ESD-TR-66-96

MTR-159

PRELIMINARY USER'S GUIDE TO MONITOR 1

DECEMBER 1966

R. Silver
L. Lamport

Prepared for

DEPUTY FOR ENGINEERING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

Project 508F

Prepared by

THE MITRE CORPORATION
Bedford, Massachusetts

Contract AF19(628)-5165

ESD-TR-66-96

MTR-159

PRELIMINARY USER'S GUIDE TO MONITOR 1

DECEMBER 1966

R. Silver
L. Lamport

Prepared for

DEPUTY FOR ENGINEERING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

Project 508F
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-5165

ABSTRACT

This report describes the commands provided for the Phoenix time-sharing supervisor named Monitor I and gives guidelines for its use. Special attention is paid to the I/O commands and the operating conventions that are interposed between the user and the machine.

REVIEW AND APPROVAL

This Technical Report has been reviewed and is approved.

CHARLES A. LAUSTRUP, Col. USAF
Director of Computers
Deputy for Engineering and Technology

TABLE OF CONTENTS

TABLE OF CONTENTS (CONCLUDED)

GLOSSARY

AC	accumulator
bcap	buffer capacity
BCH	branch instruction
bleft	buffer capacity unused
BR	B-register
bwc	buffer word count
CAR	central address register
CCAR	cycle CAR register
CCI	clear CAR bit
CK	clock
EOF	end of file mark
EOM	end of message character
idb	interrupt disabled bit
io	input-output
I/O	input-output
ir	instruction register
LSB	low speed buffer
MC	miscellaneous cpn bits register
MILZ	monitor inhibit lead zeros bit
MTC	metabit trap condition bit
NOP	no operation
PC	program counter
PCC	program counter central bit
PCH	punch
PGM	program
ptr	pointer
PUP	peripheral Unit Processor
tcu	tape control unit
TWC	typewriter word count
wwc	words-written count

SECTION I

INTRODUCTION

This paper sketches the principal features of the PHOENIX time-sharing supervisor called Monitor 1, from a user's point of view.

At the time of this writing, Monitor 1 is in the early stages of debugging. Consequently the material in this document is subject to change.

SECTION II

MONITOR 1

THE USER AT THE TYPEWRITER

When the user first sits down at the typewriter, the program to which he talks is ALTAR. He need not be aware that he is communicating to a definite program which is handled by the Monitor proper like any other user program: he may think that he is speaking to the system. It is, however, convenient to keep ALTAR and the Monitor separate.

ALTAR consists of a single file which can be run as a program for several different users at once (i. e. time-shared), having a distinct set of live registers for each user (see THE USER PROGRAM'S VIEW OF MONITOR).

A file is an ordered string of 25 bit PHOENIX words whose length is a multiple of 4K* words. Each file has a name,** which is unique for the particular user. A file generally belongs to one and only one user. However, there will be system files to which any user may have access.

The typewriter with which the user communicates with ALTAR is called the command typewriter. He may not switch typewriters in the middle of running, except possibly under direction of a system command typewriter. A system programmer can log into the system by using a special code, causing his typewriter to become a system command typewriter and having certain special capabilities not discussed in this report.

---

*K = 1024

**A name is either a string of letters and digits containing at least one letter, or a string of characters, the first of which is a left quote, the last of which is a right quote, which contains no other right quotes.

ALTAR COMMANDS

Following is a list of primitive ALTAR commands. The actual user commands will be combinations of these primitive ones.

login projectname, code - Enters the user into the system and is his initial command. "Projectname" is a name known to the system (introduced initially from a system command typewriter) and "code" is a password which must match the one stored for "projectname."

getfilespace number - Each user "owns" a certain amount of space for his files, which is the maximum number of words of files he may have. He initially starts out with no space, and this command adds "number" of 4K blocks to the amount of space he owns, if there is enough space left in the system. No effort is made to prevent knavery, but the amount of space the user has is always open for public inspection. "Number" may be any legal expression.

giveupfilespace number (n) - Gives up n 4K blocks which the user owns.

createfile name (f), number (n) - Create a file named f having n 4K blocks, if he has enough unused space.

destroyfile name (f) - Destroys file f, losing all information the file contained and freeing the space it occupied for his use.

makepgm name (f) - Makes file f the user's current program file, and sets the live registers to standard values. File f now becomes the program's "core registers" as described later.

run - Causes the running of the current program, under the control of its live registers. While the program is running, any typed input, except an EOM (end of message) character, will be considered to belong to the program. The program will terminate when one of the following conditions arises:

(1) supervisor terminate routine called,

(2) program causes out-of-bounds condition,

(3) EOM key hit on command typewriter, or

(4) program file too long to fit in core.

EOM

The end-of-message character causes the current program to stop running. The live registers are saved so that typing the run command will cause it to continue from where it stopped. However, any contents of the typewriter's input buffer preceding the EOM, and not yet read by the program, will be destroyed.

attach devicename (d), number (n), name (f) - D is the name of a device, n is a number from 3 to 15, and f is any legal symbol which is not already defined for the user (may be omitted). If the specified device is free, it is attached to the user as his device n. F is then defined to have the value n.

detach number (n) - The device attached to the user as his number n is detached, and made free for use by someone else.

connectfile file (f), arm (a) - Connects the file f to device number a.

disconnectfile arm (a) - Disconnects the file from device number a.

setreg reg (r), value (v) - Sets live register r to v.

panel - Prints out live registers.

