digital

MONITOR SYSTEMS 10/20, 10/30

10

PDP-10

PDP-10 SYSTEMS 10/20,10/30 MONITOR Programmer's Reference Manual

JANUARY 1968

Order No. DEC-10-MKB0-D Price $1.00

Direct comments concerning this manual to Software Quality Control, Digital Equipment Corp., 146 Main St., Maynard, Massachusetts, 01754

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

Original Printing
June 1967

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CHAPTER 1
INTRODUCTION

The Single-User Monitor for PDP-10/20 and PDP-10/30 Systems increases throughput by performing fast job-to-job transitions; makes programming easier by servicing I/O commands for all standard devices; allows the computer user to assign I/O devices at run time; and provides upward compatibility with Time-Sharing Systems 10/40 and 10/50.

System facilities range from a minimum system with 8K of core (16K for a PDP-10/30) and two DECtapes to larger systems with up to 262K of core and/or magnetic tapes, discs, displays, card readers, line printers, paper tape readers and punches, and plotters. Nonstandard devices are easily interfaced to the system. Monitors are customized at each installation to include only the I/O service routines required by the configuration. The Monitor is provided to users as a set of distinct subprograms; another program (called System Builder) accepts system specifications from the Teletype and generates the user's customized Monitor. During this process, users may replace any I/O service routines with their own specialized versions and add service routines for nonstandard devices. These subprograms become part of the Monitor and allow use of the same I/O calls for both standard and nonstandard devices.

1.1 INCREASING THROUGHPUT

Many more programs are processed during the work day when running under Monitor control. When one program ends, commands typed by the operator initialize and start the next program.

The Batch Processor (see "PDP-10 Batch Processor" library writeup) makes program-to-program transitions even faster. Programs to be executed are loaded sequentially on an input medium; the Batch Processor controls the execution of each job, automatically handles job-to-job transitions, postmortem dumps, and error messages.

An important Monitor function is to control and synchronize all input/output. The Monitor takes optimum advantage of the priority interrupt system to efficiently overlap I/O with computation. All devices are made to appear identical and interchangeable. Nevertheless, full control of the interrupt system and of individual device facilities is available to any user program.

1.2 EASIER PROGRAMMING

Programming is easier because I/O commands are handled by the Monitor, and are completely device independent. The programmer is concerned only with logical structure of his I/O files; the Monitor handles the physical details for any standard device in the system. Since I/O commands are device independent, there is no problem if a particular device is not available; the operator simply assigns another standard device.

The operator is always in control. He can interrupt a long, low-priority job, save it, and load and run the long job later, all by typing a few commands to Monitor. In order to continue execution, the user program must provide his own-coding to reinitialize his I/O devices.

1.3 UPWARD COMPATIBILITY

Programs running under Monitor control are compatible with Batch Processor, and with Time-Sharing Systems 10/40 and 10/50. This makes it possible to plan for future expansion of system facilities, with little or no reprogramming.

1.4 OPERATING THE MONITOR

When the system is first loaded (from tape or disk), the console Teletype is in command mode, i.e., is communicating directly with the Monitor. When execution of a job program is started (or resumed) by the commands START, CONT, DDT or REENTER, the Teletype becomes available to the user program as an I/O device. The Teletype is put back in command mode by the program by calling EXIT, or by the user typing ↑C on the Teletype.[1] All system programs exit by returning control to a Monitor subprogram which asks the user what he wants to do next.

Among the commands available to the user are the following:

RUN	Brings a program into core from a specified device and starts its execution.
GET	Brings a program into core, but does not start it.
SAVE	Copies the content of core onto a specified output device. This allows the user to continue working at some other time. It also allows him to save a program on the system device for use by others.
START	Gives control to the program at the starting location specified by the Linking Loader.
ASSIGN	Allows the user to change the I/O devices a program uses without changing the program. For example, if a program was written to output data on the Teletype (mnemonic name TTY) and the user decided to output on the line printer (mnemonic name LPT) instead, the user could assign the logical name TTY to the line printer by typing ASSIGN LPT TTY. Thereafter the Monitor would direct all Teletype output to the line printer.
DEASSIGN	Releases the device to which the logical name is assigned.
CONT	Continues execution of the program in core from its last interruption point. Thus the user could stop his program, save it for backup, and continue operation where he left off.
REENTER	Restarts an interrupted program at a point previously specified by the programmer.

[1]To type ↑C, hold down the CTRL key while striking "C".

DDT	Starts the Dynamic Debugging Technique program (DDT) to allow the user to examine and modify cells and exercise step-by-step control over the execution of his program.
DATE	Converts the date typed by the user to the standard internal format and stores it in the system date location.
INIT	Initializes the I/O package for the user. Releases every device (except the system library device) to which a logical name has been assigned.
KJOB	Releases every active device.

1.5 MODULAR DESIGN

The Monitor is written in two sections: a Console Monitor (CONMON) which interprets and executes commands typed by the user; and an I/O package which performs the actual I/O for the user.

File-structured devices, such as DECtapes, contain information which is not written in contiguous blocks. These devices, therefore, require a directory which contains information about the location of files contained on the device.

1.6 CORE LAYOUT

The core layout is shown in Figure 1-1. The I/O package resides in upper core, ending at the highest cell in the machine. The Console Monitor is immediately below the I/O package. In lower core is a job-data area, which contains parameters describing both the state of the machine and the user's program currently in core.

1.7 USER'S OWN-CODING I/O SERVICE ROUTINES FOR REAL-TIME DEVICES

The PDP-10 can be programmed to service and process incoming data from real-time devices such as high-speed communication multiplexers, photographic digitizers, and analog converters.

Such real-time systems can be effectively implemented using various combinations of user/Digital software. Three approaches are:

a. User's real-time program running as an independent job under Monitor. The real-time program would perform its own I/O, and Monitor would run other jobs under priorities established by the real-time program to utilize the processor fully.

b. The user's real-time program could be fully integrated into the Monitor system as a high-priority foreground program. The Monitor would respond to input and output commands from the real-time program, and could run independent background programs to use all available processor time.

c. A real-time device service routine can be written and made a part of the Monitor at system build time. Real-time programs may then be written as normal user pro-

programs, calling on the Monitor to do all physical I/O; the principle of device independence is thus maintained even for the specified real-time operations.

For a detailed analysis of real-time programming in PDP-10 systems, see Real-Time Use of the PDP-10 Monitors, DEC-10-NMAB-D.

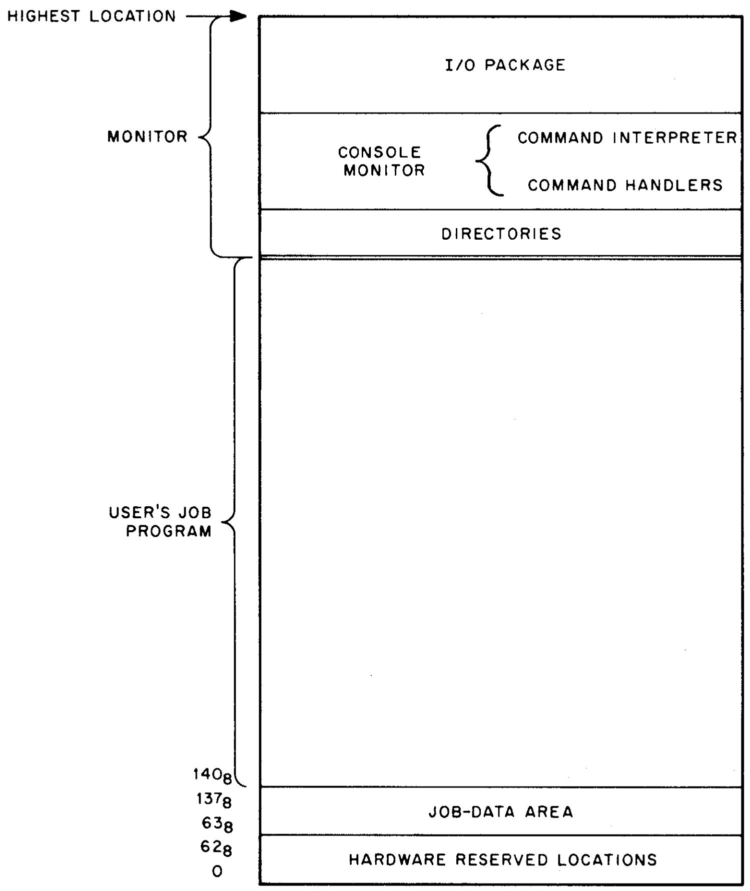

Figure 1-1 Core Layout

CHAPTER 2
SETTING UP A JOB PROGRAM

After a job program has been assembled or compiled, it must be loaded by the Linking Loader, which places the starting address and other pertinent information in the job data area.

The job-data area consists of locations 66 through 137 in lower core and contains items of information concerning the job to be executed. Some of this information is entered by the Linking Loader, some by Monitor and some by the programmer. Each item uses a single word, referenced by the symbols shown below.

2.1 JOB DATA ENTERED BY MONITOR

JOBREL	Contains the address of the highest location which can legally be accessed by a user program. Above this location are the Monitor and DECtape directories.
DATE	Contains the date typed by the programmer. Contains zero if no date is typed.
JOBOPC	Contains the address of the location at which the program was interrupted by typing ↑C. It points to the next instruction to be executed.

2.2 JOB DATA SET BY THE LINKING LOADER

JOBSA	The right half contains the starting address of the job; the left half contains the program break.
JOBFF	Contains the address of the current program break (updated by generation of the I/O buffers.)
JOBDDT	Contains the starting address of DDT, or zero.
JOBSYM	The right half points to the first location of the symbol table generated by the Assembler, loaded by the Linking Loader, and used by DDT during symbolic debugging. The left half contains the length of the symbol table, negated.
JOBUSY	Points to the first location in the undefined global symbol table. DDT may be used to define the values in this table.

2.3 JOB DATA ENTERED BY THE PROGRAMMER

JOBREN	An alternate entry point taken with REENTER is typed.
JOB41	Monitor transfers control to the user's I/O routine pointed to by this location when an unused operation code (codes 001-037) is encountered in the program. The UUO is stored in location JOBUUO.

