests. This involves moving an output request and associated buffer into available memory and adding the request location into a thread of stacked requests for the referenced unit. At the completion of an output operation, the next entry in the stack for the unit is initiated by IOC. Output/function requests for a paraticular unit are processed according to the order of the requests (first in/first out).

This version of IOC requires the use of the program MEMRY to perform the allocation and release of blocks of available memory. If available memory is exhausted when an allocation is attempted, IOC repeats the call until space is made available, i.e., previous blocks are released.

PRIORITY OUTPUT

A "priority" write or function request has been added for use with the Buffered version of IOC. A priority request is processed immediately without the request and buffer being moved to available memory. The current operation in the stack is suspended, the priority request processed and the suspended operation re-initiated. The priority feature is useful for writing messages or diagnostics for immediate action or for performing output without reserving a segment of available memory for request/buffer storage. (All output performed by the BCS Relocating Loader is done as priority requests for the latter reason.) If two or more priority requests are called in immediate succession (without intervening status checks), the last requested operation is performed with the previous ones being "lost."

A priority request (i.e., Write function) is indicated by setting bit 09 of Word 2 of the request call = 1. Bit 09 = 0 means normal operation with the Standard IOC and means the request will be stacked and buffered with the extended version.

IOC WITH OUTPUT BUFFERING

Example: Priority Write to teleprinter:

```
JSB  .IOC.
OCT  21002
JMP  REJ
DEF  BUFFER
DEC  -37
```

OPERATING ENVIRONMENT

IOC with Output Buffering provides for writing a data block on more than one output device in parallel and does not restrict output rates to the lowest speed device. Because all requests and buffers are moved into available memory for subsequent processing, peak load output processing is not delayed due to device speed or saturated buffer storage within the bounds of user programs. System I/O saturation occurs when available memory is exhausted.

RESTRICTIONS

The routines used to allocate/release blocks in available memory and to initiate stacked output requests operate with the Interrupt System disabled. Therefore, the use of medium/high speed synchronous I/O devices (e.g., HP 2020 Magnetic Tape) under program control is not recommended because of possible data loss.

An I/O driver routine operating under the extended version of IOC may not be used to control more than one like device. This is because the buffering control routine in IOC only checks for stacked requests referencing the unit on which an operation was just completed.

HALT CONDITIONS

Irrecoverable error conditions are identical to the standard version of IOC. The location of the halt is at the entry point IOERR. These conditions are:

A-Register	B-Register	Meaning
0	Location at Request	Request Code Illegal
1	Location at Request	Unit Reference Illegal
0	0	Write request for an input only device.

I/O ERROR CONDITIONS

The routine .BUFR in the version of IOC with Output Buffering checks for error conditions of the end of each output operation. If any error conditions, End-of-Tape or Tape Supply Low, etc. conditions are present, IOC halts to allow the condition to be corrected. Processing is continued by pressing RUN.

Halt: (T) = 102070
(A) = Word 2 of EQT entry (Status word)
(B) = Hardware I/O address of unit

An addition has been made to this routine to handle requests for buffered output of records too long to be buffered with the amount of memory available. If such a request is made, the following occurs:

a. IOC outputs the contents of any buffers which have been previously "stacked" for the referenced I/O device.

b. The computer halts to inform the user that his program cannot buffer output records of the length requested. The contents of the registers are as follows:

(T) = 102001
(A) = Maximum length record that can be buffered with the amount of memory available.
(B) = Memory location of the output request which caused the halt.