THE USER PROGRAM'S VIEW OF MONITOR 1

The user's program has all of the features of PHOENIX operating in protected status. Since some of these features are of no value to the program, such as being able to read actual clock (CK) or interrupt register (nr), we define a pseudo-machine containing the useful features of the protected status of PHOENIX plus others provided by the Monitor.

The user's pseudo-machine consists of a file (representing the program's core memory) and a set of live registers. The live registers are:

1. PHOENIX registers AC, BR, CAR, CCAR, CCI, MC, MTC, PC, PCC, XR1, XR2, XR3, XR4, XR5;
2. The user's interrupt register: nr;
3. The user's interrupt disable bit: idb.

The nr and idb are described below.

THE INTERRUPT SYSTEM

The user program has an interrupt register (nr) which is 24 bits long. Bit i of this register is set to 1 when an unmasked interrupt is generated by the user's device number i. This process will be explained later. If, at the completion of any machine instruction or Monitor service-routine, nr $\neq$ 0 and the user's interrupt disable bit (idb) is 0, then an interrupt takes place, causing the following:

$1 \rightarrow$ idb

$0 \rightarrow a$

$MC\downarrow 3{:}5 \rightarrow a\downarrow 3{:}5$

PCC → a↓6

CCI → a↓7

PC → a↓8:23

O → b

CCAR → b↓0:6

effectiveaddress ([7]) → i

a → mem (i)

b → mem (i+1)

nr → mem (i+2)

1 → PCC

0 → CAR

0 → CCAR

0 → CCI

0 → nr

8 → PC

This procedure saves all the information normally saved by PHOENIX in an interrupt, plus nr , in the three memory words specified by the effective address of memory location 7. This allows pure procedures to be run by more than one user at the same time. The program then continues, with interrupt disabled (idb = 1), from location 8.

MONITOR COMMANDS

Monitor commands are executed by performing a BCH with the appropriate effective address. This address will depend upon the command, and will be a number between 2^{16} and 2^{12} and 2^{16} and 1 (in the upper 4K of the 64K possible addresses.) The n^{th} argument of the command is obtained as follows (n = 1, 2, . . .). All live registers except PC are reset to their states after the BCH was performed. PC is set to n locations after the

BCH. The instruction at this location is performed, an interrupt taking place immediately after. The current contents of AC is then the n^{th} argument.

End Command

The end command terminates the running of the program, transferring control to ALTAR which will type out the location of the command. The program's live registers will be saved by ALTAR so that the program may be allowed to continue running from the next location.

Program Termination

Any out-of-bounds memory reference, or attempted transfer of control to an out-of-bounds address (other than by a legal monitor command BCH) will cause the program to halt just as by the execution of the end command, except that ALTAR will type an appropriate error indication.

Enable Command

This command has a single 16-bit argument. If interrupts are enabled ([idb] = 0), then the command is a NOP. Otherwise, the live registers are restored in an inverse manner to their being saved by an interrupt, except that [ir] is unchanged; then interrupts are enabled ($0 \rightarrow$ [idb]) and XR5 is set equal to the argument given. (Note that contents of XR5 were changed by the BCH instruction used to call the enable command.)

Godorm Command

This command puts the program in dormant status pending the arrival of an interrupt. If the program is disabled when the godorm command is issued, then it will be enabled (as by an enable command) before going dormant.

THE USER'S DEVICES

The user has available to him up to 16 different devices, numbered from 0 to 15. Device numbers 0, 1, 2 have fixed significance, while device numbers 3 to 15 may be attached to devices arbitrarily, or left unattached.

Device number 0 is unattached while the user program is running.

Device number 1 is attached to the user program file arm, a file arm to which is connected the user program file.

Device number 2 is attached to the command typewriter.

Device numbers 3 to 15 each may be unattached, or attached to one of the following, if available:

Paper Tape Reader

Paper Tape Punch

Digital Incremental Recorder (Plotter)

Magnetic Tape Drive

File Arm (see below).

File Arms

A file arm is a device used to communicate with a file. The file arm is connected to a specific file, and may be thought of as a spigot between the program and the file. There may be more than one file arm attached to a single file, which permits overlapping of several operations to that file. (Initially, as long as working files are all kept on the drum, this overlapping is mythical and will result in no saving of time.)

Device Registers

Each device contains the following four registers. The exact meaning of the various bits of the registers will vary with the device, but there will be some standard ones.

dir - device interrupt register - 12 bits - The bits of this register are set by interrupts generated by the device. Whenever a bit is set by an interrupt, for which the appropriate bit of the device mask register (dmr) equals zero, the i^{th} bit of the user's nr (interrupt register) is set to 1, where i is the device number.

dmr - device mask register - 12 bits - A 1 in a position of this register prevents the corresponding bit of the dir from causing a program interrupt (setting nr).

dsr - device status register - 12 bits - The bits of this register describe the status of the device (e.g., busy, inoperative).

dpr - device property register - 12 bits - The contents of this register describe properties of the device. For a typewriter, the dpr specifies the break character set for that device.

The device registers can be read, and dir, dmr, and dpr be changed, using the doio (do input/output operation) command.