CHAPTER 3
MONITOR COMMANDS

3.1 COMMAND FORMAT

Each direct Monitor command consists of a line of ASCII characters, terminated by a carriage return. Command names and arguments are delimited by any printing character, including "space," which is not a letter or a digit. Command names may be abbreviated to the first two letters, as shown below:

Table 3-1 Monitor Commands

Command	Arguments 1	Arguments 2	Console Mode
ASSIGN (AS)	dev	ldev^o	m
CONT (CO)			u
↑C			m
DATE (DA)	mm-dd-yy		m
DDT (DD)			u
DEASSIGN (DE)	ldev		m
GET (GE)	ldev^o	progname	m
INIT (IN)			m
KJOB (KJ)			m
REENTER (RE)			u
RUN (RU)	ldev^o	progname	u
SAVE (SA)	ldev^o	progname	m
START (ST)			u

xxx^o	optional argument	mm-dd-yy	month, day, year
dev	physical device name (CDR, PTR, PTP, MTAn, DTAn, DSK, CTY, TTY, TTYn, LPT, SYS)	progname	a program name (6 characters or less)
ldev	a physical or logical device name	m	Monitor mode
		u	user mode

3.2 COMMAND EXECUTION

The Monitor indicates its readiness to accept a command by typing a point (.). It then waits for input from the user's Teletype. When a command is typed, Monitor interprets the command and takes the required action. If the command cannot be interpreted and processed correctly, Monitor types a question mark.

In the following descriptions of the commands, arguments shown in parentheses are optional. The command, ↑C, is typed by depressing the CTRL key, and then typing "C".

3.2.1 INIT

The INIT command initializes the I/O package for the user. The logical name of every device (except SYS) is released. The directory space of every device which has a logical name is released, and JOBREL is set equal to the first location below the Monitor.

3.2.2 ASSIGN dev (ldev)

The programmer uses the ASSIGN command to assign logical names for I/O devices appearing in his job program to physical devices available at run time. This command is typed in the general form,

ASSIGN (physical-name) (logical-name)

where the physical device name is set by the hardware configuration (and may be changed by switches on DECtape and magnetic tape units) and logical names are programmer assigned.

The following physical device names are recognized by PDP-10 Monitors:

I/O Device	Physical Name	
DECtapes	DTA0 - DTA_{n-1}	where n is number of DECtapes or magnetic tapes in the system
Magnetic tapes	MTA0 - MTA_{n-1}	
Paper tape reader	PTR	
Paper tape punch	PTP	
Line printer	LPT	
Card reader	CDR	
Disk	DSK	
Teletype	TTY	

The assignment of logical device names is not absolutely necessary; physical names are acceptable in Monitor commands. Logical names have the added advantage of giving the programmer an option to choose or change device assignments at run time. Logical names take precedence over physical device names.

Assume a job program uses three I/O files: FILE1, FILE2 and FILE3. FILE1 is an input from a permanent master on DECtape, FILE2 is a scratch tape, and FILE3 is a hard copy output on either the line printer or a Teletype. At run time, the line printer is temporarily unavailable, so the programmer types:

```
ASSIGN  DTA2  FILE1
ASSIGN  MTA2  FILE2
ASSIGN  TTY   FILE3
```

so that the hard copy is printed on the Teletype.

When the physical device is a DECtape, the ASSIGN command clears the copy of the directory currently in core, forcing any new directory references to read a new copy from the tape. It is especially important that an ASSIGN be used when changing reels. If directory space has not yet been allocated for this device, 200 locations are provided for the directory ending at the contents of JOBREL, which is then updated.

If a directory reference is made to a DECtape which has not been assigned, the directory is read into the 200 locations ending at the contents of JOBREL, writing over any information which was previously stored there. If the reference requires that a copy of the directory remain in core, this space is permanently taken.

3.2.3 DEASSIGN dev

The DEASSIGN command releases the logical name assigned to the device named in the argument. If the device is a DECtape which was allocated directory space, the space is released, and JOBREL is updated.

3.2.4 RUN (dev) prognm

The RUN command zeros core from the job data area to the location pointed to by JOBREL, then loads the named program (prognm) into core from the device specified in the first argument, and starts the program at the location specified by JOBSA. If only one argument is specified, it is assumed to be the program name, and the device is assumed to be SYS.

If a job was saved with fewer DECtapes assigned than are normally assigned when it is to be run, that portion of the file which should be loaded above the current contents of JOBREL is ignored.

3.2.5 GET (dev) prognm

GET works exactly like the RUN command, except that the job is not started. Control remains in the Console Monitor.

3.2.6 SAVE (dev) prognm

SAVE copies the contents of the user's core area and part of the job data area onto device dev with the name PROGNM.SAV. If DDT is loaded (that is, if JOBDDT is nonzero), the area of core from JOBDDT in the job data area to the location pointed to be JOBREL is saved. Otherwise, if

JOBDDT is zero, Monitor saves core from JOBDDT to the program break, specified by the contents of JOBFF. The state of the user's I/O devices is not saved. Locations with zero contents are not written out, which is the reason for zeroing core before loading the program by a GET or RUN command. Accumulators are saved.

When a SAVEd job is reloaded (by GET), the state of the user's I/O devices is reset. To continue execution of an interrupted job, the user must reinitialize all needed I/O devices. The job can then be continued through a JRST 2, @ JOBOPC. If no I/O devices must be initialized, the user could simply GET the job and type CONT (see Section 3.2.9).

3.2.7 START

START gives control to a job at its starting address, as specified by the right half of JOBSA.

3.2.8 ↑C

The programmer stops execution of his running program by typing ↑C on the Teletype. Control returns to Monitor. The location at which the job was interrupted is stored in JOBOPC.

3.2.9 CONT

To continue an interrupted job, CONT starts the program from its last interruption point, specified by JOBOPC.

3.2.10 DDT

This command starts execution of the program at the location specified by JOBDDT. This location is set to the starting address of DDT by the Linking Loader.

3.2.11 REENTER

This command restarts an interrupted job at a point specified by the programmer in JOBREN. The programmer can return to the main program interruption point by executing a JRST 2,@JOBOPC.

A typical use of REENTER is for reinitializing the I/O devices of a program after a SAVEd job has been reloaded. The routine would resume the interrupted program by executing

```
JRST 2,@JOBOPC
```

If the corresponding job data word is zero when a transfer of control command (START, CONT, REENTER, or DDT) is typed, the Monitor will type a question mark.

3.2.12 KJOB (Kill Job)

Kills the job and releases every active device.

3.2.13 DATE mm-dd-yy

This command converts the date typed by the programmer to the standard internal format and stores it in the system date location.

CHAPTER 4
USER PROGRAMMING

4.1 PROGRAMMED OPERATORS

Operation codes 000 through 077, called programmed operators or UUOs (unused operations), are used by the programmer in his job program to call on Monitor (or his own-coding special I/O routine), for I/O requests and control functions.

4.1.1 User UUOs

Operation codes 001 through 037 are interpreted as user UUOs by the Monitor, which switches control to the user's own-coding at the location pointed to by the address of JOB41. This transfer path might be used, for example, to service I/O commands for special devices. The user assigns an op code (001 - 037) and does his own interpreting and handling. The Monitor executes a JSR @ JOB41 provided that JOB41 is nonzero. If JOB41 is zero, it is an illegal UUO.

4.1.2 Monitor UUOs

Operation codes 040 through 077 are interpreted by the Monitor as calls for service. These are shown, with their mnemonics, in Table 4-1.

Table 4-1 Monitor Operation Codes

Mnemonic	Op Code	Function
CALL	040	Additional mnemonic Monitor commands (see Table 4-2)
INIT	041	Initialize I/O device
CALLI	047	Same additional Monitor commands as CALL, but uses octal codes (see Table 4-2)
OPEN	050	Open file
RENAME	055	Rename or delete file
IN	056	Input and skip
OUT	057	Output and skip
SETSTS	060	Set file status
STATO	061	Skip on file status one
GETSTS	062	Read file status
STATZ	063	Skip on file status zero
INBUF	064	Set up input buffer ring

Table 4-1 Monitor Operation Codes (cont)

Mnemonic	Op Code	Function
OUTBUF	065	Set up output buffer ring
INPUT	066	Read
OUTPUT	067	Write
CLOSE	070	Close file
RELEASE	071	Release device
MTAPE	072	Position tape
UGETF	073	Get next free block number
USETI	074	Set next input block number
USETO	075	Set next output block number
LOOKUP	076	Select file
ENTER	077	Create file

Note: Op codes 042-046 and 051-054 are illegal. 042-046 are reserved for system modifications by users; 051-054 are reserved for possible future additions by Digital.

4.1.3 CALL and CALLI

Operation codes 040 through 077 limit the Monitor to 40_8 operations. The CALL operation extends this set by specifying the name of the operation by the contents of the location specified by the effective address, such as

CALL [SIXBIT /EXIT/]

This provides for indefinite extendability of Monitor operations, at the overhead cost to the Monitor of a table lookup.

The CALLI operation eliminates the table lookup of the CALL operation by having the programmer insert a value from Table 4-2. CALL operations require an additional storage word. CALLI operations have the advantage of not requiring an additional word. Table 4-2 lists the Monitor operations specified by the CALL and CALLI operations.

Table 4-2 CALL and CALLI Monitor Operations

CALLI AC, x	CALL AC, [SIXBIT/y/] Y	Function
x= 0	y= RESET	Reset I/O devices
1	DDTIN	DDT mode console input
3	DDTOUT	DDT mode console output
4	DEVCHR	Get device characteristics
10	WAIT	Wait until device inactive
12	EXIT	Release devices, stop job
13	UTPCLR	Clear directory
14	DATE	Return date
20	SWITCH	Read processor console switches
2, 5-7, 11, 15-17, 21-31	If time-sharing arguments such as CORE are used, a no-op results.	No operation
≥ 32		Illegal UUO

4.1.4 CALL [SIXBIT /APRENB/]

Note that CALL [SIXBIT /APRENB/], or CALLI 16, is no operation in the 10/20, 10/30 Monitor. There is no way to cause the Monitor to give control to a user program on an APR generated interrupt.

4.2 ERRORS

When the Monitor detects an error, such as an illegal UUO, an error message

ERROR__________

followed by a mnemonic describing the error is typed on the user's Teletype.

The UUO causing the error is stored in location 40. Location 41 contains a JSR to location UUO0, whose address field is one higher than the erroneous UUO. Thus, to find the illegal UUO and its location, locations 40 and UUO0 should be displayed. Note that this must be done with the DISPLAY key on the PDP-10 console, since DDT uses UUOS, and therefore changes the contents of 40 and UUO0.

The error messages and their meanings are listed in Table 4-3.