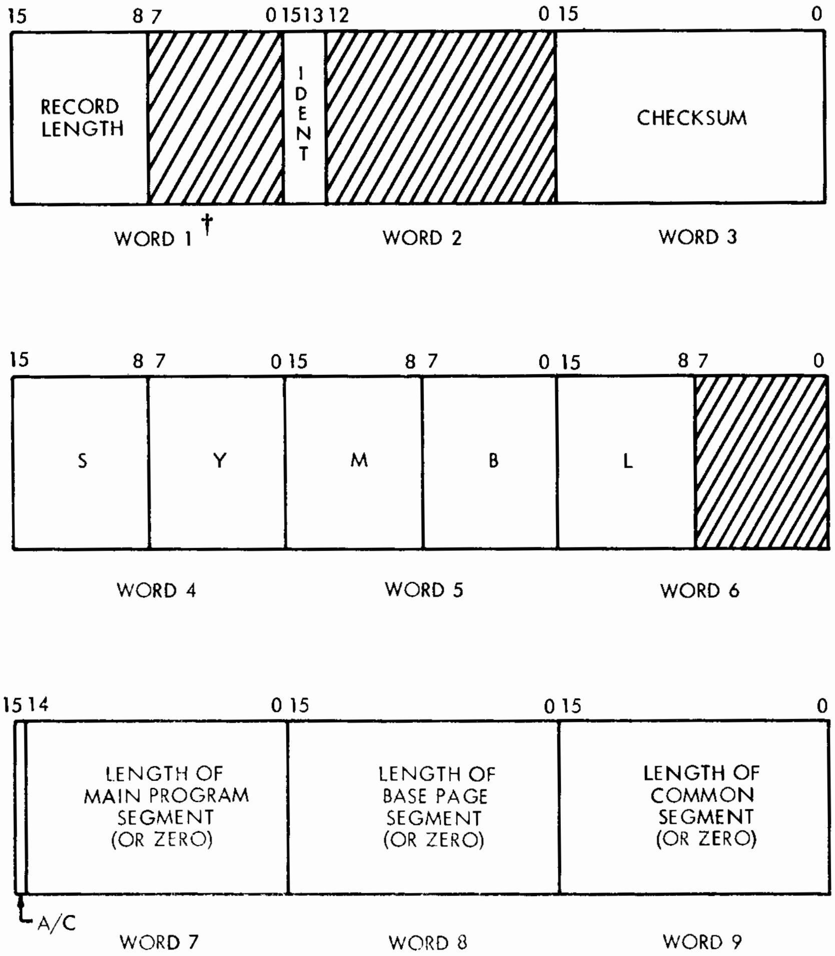
The user restarts the program by pushing the RUN button. The output request is honored immediately without buffering. IOC waits until the output operation is complete before returning control to the program. This ensures that the data area is not modified before the complete record is output, and that the output results are identical to those produced if buffered output of the record had occurred.

APPENDIX E
RELOCATABLE TAPE FORMAT

NAM RECORD

CONTENT

EXPLANATION



RECORD LENGTH = 9 WORDS

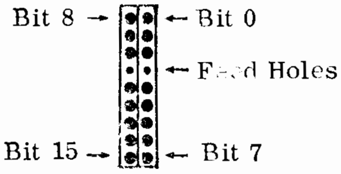
IDENT = 001

CHECKSUM: ARITHMETIC TOTAL OF ALL WORDS IN RECORD EXCLUDING WORDS 1 AND 3.