THE DOIO COMMAND: DOIO (DV, OP, WC, MA, DVA, ER)

This command is used to perform all input/output operations. It has the following arguments:

1. device (dv) - The number (between 0 and 15) of the device.
2. operation (op) - A 24-bit number specifying the operation. This number will be the same for similar operations performed on different devices (e.g. output characters to a typewriter or paper tape punch), but two different operations to different devices will not have the same operation name (e.g. read characters from typewriter, and rewind tape drive).

3. word count (wc) - A 16-bit number specifying the maximum number of words which may be transferred by the operation.
4. memory address (ma) - The starting address in memory of the operation.
5. device address (dva) - A number specifying an address within the device.
6. exception return (er) - If, either because of illegality or other demands upon the system, the i/o operation is not attempted, control will be transferred to the location indicated by this argument. [XR5] will then contain the normal return address (location of the BCH + 7) and [AC] will indicate by a code number why the operation was not done.

After execution of the doio command, control is returned to the program and the actual input/output proceeds in parallel. The progress of the i/o may be checked by reading the device status and interrupt registers, or by waiting for an interrupt. Until the operation is finished, no other doio command may be given to that device except to read or set the device's registers.

SECTION III

THE USER PROGRAM I/O

DEVICE REGISTERS

The following are the bits of the device registers which have a fairly uniform meaning for all devices.

dir - device interrupt register

bit 0 - mem free

Set after a doio command when the transfer of information is complete so far as memory is concerned. I. e., on input, all information has been read into memory; on output, the memory area involved may now be changed without affecting the operation.

bit 1 - device free

Set after a doio command when the device becomes free to accept another command. This interrupt will never occur before the mem free interrupt (bit 0).

bit 2 - inoperative

Set when an operation initiated by a doio command cannot be successfully performed because the device is found to be inoperative. Bits 0 and 1 will be set when this occurs.

bit 3 - device error

Set after a doio command when an error has occurred in the operation. The interrupt will occur when the error is detected, but before the operation is completed.

dsr - device status register

bit 0 - device busy

This bit is set to 1 if and only if the device is busy performing an operation initiated by a doio command and is not free to perform another operation.

bit 1 - device not ready

This bit is set to 1 if the device is not ready to perform an I/O operation (not because of the execution of a doio command) but should become ready within some fixed time depending upon the device, e.g., termination of typewriter output with begin message key.

dmr - device mask register

Same as dir in meaning of bits.

dpr - device property register - 12 bits

An additional register, whose meaning (if any) varies with the device. It can be set by the program.

GENERAL DOIO COMMAND OPERATIONS DOIO (DV, OP, WC, MA, DVA, ER)

Certain operations (argument 2 of the doio command) apply to a large number of devices and are described below.

doio operations

Number	Operation
0	readregs
1	setregs
2	read
3	write

Number	Operation
4	blankwrite
5	endfile
6	backspace
7	rewind
8	unload
9	nop
10	readwc
11	setlength
12	start
13	stop
14	readfileref
15	connectfile
16	disconnectfile

read, write

These are the common input and output operations. Their meanings will vary slightly with the device, and will be described fully for each device. In general, they cause the number of words specified by wc (argument 3) to be read into or written from memory beginning at location ma (argument 4). If the device specified by dv is busy or not ready, the operation will not be attempted and a busy or not ready exception return to er will occur. (The exception return will be discussed later.)

read registers

This operation reads the devices registers into memory at locations ma (argument 4) and ma + 1 in the following format:

ma	dsr	dir
ma + 1	dpr	dmr

[dir] is set equal to zero. The operation will be performed even if the device is busy or not ready.

setregisters

Or's $[ma]_{12:23}$ into dir and transfers $[ma + 1]_{0:11}$ into dpr and $[ma + 1]_{12:23}$ into dmr. The operation will be performed even if the device is not ready. The nr (interrupt register) bit for the device will be set if dmr∧dir is zero before the transfer, and nonzero after. Exception: the following list gives the codes if the exception return is taken.

Code Number	Meaning
0	Operation not performed for miscellaneous reasons (e.g. attempt to write or set length of read-only file).
1	Device number error. Device number is either not in the range 0 to 15, or is unattached.
2	Device busy or not ready. The operation requires the device to be idle or ready, but it is not.
3	Memory bounds violation. The operation, if performed, would apparently result in data transfer to or from an out-of-bounds location, i.e., ma not in the range of 0 to 2↑16-1, or ma + wc > length of program.
4	Illegal operation. The operation requested is not legal for the specified device. (0 ≤ op ≤ maxop test for all ops).
5	Bad device address. The device address specified is negative, or would cause trouble. E. g., for file read or write, dva + wc > length of file.
6	Bad word count. wc not in the range 0 to 2^{18} (all ops), or wc cannot be allowed for the operation, for some reason other than a memory bounds violation or a bad device address. E. g., for readregs, setregs, wc < 2.

Code Number	Meaning
7	Bad device condition (not covered in 2).
8	Pup busy.
9	Stack full.

SECTION IV
TYPEWRITERS

Typewriter input and output (ICS characters) is buffered inside the Monitor. The typewriters are normally in input mode, ready for type-in. The typewriter keyboard will be locked if:

(a) the typewriter is in output mode (typing out);

(b) the buffer is full -- usually because the program to which the typewriter is attached has been ignoring it;

(c) the LSB buffer has filled.