Table 4-3 Error Messages

Error Message	Meaning
ADRES	Insufficient core storage in which to assign I/O buffers (this error occurs when the contents of JOBFF is greater than the contents of JOBREL).
IOP	Unexplainable error in the I/O package section of the Monitor.
NO IN	An INPUT UUO was issued for a device which can only do output.
NO OU	An OUTPUT UUO was issued for a device which can only do input.
MODE	An INIT was issued for a device in a transmission mode which is not compatible with the selected device (e.g., dump mode on a Teletype).
CHAN	An I/O UUO was issued for a channel which was not initialized by an INIT command.
UUO	An illegal UUO, such as CALLI 1000, was executed.
DIREC	An error occurred during reading or writing of a directory.
PDL	Pushdown list overflow.
NXM	Nonexistent memory trap.

Although pushdown list overflow and nonexistent memory are not UUOs, the error handler will store the location at which they occurred in the right half of UUO0 for compatibility with the other errors.

4.3 PROGRAM CONTROL

All job programs running under Monitor should terminate with either a CALL [SIXBIT /EXIT/] or a CALLI 12 to release all opened input/output devices and stop the job.

```
EXIT
↑C
```

is printed on the user's console, which is left in Monitor mode.

CALL AC, [SIXBIT /DATE/] or CALLI AC, 14

A 12-bit binary integer computed by the formula:

$$date = ((year-1964) \times 12 + (month-1)) \times 31 + day - 1$$

represents the date.

This integer representation is returned right justified in accumulator AC.

CALL AC, [SIXBIT /SWITCH/] or CALLI AC, 20

These commands return the contents of the central processor data switches in AC.

4.4 INPUT/OUTPUT PROGRAMMING

All user input/output operations are controlled by the use of Monitor programmed operators. These are device independent, in the sense that if an operation is not pertinent to a given device, the operation is treated as a no-op. For example, a REWIND directed to a line printer does nothing. Devices are referenced by logical names or physical names (see ASSIGN command, Chapter 3), and the characteristics of a device can be obtained from the Monitor. Properly used, these system characteristics permit the programmer to delay the device specification for his program from program generation time until program run time.

4.4.1 File

A file is an ordered set of data on a peripheral device. Its extent on input is determined by an end-of-file condition dependent on the device. For instance, a file is terminated by reading an end-of-file gap from magnetic tape, by an end-of-file card from a card reader, or by depressing the end-of-file switch on a card reader. The extent of a file on output is determined by the amount of information written by the OUT or OUTPUT programmed operators up through and including the next CLOSE or RELEASE operator.

4.4.1.1 Device - To specify a file, it is necessary to specify the device from which the file is to be read or onto which the file is to be written. This specification is made by an argument of the INIT or OPEN programmed operators. Devices are separated into two categories--those with no filename directory (nondirectory devices), and those with one or more filename directories (directory devices).

a. Nondirectory Devices - For nondirectory devices, e.g., card reader, line printer, paper tape reader and punch, and user console, the only file specification required is the device name. All other file specifiers, if given, are ignored by the Monitor. Magnetic tape, which is also a nondirectory device, requires, in addition to the name, that the tape be properly positioned.

b. Directory Devices - For directory devices, (DECtape and disk), files are addressable by name. If the device has a single file directory, e.g., DECtape, the device name and filename are sufficient information to determine a file. If the device has multiple file directories, e.g., disk, the name of the file directory must also be specified. These names are specified as arguments to the LOOKUP, ENTER, and RENAME programmed operators.

4.4.1.2 Data Modes - Data transmissions are either unbuffered (dump) or buffered. The mode of transmission is specified by a 4-bit argument to the INIT, OPEN, or GETSTS programmed operators. Table 4-4 summarizes the data modes.

Table 4-4 Data Transmission Modes

Octal Code	Meaning
Buffered Modes	
0	ASCII. 7-bit characters packed left justified, five characters per word (see appendix 3).
1	ASCII line. Same as 0, except that the buffer is terminated by a form-feed (FORM), vertical tab (VT), line-feed (LINE FEED) or ALTMODE character.
2-7	Unused.
10	Image. A device dependent mode. The buffer is filled with data exactly as supplied by the device.
11-12	Unused.
13	Image. binary. 36-bit bytes. This mode is similar to binary mode, except that no automatic formatting or checksumming is done by the Monitor.
14	Binary. 36-bit byte. This is a blocked format consisting of a word count, N (the right half of the first data word of the buffer), followed by N 36-bit data words. Checksumming is done for cards and paper tape.
Unbuffered Modes	
15	Image Dump. A device-dependent dump mode.
16	Dump respecting record boundaries. Data is transmitted between any contiguous block of core and one record on the device.
17	Dump. Same as mode 16, except that record boundaries are ignored on input and arranging the data into records is automatic on output.

a. Unbuffered Data Modes - Data modes 15, 16, and 17 utilize a command list to specify areas in the user's allocated core to be read or written. The effective address of the IN, INPUT, OUT, and OUTPUT programmed operators points to the first word of the command list. Three types of entries may occur in the command list.

(1) IOWD N, LOC - Causes the N words from LOC through LOC+N-1 to be transmitted. The next command is obtained from the next location following the IOWD.

(2) XWD 0,Y - Causes the next command to be taken from location Y.

(3) 0 - Terminates the command list.

The Monitor does not return program control to the user until the command list has been completely processed. If an illegal address is encountered while processing the list, the job is stopped, the Monitor prints

ERROR - ADRES

on the user's console and leaves the console in Monitor mode.

b. Buffered Data Modes - Data modes 0, 1, 10, 13, and 14 utilize a ring of buffers and the priority interrupt system to permit the user to overlap computation with his data transmission. Core memory in the user's area serves as an intermediate buffer between the user's program and the device. A ring of buffers consist of a 3-word header block for bookkeeping and a data storage area subdivided into one or more individual buffers linked together to form a ring. During input operations, the Monitor fills a buffer, makes the buffer available to the user's program, advances to the next buffer in the ring and fills it if it is free. The user's program follows along behind, emptying the next buffer if it is full, or waiting for the next buffer to be filled. During output operations, the user's program and the Monitor exchange roles, the user filling the buffers and the Monitor emptying them.

(1) Buffer Structure - A ring of buffers consists of a 3-word header block and a data storage area subdivided into one or more individual buffers linked together to form a ring. The ring buffer layout is shown in Figure 4-1, and explained in the paragraphs which follow.

(a) Buffer Header Block - The location of the 3-word buffer header block is specified by an argument of the INIT and OPEN operators. Information is stored in the header by the Monitor in response to user execution of Monitor programmed operators. The user's program finds all the information required to fill and empty buffers in the header. Bit position 0 of the first word of the header is a flag which, if 1, means that no input or output has occurred for this ring of buffers. The right half of the first word is the address of the second word of the buffer currently in use by the user's program. The second word of the header contains a byte pointer to the next byte in the current buffer. The byte size is determined by the data mode. The third word of the header contains a count of the number of bytes remaining in the buffer.

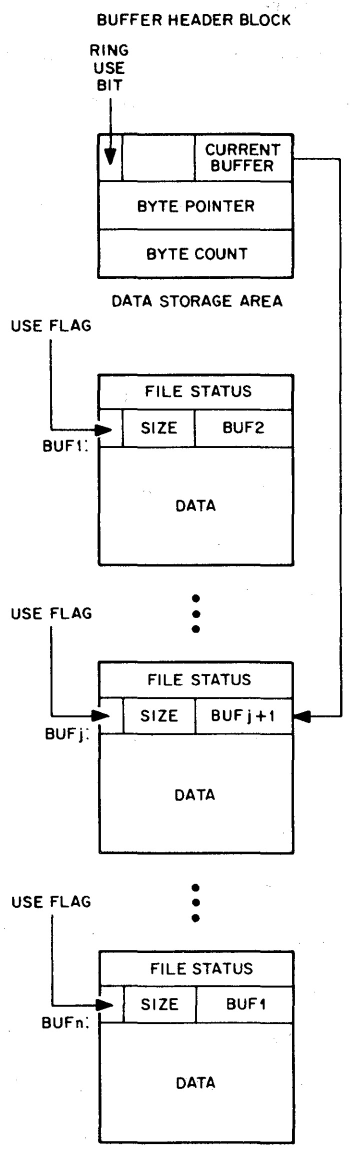

Figure 4-1 Ring of Buffers

(b) Buffer Data Storage Area - The buffer data storage area is established by the INBUF and OUTBUF operators, or, if none exists when the first IN, INPUT, OUT, or OUTPUT operator is executed, a 2-buffer ring is set up. The effective address of the INBUF and OUTBUF operators specifies the number of buffers in the ring. The location of the buffer data storage area is specified by the contents of the right half of JOBFF in the user's job data area. The Monitor updates JOBFF to point to the first location past the storage area.

All buffers in the ring are identical in structure. As Figure 4-2 shows, the right half of the first word contains the file status at the time that the Monitor advanced to the next buffer in the ring. Bit 0 of the second word of a buffer, called the use bit, is a flag that indicates whether the buffer contains active data. This bit is set to 1 by the Monitor when the buffer is full on input or being emptied on output, and set to 0 when the buffer is empty on output or is being filled on input. The use bit prevents the Monitor and the user's program from interfering with each other by attempting to use the same buffer simultaneously. Bits 1 through 17 of the second word of the buffer contain the size of the data area of the buffer which immediately follows the second word. The size of this data area depends on the device. The right half of the second word of the buffer contains the address of the second word of the next buffer in the ring.

The right half of the first word of the data area of the buffer (i.e., the third word of the buffer) is reserved for a count of the number of words (excluding itself) that actually contain data. The left half of this word is reserved for other bookkeeping purposes, depending on the particular device and the data mode.

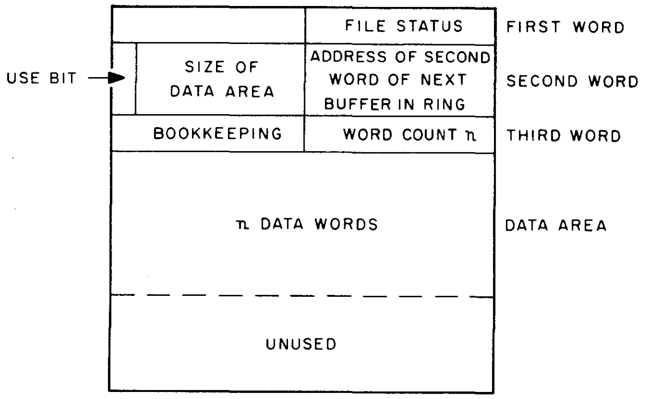

Figure 4-2 Monitor Buffer

4.4.1.3 File Status - The file status is a set of 18 bits (right half word), which reflects the current state of a file transmission. The initial status is a parameter of the INIT and OPEN operators. Thereafter, bits are set by the Monitor, and may be tested and reset by the user via Monitor programmed operators. Table 4-5 defines the file status bits.