SYMBL: FIVE CHARACTER NAME OF PROGRAM

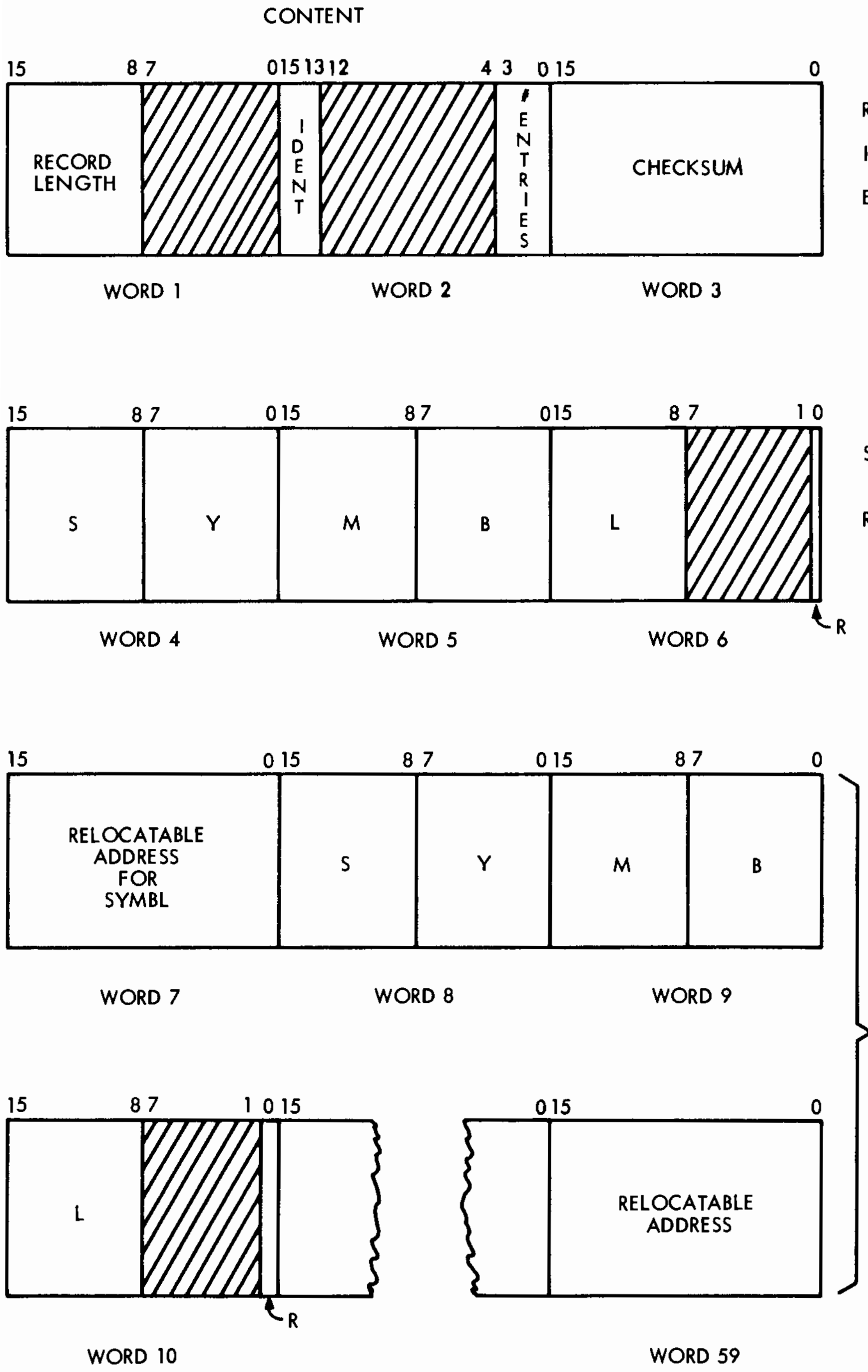
A/C: BINARY TAPE PROCESSOR
= 0 IF ASSEMBLER PRODUCED
= 1 IF COMPILER PRODUCED

†Each word represents two frames arranged as follows:



RELOCATABLE TAPE FORMAT

ENT RECORD



EXPLANATION

RECORD LENGTH = 7-59 WORDS

IDENT = 010

ENTRIES: 1 to 14 ENTRIES PER PROGRAM; EACH ENTRY IS FOUR WORDS LONG.

SYMBL: 5 CHARACTER ENTRY POINT SYMBOL

R: RELOCATION INDICATOR
= 0 IF PROGRAM RELOCATABLE
= 1 IF BASE PAGE RELOCATABLE

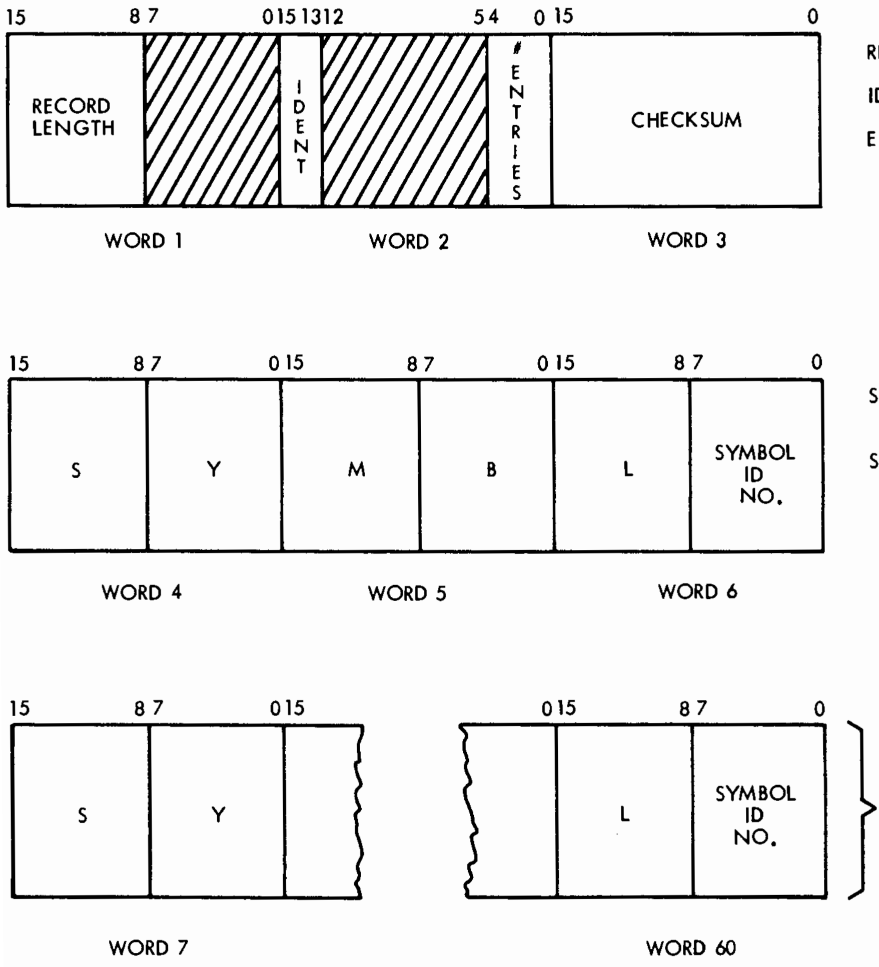
WORDS 4 THROUGH 7 ARE REPEATED FOR EACH ENTRY POINT SYMBOL.