Since the typewriters are in single-character mode and typewriter processing is done rapidly as the highest priority task of the Monitor, (c) should not occur.

The user can always break through a hangup to get connected to ALTAR, since hitting the begin-message key will unlock the keyboard, leaving the typewriter in input mode with a clean buffer and unable to type out for a short time, and hitting the end-message key will connect the typewriter to ALTAR.

REGISTERS

The meanings of the bit positions of the typewriter registers are indicated below. These meanings will become clear when the operations are explained.

dir, dmr

bit 0 - mem free

Unused.

bit 1 - device free

Typeout complete or not ready period finished.

bit 2 - device inoperative

Typewriter discovered to be inoperative during typeout.

bit 3 - device error

Transmission error occurred during typeout.

bit 4 - terminated

Begin-Message key struck during typeout.

bit 5

Unused.

bit 6 - attention

Break characters typed in.

bit 7 - bufferbell

Input buffer near full, or output buffer near empty.

bit 8 - inputerror

Transmission error occurred during typein.

bit 9 - buffercleared

Buffer cleared in response to begin-message key being struck with full input buffer, or for output.

bit 10

Unused.

bit 11 - eomcond

eom character typed in.

dsr

bit 0 - devicebusy

Typing out in progress (≡ output mode).

bit 1 - devicenotready

Set for time tdelay after the typist has manually terminated output.

bit 2 - bufferfull

Set while the input buffer is full (more than nmarg characters).

bit 3 - bufferlow

Set while the input (output) buffer has fewer than inbell (outbell) characters in it.

dpr - specifies break character set

Consider the following character classes:

num - digits 0 through 9

lc - lower case letters a through z

uc - upper case letters A through Z

pnc - punctuation: . , : ; ?

op - other printing characters: < > ↑ ↓ / \ → ← + - *
= ( ) [ ] ` ' $ | _

cr - ↵

os - other spacing characters: □ ⊣ ⊢ ⊤

on - other nonprinting characters: ∅ ℘ {error} {illegal}

dpr bit	class
0	non-num
1	non-lc
2	non-uc
3	non-op
4	non-pnc
5	non-os
6	non-on

A character is a break character if and only if it belongs to all classes for which the corresponding dpr-bit is a one.

dpr bits set to 1	break class
none	all characters
0	nonnumeric
0, 1	nonalphanumeric and upper case
0, 1, 2	nonalphanumeric
0, 1, 2, 3	nonprinting and punctuation
0, 1, 2, 3, 4	nonprinting
0, 1, 2, 3, 4, 5	DP (error) (illegal)
0, 1, 2, 3, 4, 5, 6	

TYPEWRITER I/O

The typewriter doio operations are readregs, setregs, read, write. The monitor maintains for each typewriter a buffer having the capacity to hold bcap characters. At any moment, the typewriter is in one of two modes: input or output.

Input Mode

The typewriter keyboard is normally unlocked; it will be locked if the buffer is full.

Each character typed is entered in the buffer. It will be replaced by a special error character if a transmission error occurred. In this case, the typewriter's inputerror interrupt will be set.

If the character is a break character, as defined by dpr, then the typewriter's attention interrupt is set.

If the character is an EOM character, then the eomcond interrupt will be set both for the typewriter and for its job's program file arm. In addition, the buffer is reset, and the program terminated, connecting the typewriter to ALTAR.

If the character is the inbell-th in the buffer, then bufferbell interrupt is set, and bufferlow status is cleared.

If the character is the nmargin-th in the buffer, then set bufferball status and lock the keyboard. Characters typed in while the keyboard is being locked may be lost.

If the begin-message key is struck while the typewriter is in input mode and the buffer is full, then the following events take place:

(1) clear input buffer;

(2) unlock keyboard;

(3) clear device bufferfull status bit; and

(4) set device buffercleared interrupt bit.

Output Mode

The typewriter keyboard is locked. Characters are typed out from the buffer. When the outbell-th character is typed out, bufferbell interrupt bit is set.

If the buffer becomes empty, and all characters have actually been typed out, then:

(1) set device devicefree interrupt bit;

(2) clear device devicebusy status bit; and

(3) set typewriter to input mode, with an empty input buffer, and an unlocked keyboard.

If the begin-message key is struck while the typewriter is in output mode, then:

(1) clear output buffer and terminate typeout;
(2) set device terminated interrupt bit;
(3) set device devicenotready status, leaving it set for tdelay seconds;
(4) clear device devicebusy status bit; and
(5) set typewriter to input mode, with an empty input buffer, and an unlocked keyboard.

Bufferlow status bit is on when output buffer has no more than outbell characters in it.

Transmission Errors

If a transmission error occurs, device interrupt bit error or input-error will be set, according as the typewriter is in output mode or input mode. This depends on which mode the typewriter is in, output or input.

Inoperative

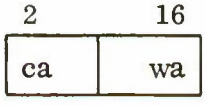
If the typewriter proves inoperative during typeout, then:

(1) clear output buffer and terminate output;
(2) set deviceinoperative interrupt bit;
(3) clear devicebusy status bit; and
(4) set typewriter to input mode, with an empty input buffer, and an unlocked keyboard.