Table 4-5 File Status

Bit	Meaning
18	Improper mode, such as an attempt to write on a write-locked tape.
19	Device detected parity error.
20	Data error, such as a computed checksum failed or invalid data was received.
21	Block too large. A block of data from a device is too large to fit in a buffer.
22	End of file.
23	Device is actively transmitting or receiving data.
24-29	Device dependent parameters (see Chapter 5).
30	Synchronous input. Stop the device after each buffer is filled.
31	Forces the Monitor to use the word count in the first data word of the buffer (output only). The Monitor normally computes the word count from the byte pointer in the buffer header.
32-35	Data transmission mode. See Table 4-4.

4.4.2 Initialization

4.4.2.1 Job Initialization - The Monitor programmed operator

```
          CALL [ SIXBIT/RESET/]  or CALLI 0
```

should be the first instruction in each job program. It immediately stops all input/output transmissions on all devices without waiting for the devices to become inactive. All device allocations made by the INIT and OPEN operators are cleared. The content of the left half of JOBSA (program break) is stored in the right half of JOBFF. The left half of JOBFF is cleared.

4.4.2.2 Device Initialization

```
OPEN D,SPEC                      INIT D,STATUS
error return                     SIXBIT /Idev/
normal return                    XWD OBUF, IBUF
     .                           error return
     .
     .
SPEC:EXP STATUS                  normal return
     SIXBIT /Idev/
     XWD OBUF,IBUF
```

The OPEN (operation code 050) and INIT (operation code 041) programmed operators initialize a file by specifying a device, ldev, and initial file status, STATUS, and the location of the input and output buffer headers.

a. Data Channel - OPEN and INIT establish a correspondence between the device, ldev, and a 4-bit data channel number, D. Most of the other input/output operators require this channel number as an argument. If a device is already assigned to channel D, it is released (see RELEASE, this chapter). The device name, ldev, is either a logical or physical name, with logical names taking precedence over physical names (see ASSIGN command, Chapter 3). If the device, ldev, does not exist, the error return is taken.

b. Initial File Status - The file status, including the data mode, is set to the value of the symbol STATUS. If the data mode is not legal for the specified device, the job is stopped and the Monitor prints

   ERROR - MODE

   on the user's console, leaving the console in Monitor mode.

c. Buffer Header - Symbols OBUF and IBUF, if nonzero, specify the location of the first word of the 3-word buffer header for output and input respectively. Only those headers which are to be used need to be specified. For instance, the output header need not be specified, if only input is to be done. Also, modes 15, 16, and 17 require no header.

   The first and third words of the buffer header are set to zero. The left half of the second word is set up with the byte pointer size field in bits 6 through 11 for the selected device-data mode combination.

4.4.2.3 Buffer Initialization - Buffer data storage areas may be established by the INBUF and OUTBUF programmed operators, or by the first IN, INPUT, OUT, or OUTPUT operator, if none exists at that time, or the user may set up his own buffer data storage area.

a. Monitor Generated Buffers - Each device has associated with it a standard buffer size. The Monitor programmed operators INBUF D, N (operation code 064) and OUTBUF D, N (operation code 065) set up a ring of N standard size buffers associated with the input and output buffer headers, respectively, specified by the last OPEN or INIT operator on data channel D. If no OPEN or INIT operator has been performed on channel D, the Monitor stops the job, prints

   ERROR - CHAN

   on the user's console, and leaves the console in Monitor mode.

The storage space for the ring is taken from successive locations, beginning with the location specified in the right half of JOBFF. This is set to the program break, which is the first free location above the program area, by RESET. If there is insufficient space to set up the ring the Monitor will stop the job, print

ERROR - ADRES

on the user's console, and leave the console in Monitor mode.

The ring is set up by setting the second word of each buffer with a zero use bit, the appropriate data area size, and the link to the next buffer. The first word of the buffer header is set with a 1 in the ring use bit, and the right half contains the address of the second word of the first buffer.

b. User-Generated Buffers - The following coding illustrates an alternative to the use of the INBUF programmed operator. Analogous code may replace OUTBUF. This user coding operates similarly to INBUF. SIZE must be set equal to the greatest number of data words expected in one physical record.

```
GO:       INIT 1, 0                       ;INITIALIZE ASCII MODE
          SIXBIT/MTA0/                    ;MAGNETIC TAPE UNIT 0
          XWD 0, MAGBUF                   ;INPUT ONLY
          JRST NOTAVL
          MOVE 0, [XWD 400000,BUF1 +1]    ;THE 400000 IN THE LEFT HALF MEANS THE
                                          ;BUFFER WAS NEVER REFERENCED.
          MOVEM 0, MAGBUF
          MOVE 0, [POINT BYTSIZ,0,35]     ;SET UP NONSTANDARD BYTE SIZE
          MOVEM 0, MAGBUF+1
          JRST CONTIN                     ;GO BACK TO MAIN SEQUENCE
MAGBUF:   BLOCK 3                         ;SPACE FOR BUFFER HEADER
BUF1:     0                               ;BUFFER 1, 1ST WORD UNUSED
          XWD SIZE+1,BUF2+1               ;LEFT HALF CONTAINS BUFFER SIZE,
                                          ;RIGHT HALF HAS ADDRESS OF NEXT BUFFER
          BLOCK SIZE+1                    ;SPACE FOR DATA, 1ST WORD RECEIVES
                                          ;WORD-COUNT. THUS ONE MORE WORD
                                          ;IS RESERVED THAN IS REQUIRED
                                          ;FOR DATA ALONE
BUF2:     0                               ;SECOND BUFFER
          XWD SIZE+1,BUF3+1
          BLOCK SIZE+1
BUF3:     0                               ;THIRD BUFFER
          XWD SIZE+1,BUF1 +1              ;RIGHT HALF CLOSES THE RING
          BLOCK SIZE+1
```

4.4.2.4 File Selection - The LOOKUP (operation code 076) and ENTER (operation code 077) programmed operators select a file for input and output respectively. Although these operators are not necessary for nondirectory devices, it is good programming practive to always use them so that directory devices may be substituted at run time. (See ASSIGN, Chapter 3.)

a. LOOKUP D,E

```
error return
normal return
     .
     .
     .
E:   SIXBIT /file/          ;file name, 1 to 6 characters
     SIXBIT /ext/           ;file name extension, 0 to 3 characters
     0
     0
```

LOOKUP selects a file for input on channel D. If the input side of channel D is not closed (see CLOSE, in this chapter), it is now closed. The output side of channel D is not affected. If the device associated with channel D does not have a directory, the normal return is now taken.

The device directory is searched for the file whose name is in location E and whose extension is in the left half of location E + 1. If the file is not found the error return is taken. Otherwise, location E + 1, and E + 2, are filled by the Monitor with the following data concerning the file, and the normal return is taken.

(1) The right half of location E+1 is set to the device block number of the first block of the file.

(2) Bits 24 through 35 of location E+2 are set to the date of the file's creation, i.e., of the last ENTER or RENAME programmed operator.

b. ENTER D,E

```
error return
normal return
     .
     .
     .
E:   SIXBIT /file/          ;filename, 1 through 6 characters
     SIXBIT /ext/           ;filename extension, 0 through 3 characters .
     DATE
     0
```

ENTER selects a file for output on channel D. If the output side of channel D is not closed (see CLOSE in this chapter), it is now closed. The input side of channel D is not affected. If the device does not have a directory, the normal return is now taken.

The device file directory is searched for the file whose name is in location E and whose extension is in the left half of location E + 1.

If the file is found, then the file is deleted, and the storage space on the device is recovered.

The Monitor then makes the file entry by recording the following informatior concerning the file and takes the normal return.

(1) The filename is taken from location E.

(2) The filename extension is taken from the left half of location E + 1.

(3) If bits 24 through 35 of location E+2 are nonzero, they are taken as the date of creation; otherwise, the current date is used.

```
RENAME D,E
error return
normal return
    :
E:  SIXBIT /file/        ;file name, 1 through 6 characters
    SIXBIT /ext/         ;file name extension, 0 through 3 characters.
    DATE
```

The RENAME programmed operator (operation code 055) is used to alter the filename, the filename extension, or to delete a file on a directory device.

If no device is associated with channel D, the error return is taken. If the device is a nor directory device, the normal return is taken. If no file is currently selected on channel D the error return is taken.

The device file directory is searched for the file currently selected on channel D. If the file is not found the error return is taken.

If the new filename in location E is zero, the file is deleted and the normal return is taken

If a new filename in location E and/or the filename extension in the left half of location E + 1 differ from the current filename and/or filename extension the device directory is searched for t new filename and extension, as in LOOKUP. If a match is found the error return is taken. If no ma is found, the file is changed to the new name in location E, the filename extension is changed to the new filename extension in the left half of location E + 1 and the normal return is taken.

4.4.2.5 Examples

General Device Initialization

```
INIDEV:        0                         ;JSR HERE
               INIT 3, 14                ;BINARY MODE,CHANNEL 3
               SIXBIT/DTA5/              ;DEVICE DECTAPE UNIT 5
               XWD OBUF,IBUF             ;BOTH INPUT AND OUTPUT
               JRST NOTAVL               ;WHERE TO GO IF DTA5 IS BUSY

;FROM HERE DOWN IS OPTIONAL DEPENDING ON THE DEVICE AND PROGRAM
;REQUIREMENTS
               MOVE 0, JOBFF
               MOVEM 0,SV JBFF           ;SAVE THE FIRST ADDRESS OF THE BUFFER
                                         ;RING IN CASE THE SPACE MUST BE
                                         ;RECLAIMED.
               INBUF 3, 4                ;SET UP 4 INPUT BUFFERS
               OUTBUF 3, 1               ;SET UP 1 OUTPUT BUFFER
               LOOKUP 3, INNAM           ;INITIALIZE AN INPUT FILE
               JRST NOTFND               ;WHERE TO GO IF THE INPUT FILENAME IS
                                         ;NOT IN THE DIRECTORY
               ENTER 3, OUTNAM           ;INITIALIZE AN OUTPUT FILE
               JRST NOROOM               ;WHERE TO GO IF THERE IS NO ROOM IN
                                         ;THE DIRECTORY FOR A NEW FILENAME.
               JRST @ INIDEV             ;RETURN TO MAIN SEQUENCE
OBUF:          BLOCK 3                   ;SPACE FOR OUTPUT BUFFER HEADER
IBUF:          BLOCK 3                   ;SPACE FOR INPUT BUFFER HEADER
INNAM:         SIXBIT/NAME/              ;FILE NAME
               SIXBIT/EXT/               ;FILE NAME EXTENSION (OPTIONALLY 0),
                                         ;RIGHT HALF WORD RECEIVES THE
                                         ;FIRST BLOCK NUMBER
               0                         ;RECEIVES THE DATE
               0                         ;UNUSED FOR NONDUMP I/O
OUTNAM:        SIXBIT/NAME/              ;SAME INFORMATION AS IN INNAME
               SIXBIT/EXT/
               0
               0
```

4.4.3 Data Transmission

The programmed operators

```
        INPUT D,E       and       IN D,E
                                  normal return
                                  error return
```

transmit data from the file selected on channel D to the user's core area. The programmed operators

```
        OUTPUT D,E      and       OUT D,E
                                  normal return
                                  error return
```

transmit data from the user's core area to the file selected on channel D.