RELOCATABLE TAPE FORMAT

EXT RECORD

CONTENT

EXPLANATION



RECORD LENGTH = 6-60 WORDS

IDENT = 100

ENTRIES: 1 TO 19 PER RECORD; EACH ENTRY IS THREE WORDS LONG

SYMBL: 5 CHARACTER EXTERNAL SYMBOL

SYMBOL ID. NO.: NUMBER ASSIGNED TO SYMBL FOR USE IN LOCATING REFERENCE IN BODY OF PROGRAM.

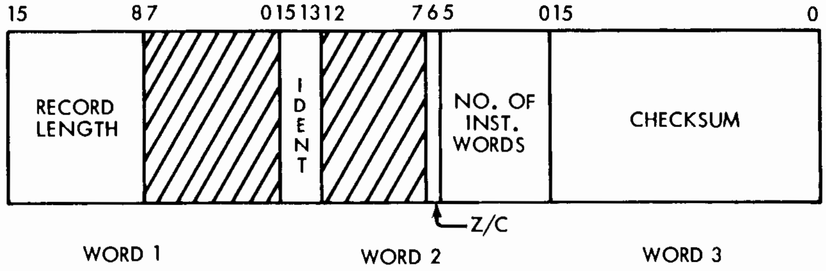
WORDS 4 THROUGH 6 REPEATED FOR EACH EXTERNAL SYMBOL (MAXIMUM OF 19 PER RECORD).

RELOCATABLE TAPE FORMAT

DBL RECORD

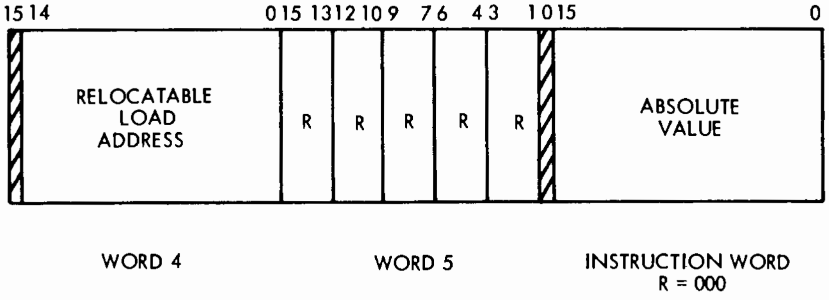
CONTENT

EXPLANATION



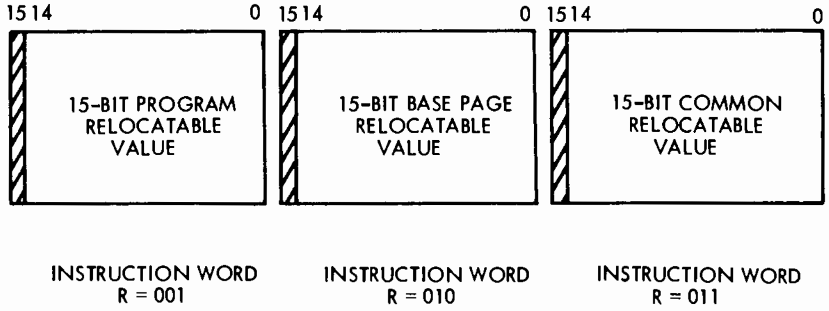
RECORD LENGTH = 5-60 WORDS
IDENT = 011
Z/C: BASE/CURRENT PAGE LOADING
= 0 FOR BASE PAGE
= 1 FOR CURRENT PAGE