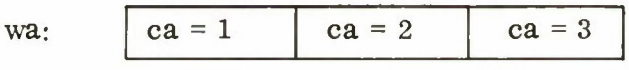
TYPEWRITER CHARACTER TRANSMISSION OPERATIONS

The doio operations getchars, putchars are character-oriented routines used to transmit strings of characters to and from the Monitor typewriter buffers.

These commands operate on a character basis rather than a word basis; as opposed to word location -- a 16-bit quantity specifying the location in memory of a 24-bit word, the commands require the concept of a character-location -- an 18-bit quantity specifying the location in memory of an 8-bit character. The commands require the concept of a character location rather than that of a word location. A word location is a 16-bit quantity specifying the location in memory of a 24-bit word; a character is an 18-bit quantity specifying the location in memory of an 8-bit character. A character-location has the following format:



where $1 \leq ca \leq 3$. This character-location specifies character number ca of the word at location wa:



getchars command - doio (d, read, p_0, p_1, ex) = p

Normal Case: d is the device number of a typewriter. Characters from the typewriter buffer are transmitted to character-locations p_0, ... , until either:

(1) A break character is transmitted; or

(2) All characters in the buffer are transmitted; or

(3) A character has been transmitted to the character-location just before p_1.

Getchars then returns with value the character-location one beyond that of the last character transmitted. The characters transmitted are deleted from the Monitor buffer.

For abnormal cases the location of the normal return is put in XR5, an exception code is put in the AC, and control is transferred to location ex.

Code Number	Meaning
0	Typewriter buffer is empty
1	Device number d is either not in the range 0 to 15, or it is unattached
2	Typewriter is busy
3	Memory bounds violation. Locations $p_0, \ldots, p_{l-1}$ are not all in bounds
4	Specified device is not a typewriter
5	Character address of p_0 or p_1 is not 1, 2, or 3.

putchars command - doio (d, write, p_0 ,, p_1 , ex) = p

Normal Case: d is the device number of a typewriter. Characters in character-locations $p_0, \ldots$ are transmitted to the typewriter buffer until either:

(1) The character in location p_{l-1} has been transmitted; or

(2) The Monitor buffer fills.

Putchars then returns with value the character-location one beyond that of the last character transmitted.

Abnormal Cases: In case of an exception, the location of the normal return is put in XR5, an exception code is put in the AC, and control is transferred to location ex.

Code Number	Meaning
0	Typewriter buffer is full
1	Device number d is either not in the range 0 to 15, or it is unattached
2	Typewriter is not ready
3	Memory bounds violation. Locations $p_0, \ldots, p_{l-1}$ are not all in bounds
4	Specified device is not a typewriter
5	Character address of p_0 or p_1 is not 1, 2, or 3.

CONSTANTS

bcap	156 characters
inbell	130 characters
nmarg	153 characters
outbell	50 characters
tdelay	0.5 second

SECTION V
FILES AND FILE ARMS

In general, the user has several files, i.e., ordered collection of data, available for his use. A program operates on a file by having a device called a file arm attached to itself, and connected to the file.

The user attaches a file arm to his program as a device numbered 3 to 15 using an appropriate ALTAR command.

A file arm may be connected to a file by the user, through typing an ALTAR command or by the program itself.

The user program may only read, write or otherwise interrogate file through file arms which have been so attached, and can refer to them only by device number. The user may have files which are not connected to file arms. Any file may be run as the program file by being connected to device number 1: the program file arm. A file may be attached to the job through more than one file arm. Each file arm will have a separate independent set of device registers. The file may be operated on through one file arm, even if it is busy under another. Each file arm to which the file is attached thus acts as a separate data channel to the file, allowing parallel operation. In Monitor 1.0, this parallel operation is merely an illusion and no overlapping of the operations will actually be done.

When active files are kept on a disk, in future systems, overlapping may be done.

REGISTERS

dir, dmr - device interrupt and mask registers

bit 0 - memoryfree

Set by a read or write operation when the data transfer is completed.

bit 1 - devicefree

Set at the same time as the memoryfree interrupt bit.

bit 2 - 11

Unused.

dsr - device status register

bit 0 - devicebusy

Equals 1 when a read or write operation is in progress.

dpr - device property register

Unused.

FILE ARM I/O OPERATIONS

User programs use file arms to manipulate files using the doio routine. The format of the doio call is:

doio(dv, op, wc, ma, dva, er)

where

dv is one of the job's device numbers, to which a file arm is attached.

op is one of the operations readregs, setregs, readfileref, read, write, setlength, connectfile, or disconnectfile.

wc is the word count.

ma is the memory address.

dva is the starting address within the file (first word of file is at address zero).

er is the exception return location.

Recall that for each job, devices numbered 0, 1, and 2 have special significance:

Device 0 is unattached.

Device 1 is attached to a file arm to which the user program file is connected.