If no OPEN or INIT operator has been perfomed on channel D, the Monitor stops the job and prints

ERROR - CHAN

on the user's console leaving the console in Monitor mode. If the device is a multiple-directory device and no file is selected on channel D, bit 18 of the file status is set to 1, and control returns to the user's program. Control always returns to the location immediately following an INPUT (operation code 066) and an OUTPUT (operation code 067). A check of the file status for end-of-file and error conditions must then be made by another programmed operator. Control returns to the location immediately following an IN (operation code 056) and an OUT (operation code 057), if no end-of-file or error condition exists, i.e., if bits 18 through 22 of the file status are all zero. Otherwise, control returns to the second location following the IN or OUT.

4.4.3.1 Unbuffered (Dump) Modes - In data modes 15, 16, and 17 the effective address E of the INPUT, IN, OUTPUT, and OUT programmed operators is the address of the first word of a command list (see section 4.4.1). Control does not return to the program until the transmission is terminated or an error is detected.

a. Example

Dump Output

Dump input is similar to dump output. This routine outputs fixed-length records.

```
DMPINI:     0                       ;JSR HERE TO INITIALIZE A FILE
            INIT 0, 16              ;CHANNEL 0, DUMP MODE
            SIXBIT/MTA2/            ;MAGNETIC TAPE UNIT 2
            0                       ;NO RING BUFFERS
            JRST NOTAVL             ;WHERE TO GO IF UNIT 2 IS BUSY
            JRST @ DMPINI           ;RETURN
DMPOUT:     0                       ;JSR HERE TO OUTPUT THE OUTPUT AREA
            OUTPUT 0, OUTLST        ;SPECIFIES DUMP OUTPUT ACCORDING
                                    ;TO THE LIST AT OUTLIST
            STATZ 0, 740000         ;CHECK ERROR BITS
            CALL [ SIXBIT/EXIT/]    ;QUIT IF AN ERROR OCCURS
            JRST @ DMPOUT           ;RETURN
DMPDON:     0                       ;JSR HERE TO WRITE AN END OF FILE
            CLOSE 0,                ;WRITE THE END OF FILE
            STATZ 0, 740000         ;CHECK FOR ERROR DURING WRITE
                                    ;END OF FILE OPERATION
            CALL [SIXBIT/EXIT/]     ;QUIT IF ERROR OCCURS
            RELEAS 0,               ;RELINQUISH THE DEVICE
            JRST @ DMPDON           ;RETURN
OUTLST:     IOWD BUFSIZ,BUFFER      ;SPECIFIES DUMPING A NUMBER OF
                                    ;WORDS EQUAL TO BUFSIZ, STARTING
                                    ;AT LOCATION BUFFER
            0                       ;SPECIFIES THE END OF THE COMMAND LIST
BUFFER:     BLOCK BUFSIZ            ;OUTPUT BUFFER, MUST BE CLEARED
                                    ;AND FILLED BY THE MAIN PROGRAM
```

4.4.3.2 Buffered Modes - In data modes 0, 1, 10, 13, and 14 the effective address E of the INPUT, IN, OUTPUT, and OUT programmed operators may be used to alter the normal sequence of buffer reference. If E is zero, the address of the next buffer is obtained from the right half of the second word of the current buffer. If E is nonzero, it is the address of the second word of the next buffer to be referenced. The buffer pointed to by E can be in an entirely separate ring from the present buffer. Once a new buffer location is established, the following buffers are taken from the ring started at E.

a. Input - If no input buffer ring is established when the first INPUT or IN is executed, a 2-buffer ring is set up. (See INBUF, this chapter.)

Buffered input may be performed synchronously or asynchronously at the option of the user. If bit 30 of the file status is 1, each INPUT and IN programmed operator

(1) Clears the use bit in the second word of the buffer whose address is in the right half of the first word of the buffer header, thereby making it available for refilling;

(2) Advances to the next buffer by setting the address of the second word of the buffer in the right half of the first word of the buffer header;

(3) If an end-of-file or an error condition exists, control is returned to the user's program. Otherwise, the Monitor starts the device which fills the buffer and stops transmission;

(4) Computes the number of bytes in the buffer from the number of words in the buffer (right half of the first data word of the buffer) and the data mode, and stores the result in the third word of the buffer header;

(5) Sets the position and address fields of the byte pointer in the second word of the buffer header, so that the first data byte is obtained by an ILDB instruction; and

(6) Returns control to the user's program.

Thus, in synchronous mode, the position of a device, such as magnetic tape, relative to the current data is easily determined. The asynchronous input mode differs in that once a device is started, successive buffers in the ring are filled at the interrupt level without stopping transmission until a buffer whose use bit is 1 is encountered. Control returns to the user's program after the first buffer is filled. The position of the device relative to the data currently being processed by the user's program depends on the number of buffers in the ring and when the device was last stopped.

(1) Example

General Input-One-Character Subroutine

```
GETCHR:   0                     ;JSR HERE
          SOSLE IBUF+2          ;DECREMENT THE BYTE COUNT
          JRST GETOK            ;DATA IS ALREADY IN THE BUFFER
          INPUT 3,              ;REFILL THE BUFFER
          STATZ 3, 740000       ;CHECK THE 4 ERROR BITS
          JRST INERR            ;WHERE TO GO ON AN ERROR
          STATZ 3, 20000        ;CHECK END OF FILE BIT
          JRST ENDFIL           ;WHERE TO GO ON END OF FILE
GETOK:    ILDB AC,IBUF+1        ;FETCH CHARACTER FROM THE BUFFER
          JRST @GETCHR          ;RETURN.
```

b. Output - If no output buffer ring has been established, i.e., if the first word of the buffer header is zero, when the first OUT or OUTPUT is executed, a 2-buffer ring is set up (see OUTBUF, this chapter). If the ring use bit (bit 0 of the first word of the buffer header) is 1, it is set to 0, the current buffer is cleared to all zeros, and the position and address fields of the buffer byte pointer (the second word of the buffer header) are set so that the first byte is properly stored by an IDPB instruction. The byte count (the third word of the buffer header) is set to the maximum of bytes that may be stored in the buffer, and control is returned to the user's program. Thus, the first OUT or OUTPUT initializes the buffer header and the first buffer, and does not result in data transmission.

If the ring use bit is 0 and bit 31 of the file status is 0, the number of words in the buffer is computed from the address field of the buffer byte pointer (the second word of the buffer header) and the buffer pointer (the first word of the buffer header), and the result is stored in the right half of the first data word of the buffer. If bit 31 of the file status is 1, it is assumed that the user has already set the word count in the right half of the first data word. The buffer use bit (bit 0 of the second word of the buffer) is set to 1, indicating that the buffer contains data to be transmitted to the device. If the device is not currently active, i.e., not receiving data, it is started. The buffer header is advanced to the next buffer by setting the buffer pointer in the first word of the buffer header. If the buffer use bit of the new buffer is 1, the job is put into a wait state until the buffer is emptied at the interrupt level. The buffer is then cleared to all zeros, the buffer byte pointer and byte count are initialized in the buffer header, and control is returned to the user's program.

(1) Example

General Output-One-Character Subroutine

```
PUTCHR:         0                       ;SIMILAR TO GETCHR BUT NO END
                                        ;OF FILE CHECKING IS REQUIRED
                SOSLE OBUF+2
                JRST PUTOK
                OUTPUT 3,
                STATZ 3, 740000
                JRST OUTERR
PUTOK:          IDPB AC, OBUF+1
                JRST @ PUTCHR
```

4.4.4 Status Checking and Setting

The file status is manipulated by the GETSTS (operation code 062), STATZ (operation code 063), STATO (operation code 061) and SETSTS (operation code 060) programmed operators. In each case the accumulator field of the instruction selects a data channel. If no device is associated with the specified data channel, Monitor stops the job and prints,

ERROR - CHAN

console leaving the console in Monitor mode.

GETSTS D, E stores the file status of data channel D in the right half and zero in the left half of location E.

STATZ D, E skips, if all file status bits selected by the effective address E are zero.

STATO D, E skips, if any file status bit selected by the effective address E is one.

SETSTS D, E waits until the device on channel D stops transmitting data and then performs an input close (see CLOSE, this chapter), if the input side of channel D is open, and replaces the current file status, except bit 23, with the effective address E. If the new data mode, indicated in the right four bits of E, is not legal for the device, the job is stopped and Monitor prints

ERROR - MODE

leaving the console in Monitor mode. If the data mode is changed by SETSTS, the byte pointers in the buffer headers are changed appropriately.

4.4.5 Terminating a File

File transmission is terminated by the CLOSE D, N (operation code 070) programmed operator. If no device is associated with channel D or if bits 34 and 35 of the instruction are both one, control returns to the user's program immediately.

If bit 34 is zero and the input side of data channel D is open, it is now closed. In data modes 15, 16, and 17, the effect is to execute a device-dependent function and clear the end-of-file flag, bit 22 of the file status. Data modes 0, 1, 10, 13, and 14 have the additional effect, if an input buffer ring exists, of setting the ring use bit (bit 0 of the first word of the buffer header) to 1, setting the buffer byte count (the third word of the buffer header) to zero and setting the buffer use bit (bit 0 of the second word of the buffer) of each buffer to zero.