NO. OF INST. WORDS: 1 TO 45 LOADABLE INSTRUCTION WORDS PER RECORD



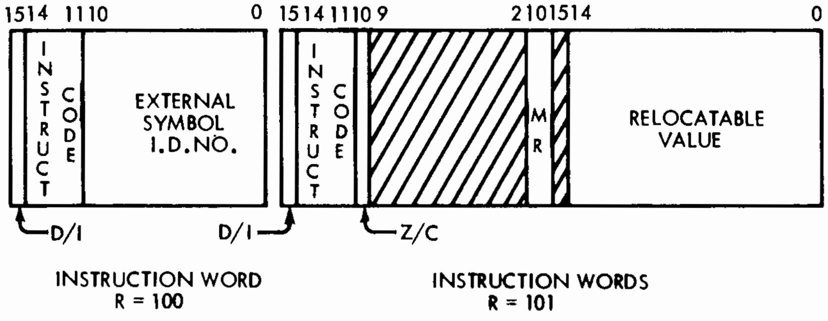
RELOCATABLE LOAD ADDRESS: STARTING ADDRESS FOR LOADING THE INSTRUCTIONS WHICH FOLLOW.

R's: RELOCATION INDICATORS:
000 = ABSOLUTE
001 = 15-BIT PROGRAM RELOCATABLE
010 = 15-BIT BASE PAGE RELOCATABLE
011 = 15-BIT COMMON RELOCATABLE
100 = EXTERNAL REFERENCE
101 = MEMORY REFERENCE


15 14 0 15 14 0 15 14 0
15-BIT PROGRAM RELOCATABLE VALUE
15-BIT BASE PAGE RELOCATABLE VALUE
15-BIT COMMON RELOCATABLE VALUE
INSTRUCTION WORD R = 001
INSTRUCTION WORD R = 010
INSTRUCTION WORD R = 011

R_1 IS RELOCATION INDICATOR FOR INSTRUCTION $WORD_1$; R_2, FOR INSTRUCTION $WORD_2$; ETC-MEMORY REFERENCE INSTRUCTIONS USE TWO WORDS, WITHIN THE TWO-WORD GROUP, "MR" INDICATES RELOCATABILITY OF OPERAND SPECIFIED IN SECOND WORD:

00 = PROGRAM RELOCATABLE
01 = BASE PAGE RELOCATABLE
10 = COMMON RELOCATABLE



D/I: INDIRECT ADDRESSING

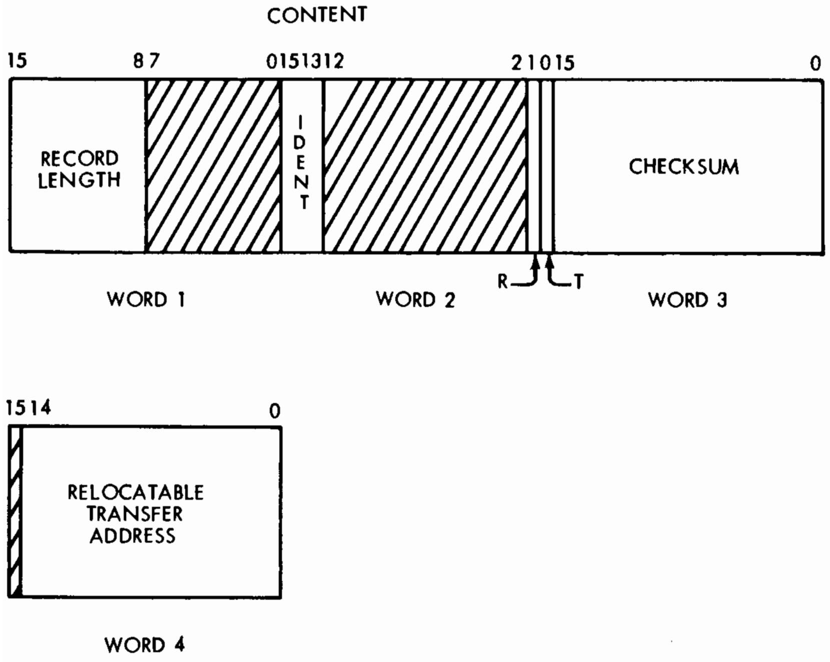
0 = DIRECT
1 = INDIRECT

Z/C: BASE/CURRENT PAGE LOCATION OF OPERAND ADDRESS AS DETERMINED BY LOADER.