Device 2 is attached to the command typewriter.

readregs

Normal Case: The two device registers associated with the given file arm are loaded into memory, locations ma and ma + 1, and the device interrupt register is cleared:

ma

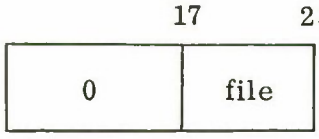
dsr	dir
dpr	dmr

setregs

Normal Case: Or [ma] ↓ 12:23 into dir; load dvr2 from ma + 1; if dmr and dir were zero before the change, and become nonzero after it, then set pir ↓ dv to one, and interrupt.

readfileref

Normal Case: The file number of the file connected to the file arm is stored at location ma in the format:



Exceptions:

4 The device attached to dv is not a file arm.

7 No file connected to the file arm.

read

Normal Case: I/O operations are initiated to transfer wc words from the file to the program. Starting location in the file is dva and in the program is ma.

write

Normal Case: I/O operations are initiated to transfer wc words from the program to the file. Starting location in the program is ma and in the file is dva.

setlength

Normal Case: The number of 4K blocks of the file attached to the specified file arm is set to one more than the greatest multiple of 4K not greater than dva.

This operation generates no interrupts and is executed immediately, even if the device is busy.

connect file

Normal Case: The file whose reference number is dva is connected to the file arm attached to dv.

disconnectfile

Normal Case: The file (if any) connected to the file arm attached to dv is disconnected from it.

EXCEPTION CODES FOR FILE ARM DOIO

doio(dn, op, wc, ma, dva, er), where op is one of readregs, setregs, readfileref, read, write, setlength, connectfile, or disconnectfile.

Code Number	Meaning
0	File is read-only (write, setlength). File not available to job (connectfile). Cannot disconnect program file (disconnectfile).
1	Device number dn is either not in the range 0 to 15, or it is unattached.
2	File arm is busy (read, write, disconnectfile).
3	Memory bounds violation: ma not in the range 0 to 64K-1 or ma + wc > length of program.
4	Illegal operation for file arm.
5	Bad device address: dva < 0, or dva + wc > length of file (read, write), or not enough unused file space (setlength), or dva not a legal file number (connectfile).
6	Bad word count: wc not in the range 0 to 2^{18}, or wc < 2 (readregs, setregs).
7	Bad device condition: file arm not connected to a file (read, write, setlength), or file arm already connected to a file (connectfile).
8	Some tape-drive or file arm attached to this job is busy with data transmission (read, write).
9	Cannot stack (read, write).

SECTION VI
TAPE-DRIVE

The doio operations for a tape-drive are readregs, setregs, read, write, backspace, rewind, unload, endfile, blankwrite, readwc, and nop. Read, write, and blankwrite are data operations. Readregs, setregs, backspace, rewind, unload, endfile, readwc, and nop are nondata operations.

dir, dmr

bit 0 - memory free

Set by data operations (ops) when information transfer is complete as far as central memory is concerned.

bit 1 - device free

Set when tape-drive is free to accept another command. This will never occur before memory free is set (read, write, blankwrite).

bit 2 - inoperative

Set when a tape operation cannot be successfully performed because the drive is found to be inoperative.

bit 3 - rwerr

Read/write error has occurred during tape operation. Set when error is detected, but before operation is completed (read, write, blankwrite, or endfile).

bit 4 - exception

Unusual condition has arisen during tape operation.

read	-	End-of-file (EOF) passed.
write	-	End-of-tape condition during operation.
blankwrite	-	End-of-tape condition during operation.
endfile	-	End-of-tape condition during operation.

bit 5 - abort

Tape Control Unit (TCU) operation was aborted: TCU busy too long.

bit 6 - rewind complete

Tape reel has reached load point.

dsr

bit 0 - device busy

Tape drive is busy.

bit 1 - ready read

Tape drive not busy and ready for reading.

bit 2 - ready write

Tape drive not busy and ready for writing.

bit 3 - load point

Tape drive at load point.

Status information about tape as of last device free or rewind complete interrupt in doio.

readcount

Register containing the number of words read by a read operation.

TAPE DRIVE OPERATIONS

doio (dv, op, wc, ma, den, er). Den specifies density: 0 = 556 bpi; 2 = 800 bpi; other = illegal.

read

1. If any exceptions occur, the operation is not attempted (return to er).
2. Set device busy status. Set up status bits 1, 2, 3.
3. Stack the operation and return. At some later time, the operation will actually be attempted.
4. If the drive proves inoperative or "busy" then set inoperative and return free interrupts, set readcount to zero, and go to step 6.
5. Read words from the tape at specified density and transmit, with zero metabits, to central memory locations ma onward, until data transmission is finished.
6. Get readcount to show the actual number of words transmitted and set memory free interrupt.
7. When the tape drive becomes idle, set the following interrupts and status bits:

   device free error - if a redundancy error occurred.

   exception - if EOF condition occurred.

   device busy - set to 0.

   dsr lists 1, 2, 3 - set to one or zero if the tape drive is ready for reading or not, ready for writing or not, at load point or not, respectively.

write, blankwrite

(1) If any exceptions occur, the operation is not attempted (return to er).

(2) Set device busy status. Set up status bits 1, 2, 3.

(3) Stack the operation and return. At some later time, the operation will actually be attempted.

(4) If the drive is inoperative or busy then set inoperative and memory free interrupts and go to step 6.

(5) If the op is blankwrite then write a blank space on tape. Obtain wc words from central memory locations ma onward, and write (ignoring metabits) on tape at the specified density.

(6) Set memory free interrupt.

(7) When the tape drive becomes idle, set the following interrupt and status bits.

device free error	-	if a redundancy error occurred.
exception	-	if end-of-tape condition occurred.
device busy	-	set to 0.
status 1, 2, 3	-	set appropriately.

backspace, endfile, nop

(1) If any exceptions occur, the operation is not attempted (return to er).