If bit 35 of the instruction is 0 and the output side of channel D is open, it is now closed. In data modes 15, 16, and 17, the effect is to execute a device dependent function. In data modes 0, 1, 10, 13, and 14, if a buffer ring exists, all buffers that have not yet been transmitted to the device are now written, device-dependent functions performed, the ring use bit is set to 1, the buffer byte count is set to zero, and control returns to the user after transmission is complete.

4.4.5.1 Examples

Terminating A File

```
DROPDV:         0                       ;JSR HERE
                CLOSE 3,                ;WRITE END OF FILE AND TERMINATE
                                        ;INPUT
                STATZ 3, 740000         ;RECHECK FINAL ERROR BITS
                JRST OUTERR             ;ERROR DURING CLOSE
                RELEAS 3,               ;RELINQUISH THE USE OF THE
                                        ;DEVICE, WRITE OUT THE DIRECTORY
                MOVE 0, SV JBFF
                MOVEM 0, JOBFF          ;RECLAIM THE BUFFER SPACE
                JRST @ DROPDV           ;RETURN TO MAIN SEQUENCE
```

4.4.6 Synchronization of Buffered I/O

In some instances, such as recovery from transmission errors, it is desirable to delay until a device completes its input/output activities. The programmed operators,

CALL D, [SIXBIT/WAIT/] and CALLI D,10

return control to the user's program when the device on channel D becomes inactive. If no device is associated with data channel D, control returns immediately. After the device is stopped, the position of the device relative to the data currently being processed by the user's program can be determined by the buffer use bits.

4.4.7 Relinquishing a Device

When all transmission between the user's program and a device is finished, the program must relinquish the device by performing a

RELEASE D,.

RELEASE (operation code 071) returns control immediately, if no device is associated with data channel D. Otherwise, both input and output sides of data channel D are CLOSEd and the correspondence between channel D and the device, which was established by the INIT or OPEN programmed operators, is terminated. (CALL [SIXBIT/EXIT/] performs a RELEASE on all devices initialized by INIT.

CHAPTER 5
DEVICE DEPENDENT FUNCTIONS

This chapter explains the unique features of each standard I/O device. All devices accept the programmed operators explained in Chapter 4 unless otherwise indicated. Buffer sizes are given in octal and include two bookkeeping words. Table 5-1 is a summary of the characteristics of all devices.

Table 5-1 Device Summary

Physical Name	Name	Hardware Type Number	Prog. Op.	Data Modes	Buffer[1] Size (octal)
TTY	Teletype	626	INPUT OUTPUT	A, AL	21
PTR	Paper Tape Reader	760	INPUT	A, AL, IB, B, I	43
PTP	Paper Tape Punch	761	OUTPUT	A, AL, IB, B, I	43
LPT	Line Printer	646	OUTPUT	A, AL	34
CDR	Card Reader	461	INPUT	A, AL, B, I	36
DTA1, DTA2, ..., DTA0,	DECtape	551/555	INPUT OUTPUT LOOKUP ENTER USETO USETI UGETF CALL [SIXBIT/UTPCLR/]	A, AL, IB, B, I, DR, D	202
MTA0, MTA1, ..., MTA7	Magnetic Tape	516	INPUT OUTPUT MTAPE	A, AL, IB, B, I, DR, D	203
DSK	Disk		INPUT OUTPUT LOOKUP ENTER RENAME USETO	A, AL, I, IB, B, DR, D	203

[1] Buffer sizes are subject to change and should be calculated rather than assumed by user programs. A dummy INBUF or OUTBUF may be employed for this purpose.

5.1 TELETYPE

Device Name - TTY

Buffer Size - 21_8 words.

5.1.1 Data Modes

5.1.1.1 A(ASCII) - All characters typed in appear in the input buffer as typed with the following exceptions:

RUBOUT	Erases the previous character. Successive RUBOUTs erase characters to the left until the beginning of the current bufferful. For each character erased, the character being erased is typed. If there is no character to erase, a carriage-return/line-feed is performed.
RETURN (carriage return)	Followed automatically with line-feed, both of which appear in the input buffer.
↑U (CTRL U)	Types back as ↑U followed by a carriage-return/line-feed. This character deletes the entire current bufferful of ASCII characters.
↑O (CTRL O)	Similar to ↑U but has special action during output.
↑Z (CTRL Z)	Types as ↑Z and appears in the buffer as 032. This character serves as an end-of-file.
↑C (CTRL C)	Types as ↑C followed by carriage-return/line-feed. This character places the console in the Monitor mode, ready to accept Monitor commands.

On output, all characters are typed just as they appear in the output buffer.

The Teletype service routine is full duplex; i.e., it tests for input even while output is in progress. If input is sensed, it is placed in the output buffer and echoed back to the Teletype after the current output operation is completed. If ↑O is typed, the current output operation is stopped until the next input character is typed. If ↑C is typed, Monitor mode is entered immediately but not before the current output operation has been completed (whereupon the ↑C is echoed).

The buffer is terminated on RETURN (carriage-return), LINE FEED, FORM, VT, ↑Z, and ALTMODE (175) sometimes labeled ESC (escape). One of these characters always appears at the end of each bufferful. RETURN is always followed by LINE FEED so that RETURN is never the last character in the buffer.

5.1.1.2 AL (ASCII Line) - Same as ASCII mode (usually preferred).

5.1.2 DDT Submode

To allow a user's program and the DDT debugging program to use the same Teletype without interfering with one another, the Teletype service routine provides the DDT submode. This mode does not affect the Teletype status if it is initialized with the INIT operator. It is not necessary to use INIT in order to do I/O in the DDT submode. I/O in DDT mode is always to the user's Teletype and not to any other device.

In the DDT submode, the user's program is responsible for its own buffering. Input is usually one character at a time, but if the typist types characters faster than they are processed, the Teletype' service routine supplies a bufferful of characters at a time.

To input characters in DDT mode, use the sequence

```
MOVEI AC, BUF
CALL AC, [SIXBIT/DDTIN/]
```

BUF is the first address of a 21-word block in the user's area. The DDTIN operator delays, if necessary, until one character is typed in. Then all characters (in 7-bit packed format) typed in since the previous occurrence of DDTIN are moved to the user's area in locations BUF, BUF 1, etc. The character string is always terminated by a null character (000). RUBOUTs are not processed by the service routine but are passed on to the user. The special control characters ↑O and ↑U have no effect. Other characters are processed as in ASCII mode.

To perform output in DDT mode, use the sequence

```
MOVEI AC, BUF
CALL AC, [SIXBIT/DDTOUT/]
```

BUF is the first address of a string of packed 7-bit characters terminated by a null (000) character. The Teletype service routine delays until the previous DDTOUT operation is complete, then moves the entire character string into the Monitor, begins to output the string, and restarts the user's program. Character processing is the same as for ASCII mode output.

5.2 PAPER TAPE READER

Device Mnemonic - PTR

Buffer Size - 43_8 words

5.2.1 Data Modes (Input Only)

NOTE: To initialize the paper tape reader, the input tape must be threaded through the reading mechanism and the FEED button depressed.

5.2.1.1 A (ASCII) - Blank tape (000), RUBOUT (377), and null characters (200) are ignored. All other characters are truncated to seven bits and appear in the buffer. The physical end of the paper tape serves as an end-of-file and results in the character 032 (↑Z) appearing in the buffer.

5.2.1.2 AL (ASCII Line) - Character processing is the same as for the A mode. The buffer is terminated by LINE FEED, FORM, or VT.

5.2.1.3 I (Image) - There is no character processing. The buffer is packed with 8-bit characters exactly as read from the input tape. Physical end of tape is the end-of-file indication but does not cause a character to appear in the buffer.

5.2.1.4 IB (Image Binary) - Characters not having the eighth hole punched are ignored. Characters are truncated to six bits and packed six to the word without further processing. This mode is useful for reading binary tapes having arbitrary blocking format.

5.2.1.5 B (Binary) - Checksummed binary data is read in the following format. The right half of the first word of each physical block contains the number of data words that follow and the left contains half a folded checksum. The checksum is formed by adding the data words using 2s complement arithmetic, then splitting the sum into three 12-bit bytes and adding these using 1s complement arithmetic to form a 12-bit checksum. The data error status flag (IODERR) is raised if the checksum miscompares. Because the checksum and word count appear in the input buffer, the maximum block length is 40. The byte pointer, however, is initialized so as not to pick up the word count and checksum word.

Again, physical end of tape is the end-of-file indication but does not result in putting a character in the buffer.

5.3 PAPER TAPE PUNCH

Device Mnemonic - PTP

Buffer Size - 43_8 words

5.3.1 Data Modes

5.3.1.1 A (ASCII) - The eighth hole is punched for all characters. Tape-feed without the eighth hole (000) is inserted after form-feed. A rubout is inserted after each vertical or horizontal tab. Null characters (000) appearing in the buffer are not punched.

5.3.1.2 AL (ASCII Line) - The same as A mode. Format control must be performed by the user's program.

5.3.1.3 I (Image) - Eight-bit characters are punched exactly as they appear in the buffer with no additional processing.

5.3.1.4 IB (Image Binary) - Binary words taken from the output buffer are split into six 6-bit bytes and punched with the eighth hole punched in each line. There is no format control or checksumming performed by the I/O routine. Data punched in this mode is read back by the paper tape reader in the IB mode.

5.3.1.5 B (Binary) - Each bufferful of data is punched as one checksummed binary block as described for the paper tape reader. Several blank lines are punched after each bufferful for visual clarity.

5.3.2 Special Programmed Operator Service

The first output programmed operator of a file causes about two fanfolds of blank tape to be punched as leader. Following a CLOSE, an additional fanfold of blank tape is punched as trailer. No end-of-file character is punched automatically.

5.4 LINE PRINTER

Device Mnemonic - LPT

Buffer Size - 34_8 words

5.4.1 Data Modes

5.4.1.1 A (ASCII) - ASCII characters are transmitted to the line printer exactly as they appear in the buffer. See the PDP-10 System Reference Manual, for a list of the vertical spacing characters.

5.4.1.2 AL (ASCII Line) - This mode is exactly the same as A and is included for programming convenience. All format control must be performed by the user's program; this includes placing a RETURN, LINE-FEED sequence at the end of each line.

5.4.2 Special Programmed Operator Service

The first output programmed operator of a file and the CLOSE at the end of a file cause an extra form-feed to be printed to keep files separated.