0 = BASE PAGE
1 - CURRENT PAGE

RELOCATABLE TAPE FORMAT

END RECORD



EXPLANATION

RECORD LENGTH = 4 WORDS

IDENT = 101

R: RELOCATION INDICATOR
FOR TRANSFER ADDRESS

= 0 IF PROGRAM RELOCATABLE
= 1 IF BASE PAGE
RELOCATABLE

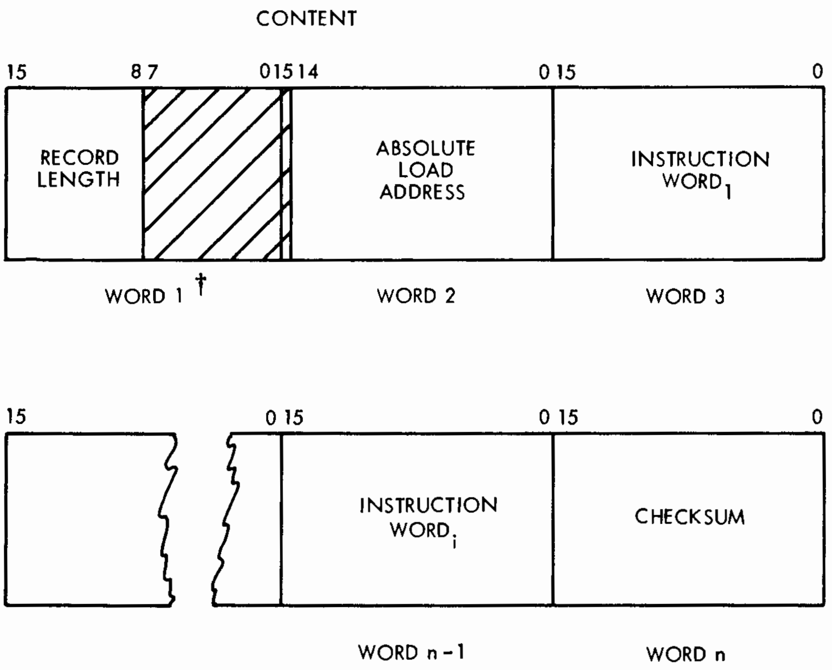
T: TRANSFER ADDRESS
INDICATOR

= 0 IF NO TRANSFER
ADDRESS IN RECORD

= 1 IF TRANSFER ADDRESS
PRESENT

APPENDIX F

ABSOLUTE TAPE FORMAT



EXPLANATION

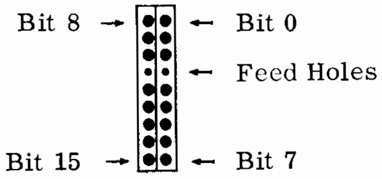
RECORD LENGTH = NUMBER OF WORDS IN RECORD EXCLUDING WORDS 1 AND 2 AND THE LAST WORD.

ABSOLUTE LOAD ADDRESS: STARTING ADDRESS FOR LOADING THE INSTRUCTIONS WHICH FOLLOW

INSTRUCTION WORDS: ABSOLUTE INSTRUCTIONS OR DATA

CHECKSUM: ARITHMETIC TOTAL OF ALL WORDS EXCEPT FIRST AND LAST

†Each word represents two frames arranged as follows:



HEWLETT hp PACKARD

READER COMMENT SHEET

BASIC CONTROL SYSTEM

HP 02116—9017 December 1971

Hewlett-Packard welcomes your evaluation of this text. Any errors, suggested additions, deletions, or general comments may be made below. Use extra pages if you like.

CUT ALONG LINE

FROM PAGE__OF__

NAME: ______________________________

ADDRESS: ______________________________

____

NO POSTAGE NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AS SHOWN ON OTHER SIDE AND TAPE

FOLD FOLD

FIRST CLASS
PERMIT NO.141
CUPERTINO
CALIFORNIA

BUSINESS REPLY MAIL

No Postage Necessary if Mailed in the United States Postage will be paid by

MANAGER, SOFTWARE PUBLICATIONS
HEWLETT - PACKARD
CUPERTINO DIVISION
11000 Wolfe Road
Cupertino, California
95014

FOLD FOLD