(2) Set device busy status. Set up status bits 1, 2, 3.

(3) Stack the operation and return. At some later time, the operation will actually be attempted.

(4) If the drive is inoperative or busy then set inoperative interrupt and go to step 6.

(5) Perform the specified action:

backspace - move the tape backward until either one record is passed or the load point is encountered.

endfile - write EOF on the tape in the specified density.

nop - no action.

(6) When the device becomes idle, set the following interrupt and status bits:

device free error - if a redundancy error occurred (endfile).

exception - if an end-of-tape condition occurred (endfile).

device busy - set to zero.

status 1, 2, 3 - set appropriately

rewind, unload

(1) If any exceptions occur, the operation is not attempted (return to er).

(2) Set device busy status. Set up status bits 1, 2, 3.

(3) Stack the operation and return. At some later time, the operation will actually be attempted.

(4) If the drive proves inoperative or busy then set inoperative interrupt and go to step 6.

(5) Move the tape backward to load point. If the operation is unload, then unload it.

(6) When the device becomes idle, set the following interrupt and status bits:

device free

rewind complete

device busy - set to 0.

status 1, 2, 3 - set appropriately.

readwc

(1) If any exceptions occur, the operation is not attempted (return to er).

(2) Return with value readcount.

EXCEPTIONS

Code Numbers

0	Some tape drive attached to job is busy with a non-data op. *
1	Device number error: device number dv is either not in the range 0 to 15 or is unattached.
2	Device busy or not ready for the specified operation. *
3	Memory bounds violation: ma not in the range 0 to 64K-1, or ma + wc > length or program.
4	Illegal operation: op not legal for tape drive.
5	Illegal density, den not 0 or 2.
6	Bad word count: either wc not in the range 0 to 2^{18} or wc < 2 for readregs or setregs.

* dsr readyread, readywrite, and loadpoint already updated.

Code Number	
7	Does not apply.
8	Some tape drive or file arm attached to this job is busy with data transmission. *
9	Cannot stack the operation. *

* dsr readyread, readywrite, and loadpoint already updated.

SECTION VII

PAPER TAPE READER (RDR)

The doio operations for the reader are readregs, setregs, start, read, stop.

The monitor maintains a buffer inside the Monitor for the RDR, as well as an LSB buffer.

Start specifies the size of the LSB buffer (gulpsize) and begins input from RDR.

Read transmits characters from the Monitor buffer to the program.

Stop arranges to terminate input from RDR.

REGISTERS

dir, dmr

0

1 device free - RDR input activity terminated.

2 inoperative - RDR discovered to be inoperative or faulty (not transmission error).

3 error - transmission error occurred during input.

4 does not apply.

5 does not apply.

6 does not apply.

7 attention - buffer contains at least one gulp.

8 does not apply.

9 does not apply.

10 does not apply.

11 does not apply.

dsr

0 device busy - RDR actively inputting.

1 does not apply

2 buffer full.

3 buffer empty.

4 stop read - terminate RDR activity as soon as possible.

5 does not apply.

6 does not apply.

7 does not apply.

8 does not apply.

9 does not apply.

10 does not apply.

11 milz.

dpr

Unused.

RDR I/O

device busy

This condition obtains while the RDR is actively inputting. The input operation terminates either by filling the LSB RDR buffer, or as a result of a stop doio turning off the RDR Input Status indicator.

0→ dis↓ rdn

(1) Unload LSB RDR input buffer. If RDR proves inoperative, then set the inoperative and device free interrupts, clear device busy status, and omit the remaining steps.

(2) Transfer the unloaded data (if any) to the RDR Monitor buffer.

(3) If any data was transmitted, then

a. If buffer empty status is on, turn it off and set attention interrupt.

b. If there is too little room in the Monitor buffer to accommodate another gulp, set bufferfull status.

(4) If either bufferfull status or stop read status is on, then set device free interrupt and clear device busy status. Otherwise re-initiate input activity, inhibiting lead zeros if milz = 1. Clear milz. If RDR now proves inoperative, then set inoperative and device free interrupts, and clear device busy status, otherwise 1 → dis ↓ rdr.

start

doio(ptr, start, gulpsize, ilz, , er)

(1) If device busy status on, then exception 2.

(2) If gulpsize > maxgulpsize then exception 6.

(3) If gulpsize ≤ 0 then let gulp = standard gulpsize, otherwise let gulp = gulpsize. Set the LSB input buffer for RDR to have length gulp words.

(4) Clear the Monitor buffer for RDR.

(5) Clear bufferfull and stopread status. Set bufferempty status.

(6) Arrange to initiate RDR input (after which milz will be cleared).

(7) Return with value = gulp.

read

doio(ptr, read, wc, ma, , er)

(1) If ma + wc is out of bounds then exception 3 results.

(2) Let bwc be the number of words in the Monitor RDR buffer,

and let twc = min(bwc, wc). Transmit the first twc words from the buffer to ma, . . . , ma = twc-1.

(3) Delete the first twc words from the buffer. Let bwc1 be the number of words now remaining in the buffer: bwc1 = bwc - twc.

(4) If bufferfull status is on, and there is room in the buffer for a gulp (i. e. , bwc1 + gulp + 1 ≤ buffer size) then

a. Clear bufferfull status.

b. If stop read status is off, then arrange to initiate RDR activity.