5.5 CARD READER

Device Mnemonic - CDR

Buffer Size - 36_8 words

5.5.1 Data Modes

5.5.1.1 A (ASCII) - All 80 columns of each card are read and translated to 7-bit ASCII code. Blank columns are translated to spaces. At the end of each card a carriage-return/line-feed is appended. A card with the character 12-11-0-1 punched in column 1 is an end-of-file card. Columns 2 through 80 are ignored, and an end-of-file character, 032, appears as the last character in the input buffer. The end-of-file button on the card reader has the same effect as the end-of-file card. As many complete cards as can fit are placed in the input buffer, but cards are not split between two buffers. Using the standard-sized buffer, only one card is placed in each buffer. The left arrow character, 137, appears in each column containing an invalid punch.

5.5.1.2 AL (ASCII Line) - Exactly the same as the A mode.

5.5.1.3 I (Image) - All 12 punches in all 80 columns are packed into the buffer as 12-bit bytes. The first 12-bit byte is column 1. The last word of the buffer contains columns 79 and 80 as the left and middle bytes respectively. The end-of-file card and the end-of-file button are processed the same as in the A mode with the character 0032 appearing in the buffer as the last character of the file. Cards are not split between two buffers.

5.5.1.4 B (Binary) - Card column 1 must contain a 7-9 punch to verify that the card is in binary format. The absence of the 7-9 punch results in raising the IOIMPM (improper mode) flag in the card reader status word. Card column 2 must contain a 12-bit checksum as described for the paper tape reader binary format. Columns 3 through 80 contain binary data, 3 columns per word for 26 words. Cards are not split between two buffers. The end-of-file card and the end-of-file button are processed the same as in the A mode with a word containing 003200000000 appearing as the last word in the file.

5.6 DECTAPE

Device Mnemonic - DTA0, DTA1, ..., DTA7

Buffer Size - 202_8 words

5.6.1. Data Modes

5.6.1.1 A(ASCII) - Data is written on DECtape exactly as it appears in the buffer. No processing or checksumming of any kind is performed by the service routine. The self-checking of the DECtape system is sufficient assurance that the data is correct. See the description of DECtape format below for further information concerning blocking of information.

5.6.1.2 AL (ASCII Line) - Same as A.

5.6.1.3 I (Image) - Same as A. Data consists of 36-bit words.

5.6.1.4 IB (Image Binary) - Same as I.

5.6.1.5 B (Binary) - Same as I.

5.6.1.6 DR (Dump Records) - This mode is accepted but actually functions as dump mode 17.

5.6.1.7 D (Dump) - Data is read into or written from anywhere in the user's core area without regard to the standard buffering scheme. Control for read or write operations must be via a command list in core memory. The command list format is as described in Chapter 4, "Unbuffered (Dump) Modes;" any positive number appearing in a command list terminates the list. Dump data ia automatically blocked into standard-length DECtape blocks by the DECtape control. Unless the number of data words is an exact multiple of the standard length of a DECtape block (128_{10}), after each output programmed operator, the remainder of the last block written is wasted. The input programmed operator must specify the same number of words that the corresponding output programmed operator specified in order to skip over the wasted fractions of blocks.

5.6.2 DECtape Block Format

A standard reel of DECtape consists of 578 (1102_8) prerecorded blocks each capable of storing 128 (200_8) 36-bit words of data. Block numbers which label the blocks for addressing purposes are recorded between blocks. These block numbers run from 0 to 1101_8. Blocks 0, 1, and 2 are normally not used during time-sharing and are reserved for a bootstrap loader. Block 100_{10} (144_8) is the directory block which contains the names of all files on the tape and the information relating to each file. Blocks 1_{10} through 99_{10} ($1-143_8$) and 101_{10} through 577_{10} ($145-1101_8$) are usable for data.

If in the process of DECtape I/O, the I/O service routine is requested to use a block number larger than 1101_8 or smaller than 0, the Monitor sets an error bit and returns.

5.6.3 DECtape Directory Format

The directory block (block 100_{10}) of a DECtape contains directory information for all files on that tape; a maximum of 22 files can be stored on any one DECtape.

Words 0 through 82_{10}	The first 83 words of the directory contain "slots", each "slot" representing one of the 578 blocks on the DECtape. Each slot occupies five bits (seven slots are stored per word) and contains the number of the file ($1-26_8$) to which the block represented by the slot is assigned.	
Words 83 through 104_{10}	The next 22 words contain the filenames of the 22 files residing on the DECtape. Word 83 contains the filename for file 1, word 84 the filename for file 2, etc. Filenames are stored in 6-bit code.	
Words 105 through 126_{10}	The next 22 words contain the extension names and dates of the 22 files, in the same relative order as their filenames above.	
	Bits 0 through 17_{10}	The extension name of the file (in 6-bit code)
	Bits 18 through 23_{10}	Number of 1K blocks minus 1 needed to load the file (maximum value=63). This information is stored for SAVEd files only.
	Bits 24 through 35_{10}	The date the file was last updated, according to the formula: ((year-1964)*12+(month-1))*31+day-1
Word 127_{10}	Unused.	

The message

BAD DIRECTORY FOR DEVICE DATAX: EXEC CALLED FROM USER LOC n

is produced whenever any of the following conditions are detected.

1. A parity error while reading the directory block.

2. No "slots" are assigned to the file number of the file.

3. The tape block which may possibly be the first block of the file (i.e., the first block for the file encountered while searching backwards from the directory block) cannot be read.

5.6.4 DECtape File Format

A file consists of any number of DECtape blocks. Each block contains:

Word 0	Left half	The link. The link is the block number of the next block in the file. If the link is zero, this block is the last in the file.
	Right half	A count of the number of words in this block that are used (maximum 177_8).
Words 1 through 177_8		Data packed exactly as the user places it in his buffer.

5.6.5 Special Programmed Operators Service

Several programmed operators are provided for manipulating DECtape. These allow the user to manipulate block numbers and to handle directories.

In addition to the operators above, INPUT, OUTPUT, CLOSE, and RELEAS have special effects. When performing nondump input operations, the DECtape service routine reads the links in each block to determine the next block to read and when to raise the end-of-file flag.

When an OUTPUT is given, the DECtape service routine examines the left half of the first data word in the output buffer (the word containing the word count in the right half). If this half word contains -1, it is replaced with a 0 before being written out, and the file is thus terminated. If this half word is greater than -1, it is not changed and the service routine uses it as the block number for the next OUTPUT.

Table 5-2 DECtape Programmed Operators

Programmed Operator	Effect
USETI D, E	Sets the DECtape on device channel D to input block E next. Input operations on this DECtape must not be active because otherwise the user has no way of determining which buffer contains block E.
USETO D, E	Similar to USETI but sets the output block number. USETO waits until the device is inactive before setting up the new output block number.
UGETF D, E	Places the number of the first free block of the file in user's location E.
ENTER D, E	User's locations E, E+1, E+2, and E+3, must be reserved for a directory entry. The DECtape service routine searches the directory for a filename and extension that match the contents of E and the left half of E+1. If no match is found and there is room in the directory, the

Table 5-2 DECtape Programmed Operators (cont)

Programmed Operator	Effect
ENTER D, E (cont)	service routine places the first free block number into the right half of E+1, places the date in E+2 (unless already nonzero), and places the necessary information into the directory. If a match is found, similar actions occur, but the new entry replaces the old. If there is no room in the directory, ENTER returns to the next location. Otherwise, ENTER skips one location.
LOOKUP D, E	Similar to ENTER but sets up an input file. The contents of E and E+1 are matched against the filenames and extension names in the DECtape directory. If a match is found, information about the file is read from the directory into the appropriate portions of the 4-word block beginning at E. The first block of the file is then found as follows. 1. The first 83 words of the DECtape directory are searched in a backwards manner, beginning with the slot immediately prior to the directory block, until the first slot containing the desired file number is found. 2. The block associated with this slot is then read in and bits 18 through 27 of the first word of the block (these bits contain the block number of the first block of the file) are checked. If they are equal to the block number of this block, then this block is the first block of the file; if not, then the block with that block number is read as the first block of the file. LOOKUP then skips one location. If no match is found, LOOKUP returns to the user's program at the next location.
CALL D, [SIXBIT/UTPCLR/]	UTPCLR clears the directory of the DECtape on device channel D. A cleared directory has zeroes in the first 83 words except in those slots related to blocks 0, 1, 2, and 100_{10}. Only the directory block (block 100) is affected by UTPCLR; the other blocks are unaffected. This programmed operator does nothing if the device on channel D is not DECtape.

For both INPUT and OUTPUT, block 100 (the directory) is treated as an exception case. If the user's program gives

USETI D, 1

to read block 100, it is treated as a 1-block file.

The CLOSE operator places a -1 in the left half of the first word in the last output buffer, thus terminating the file.

The RELEAS operator writes the copy of the directory which is normally kept in core onto block 100, but only if any changes have been made. Certain console commands, such as KJOB perform an implicit RELEAS of all devices and, thus, write out a changed directory even though the user's program failed to give a RELEAS.

5.6.6 Special Status Bits

If an attempt is made to write on a unit with the WRITE-LOCK switch on, the device error flag (IODERR) is raised.

5.6.7 Important Considerations

The DECtape service routine reads the directory from a tape the first time it is required to perform a LOOKUP, ENTER, or UGETF; the directory image remains in core until a new ASSIGN command is executed from the console. To inform the DECtape service routine that a new tape has been mounted on an assigned unit, the user must use an ASSIGN command. The directory from the old tape could be transferred to the new tape, thus destroying the information on that tape unless the user reassigns the DECtape transport every time he mounts a new reel.

5.7 MAGNETIC TAPE

Magnetic tape format is IBM compatible and is not described here.

Device Mnemonic - MTA0, MTA1,...,MTA7

Buffer Size - 203_8 words

5.7.1 Data Modes

5.7.1.1 A (ASCII) - Data is written on magnetic tape exactly as it appears in the buffer. No processing or checksumming of any kind is performed by the service routine. The parity checking of the magnetic tape system is sufficient assurance that the data is correct. Normally, all data, both binary and ASCII, is written with odd parity and at 556 bits per inch. A maximum of 200 words per second is standard. The word-count is not written on the tape.

5.7.1.2 AL (ASCII Line) - Same as A.

5.7.1.3 I (Image) - Same as A but data consists of 36-bit words.

5.7.1.4 IB (Image Binary) - Same as I.

5.7.1.5 B (Binary) - Same as I.

5.7.1.6 DR (Dump Records) - Variable length records are read into or written from anywhere in the user's core area without regard to the standard buffering scheme. Control for read or write operations must be via a command list in core memory. The command list format is as described in Chapter 4, "Unbuffered (Dump) Modes." For input operations a new record is read for each word in the command list (except GOTO words); if the record terminates before the command word is satisfied, the service routine skips to the next command word. If the command word runs out before the record terminates, the remainder of the record is ignored. For each output command word, exactly one record is written.

5.7.1.7 D (Dump) - This mode is accepted but actually functions as DR mode 16.

5.7.2 Special Programmed Operator Service

CLOSE performs a special function for magnetic tape. When an output file is closed (both dump and nondump), the I/O service routine automatically writes two end-of-file marks and backspaces over one of them. If another file is now opened, the second end-of-file is wiped out leaving one end-of-file between files. At the end of the in-use portion of the tape, however, there appears a double end-of-file character which is defined as the logical end of tape. When an input dump file is closed, the I/O service routine automatically skips to the next end-of-file.

A special programmed operator called MTAPE provides for such tape manipulation functions as rewind, backspace record, backspace file, etc. The format is

MTAPE D,FUNCTION

where D is the device channel on which the magnetic tape unit is initialized. FUNCTION is selected according to the following table:

Table 5-3 MTAPE Functions

Function	Action
1	Rewind to load point
11	Rewind and unload[1]
7	Backspace record
17	Backspace file
3	Write end of file
6	Skip one record
13	Write 3 inches of blank tape
16	Skip one file
10	Space to logical end of tape

[1]On the 516 control, the rewind and unload function is not currently implemented, as such, but is treated as a rewind function only.

MTAPE waits for the magnetic tape unit to complete whatever action is in progress before performing the indicated function. Bits 18 through 25 of the status word are then cleared, the indicated function is initiated, and control is returned to the user's program immediately. It is important to remember that when performing buffered input/output, the I/O service routine can be reading several blocks ahead of the user's program. MTAPE affects only the physical position of the tape and does not change the data that has already been read into the buffers.

5.7.3 Special Status Bits

Special bits of the status word are reserved for selecting the density and parity mode of the magnetic tape. Table 5-4 lists the bits that are set and cleared by INIT or SETSTS.

Table 5-4 Magnetic Tape Special Status Bits

Bit	Name	Action
19[1]	IODERR	I/O Device ERRor. When set to one during an output operation means that the write protect ring is out.
24[1]	IOBOT	I/O Beginning of Tape. The tape is at the load point.
25[1]	IOTEND	I/O Tape END. The tape is at or past the end point.
26	IOPAR	I/O Parity. 0 for odd parity, 1 for even parity.[2]
27-28	IODENS	I/O Density. 00 or 10 =556 bpi 01 =200 bpi 11 =800 bpi
29	IORCK	I/O No Read Check. Suppress automatic error correction if bit 29 is a 1. Normal error correction is to repeat the desired operation 10 times before setting an error status bit.

[1] These bits indicate special magnetic tape conditions and are set by the magnetic tape service routine when the conditions occur.

[2] Odd parity is preferred. Even parity should be used only when creating a file to be read in BCD on another computer.

5.8 DISK

Device Mnemonic - DSK

Buffer Size - 203_8 words

5.8.1 Data Modes

5.8.1.1 A (ASCII) - Data is written on the disk exactly as it appears in the buffer.

5.8.1.2 AL (ASCII Line) - Same as A.

5.8.1.3 I (Image) - Same as A. Data consists of 36-bit words.

5.8.1.4 IB (Image Binary) - Same as I.

5.8.1.5 B (Binary) - Same as I.

5.8.1.6 DR (Dump Records) - Functions exactly the same as D.

5.8.1.7 D Dump - Data is read into or written from anywhere in the user's core area without regard to the normal buffering scheme. Control for read or write operations must be via a command list in core memory. The command list format is as described in Chapter 4, "Unbuffered (Dump) Modes." The disk control automatically measures dump data into standard-length disk blocks of 200 octal words. Unless the number of data words is an exact multiple of the standard length of a disk block (200 words) after each output programmed operator, the remainder of the last block is wasted.

5.8.2 Disk Format

The standard block length of the disk is 200 octal words. In contrast to DECtape, all 200 words are used for data.

Since more than one user may be accessing files on the disk at the same time, a separate directory is created by the disk service routine for each user. Each directory lists the files owned by a single user. These directories are called User File Directories (UFDs), and may contain a list of an indefinite number of files. Each UFD is in turn a data file with the same file format as normal data files (see below). The Monitor creates a special directory, called the Master File Directory (MFD), which lists all UFDs currently residing on the disk. Each directory entry contains two words.

First word		The filename in 6-bit code (the project-programmer number if the file is a UFD).
Second word	Left half	The filename extension in 6-bit code. The filename extension used to read UFDs is UFD.
	Right half	The location of the first block on the disk which contains information used to retrieve the file.

5.8.3 File Format

A file is composed of two parts: data, and information needed by the Monitor to retrieve the file. Each data block contains exactly 200 words. If the user does not fill a data buffer before doing an OUTPUT, the remainder of the buffer is filled with zeros; upon reading that particular block, it is assumed that there are 200 words of data.

The disk service routine creates a packet of retrieval information in the following format:

First word		The filename (used to check the hardware for a possible misread).
Second word	Left half	The filename extension (opposite left half).
	Bits 18-23	Unused.
	Bits 24-35	The date the file was last referenced, in the form used by the CALL SIXBIT/DATE/ UUO.
Third word	Bits 0-8	Protection (see below).
	Bits 9-12	The data mode in which the file was written.
	Bits 13-23	The 24-hour time (in minutes) that the file was originally created.
	Bits 24-35	The date that the file was originally created, in the same form as word 2.
Fourth word	Left half	The size of the file in blocks.
	Right half	A number identifying the user who created the file.

Following these four words is one word for each data block. The left half of each word is a 1s-complement sum of the data in the data block; the right half of each word is the logical address of the data block.

It should be noted that the Monitor addresses the device as if the device consisted of a number of blocks with consecutive record addresses. This is done only for convenience to the Monitor; the physical record address is computed from the logical address at the time that actual transmission of data from or to the device is initiated.

The user need not be concerned with either logical or physical disk addresses. He numbers blocks sequentially (1, 2, 3, ...n) as they are written on the disk, and specified a block to be retrieved by these sequence numbers.

5.8.4 Project-Programmer Numbers

Since any number of users may store information on the disk, each user must have a way to specify which UFD belongs to him. This is done with the project-programmer number, a 36-bit octal number. The right half of this word (programmer number) is unique with each user.

Since a user may be working on more than one project at a time, and may desire to have separate UFDs for each project, the left half of the project-programmer number specifies which of the user's UFDs is desired.

APPENDIX 1
GENERATING A CUSTOMIZED MONITOR

RUNNING SYSTEM BUILDER

1. Mount the BUILD tape on a DECtape transport.
2. Type INIT.
3. Type ASSIGN phy-name DTAI, where phy-name is the device upon which the BUILD tape was mounted in step 1.
4. Type RUN DTAI BUILD

The System Builder program types out questions, about the configuration, which the user answers. After these questions have been answered, System Builder loads the user's customized Monitor. The System Builder sets up the priority interrupt chains, and links the device data blocks, creating additional copies for multiple unit devices.

A complete description of System Builder is available in a separate document, PDP-10 System Builder (BUILD) (DEC-10-SBAA-D), which may be ordered from

Program Library Services
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts 01754

DETERMINING CORE REQUIREMENTS FOR CUSTOMIZED MONITORS

Core requirements for the specialized Monitor can be obtained by requesting a storage map during the dialogue with System Builder.

The size of the customized Monitor depends on the equipment configuration. This can be calculated, as shown on the following page.

Control routines in all Monitors:	2750_8 locations
If 7 PI channels, add 44_8 times the number used, plus 2 times the number of channels not used.	________
If your PDP-10 does not have Byte and Floating Point hardware, add 610_8.	________
If this Monitor handles I/O for the following devices, add the numbers given.	________
Paper tape reader, 137_8.	________
Paper tape punch, 236_8.	________
Card reader, 250_8.	________
If you have either a PTR, or PTP or CDR, add 125_8.	________
Line printer, 140_8.	________
Disk (not yet available)	________
DECtapes, $1100_8 + 15_8$ (number of DTAs)	________
Magnetic tapes, $400_8 + 11_8$ (number of MTAs)	________
Total core requirements for Monitor	________

APPENDIX 2

SUMMARY OF TELETYPE COMMANDS TO MONITOR

To Do This	Type This	And Monitor Does This
To initialize a job program	INIT	Releases logical names and directories used by previous job; resets JOBREL to first location below Monitor.
To assign a logical name to a device	ASSIGN dev nam	Assigns the logical name to the device specified, allocates directory space if needed. Example: AS DTA5 IMPUT
To release a logical name	DEASSIGN dev	Releases logical name assigned to device specified. Releases directory space, if any, and updates JOBREL.
To save a job for later execution	SAVE dev prognam	Copies user's job program area, except locations with zero contents onto the device specified, and assigns the PROGNAM.SAV. Also saves the job-data items entered by the Loader and by the user.
To load and start job	RUN dev prognam	Zeros job program area of core; loads the program named from the device specified; starts execution at the location specified by JOBSA.
To load a job	GET dev prognam	Zeros job program area; loads the job, but does not start it.
To start execution of a job	START	Starts the job at the address specified by the right half of JOBSA.
To interrupt a job	↑C	Stops the job; stores the address of the next instruction in JOBOPC.
To continue an interrupted job	CONT	Restarts the interrupted job at the instruction whose address is in JOBOPC.
To debug with DDT: a. Load the job with Linking Loader b. Load DDT with Linking Loader. c. Start debugging by typing:	DDT	Transfers control to DDT.
To restart at a specified reentry point.	REENTER	Restarts the job at the instruction pointed to by JOBREN.
To kill a job	KJOB	Releases all active devices.
To enter the date in the job-data area	DATE mm-dd-yy	Enters this date (in internal format) in the location labelled DATE in the job data area.

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively, we need user feedback: your critical evaluation of this manual and the DEC products described.

Please comment on this publication. For example, in your judgment, is it complete, accurate, well-organized, well-written, usable, etc? ______

Did you find this manual easy to use? ______

What is the most serious fault in this manual? ______

What single feature did you like best in this manual? ______

Did you find errors in this manual? Please describe. ______

Please describe your position. ______

Name ______ Organization ______

Street ______ State ______ Zip ______

.... Fold Here ....

.... Do Not Tear - Fold Here and Staple ....

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Quality Control
Building 12
146 Main Street
Maynard, Mass. 01754

digital

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

Printed in U.S.A.