(5) If bwc1 = 0, then set bufferempty status on.

(6) Return with value twc, the number of words transmitted.

stop

doio(ptr, stop, , , , er)

1. Set stopread status.
2. If device busy status is on, then arrange to terminate RDR activity.

NOMINAL VALUES OF CONSTANTS

maxgulpsize - the largest admissible size for LSB RDR buffer (192 words).

standard gulpsize - the standard value for LSB RDR buffer (192 words).

buffer size - the size of the Monitor RDR buffer (400 words).

EXCEPTIONS FOR READER

doio(dv, op, wc, ma, dva, er)

op = read, start, stop, readregs, setregs

1. Device number error. Either dv not in the range 0 to 15 or dv is unattached.
2. Device busy (start).
3. Memory bounds violation. ma not in the range 0 to 64K-1 or ma + wc > length of program.
4. Illegal operation. op not in the range 0 to maxop, or op not legal for RDR.
5. Bad device address. dva < 0.
6. Bad word count. wc not in the range 0 to 2^{18}, or wc < 2 (readregs, setregs), or wc > maxgulpsize (start).

SECTION VIII
PAPER TAPE PUNCH

The doio operations for the punch are readregs, setregs, and write.

The Monitor maintains a buffer of bcap words for the punch, from which characters are punched. The buffer is loaded using a write command, and is unloaded automatically as long as the buffer is not empty.

DEVICE REGISTERS

dir, dmr

1	device free	-	buffer empty and punch idle.
2	inoperative	-	punch discovered to be inoperative during punch out.
3	error	-	transmission error punching characters.
7	bufferbell	-	bufferlow status has been set during punch out.

dsr

0	device busy	-	punching out in progress.
1	bufferlow	-	buffer contains no more than bell words.

TRANSMISSION ERRORS

If a transmission error occurs while the punch is busy, then the error interrupt bit is set.

INOPERATIVE

If the punch proves inoperative during punch out, then the buffer is cleared, the inoperative interrupt is set, and the status bits are reset.

WRITE OPERATIONS

The write operation is doio(pch, write, wc, ma, , er)

(1) Let there be bleft words in the Monitor punch buffer. Let wwc = min(wc, bleft). Transmit wwc words to the punch buffer from ma, . . . , max + wwc-1.

(2) If the buffer is not empty, then set device busy status.

(3) If the buffer contains no fewer than bell words, clear bufferlow status.

(4) If the buffer is not empty, arrange to punch out its contents.

(5) Return with value wwc, the number of words transmittted.

CONSTANTS

bcap - 100 words

bell - 50 words

EXCEPTION CODES

doio(dev, op, wc, ma, dva, er)

op = write, readregs, setregs.

1 Device number error. Either dv not in the range 0 to 15 or dv is unattached.

3 Memory bounds violation. ma not in the range 0 to 64K-1 or ma + wc > length of program.

4 Illegal operation. op not in the range 0 to maxop, or op not legal for PCH.

5 Bad device address. dva < 0.

6 Bad word count. wc not in the range 0 to 2^{18}, or wc < 2 (readregs, setregs).

SECTION IX
PLOTTER

The doio operations for the plotter are readregs, setregs, and write.

The Monitor maintains a buffer of bcap words for the plotter, from which data is sent to the lob to be plotted. The buffer is loaded using a write command, and is unloaded automatically as long as it is not empty.

DEVICE REGISTERS

dir, dmr

1	device free	- buffer empty and plotter idle.
2	inoperative	- plotter discovered to be inoperative during output.
3	error	- transmission error plotting points.
7	bufferbell	- bufferlow status has been set during plotter output.

dsr

0	device busy	- plotting in progress.
1	bufferlow	- buffer contains no more than bell words.

TRANSMISSION ERRORS

If a transmission error occurs while the plotter is busy, then the error interrupt bit is set.

INOPERATIVE

If the plotter proves inoperative during output, then the buffer is cleared, the inoperative interrupt is set, and the status bits are reset.

WRITE OPERATION

doio(plotter, write, wc, ma, , er)

1 Let there be bleft words in the plotter buffer. Let wwc = min(wc, bleft). Transmit wwc words to the plotter buffer from ma, . . . , ma + wwc-1.

2 If the buffer is not empty, then set devicebusy status.

3 If the buffer contains no fewer than bell words, then clear bufferlow status.

4 If the buffer is not empty, arrange to plot its contents.

5 Return with value wwc, the number of words transmitted.

CONSTANTS

bcap - 300 words

bell - 150 words

EXCEPTION CODES

doio(dev, op, wc, ma, dva, er)

op = write, readregs, setregs.

1 Device number error. Either dv not in the range 0 to 15 or dv is unattached.

3 Memory bounds violation. ma not in the range 0 to 64K-1 or ma + wc > length of program.

4 Illegal operation. op not in the range 0 to maxop, or op not legal for PCH.

5 Bad device address. dva < 0.

6 Bad word count. wc not in the range 0 to 2^{18}, or wc < 2 (readregs, setregs).

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*
The MITRE Corporation
Bedford, Massachusetts

2a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

2b. GROUP

3. REPORT TITLE
PRELIMINARY USER'S GUIDE TO MONITOR I

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
n/a

5. AUTHOR(S) *(Last name, first name, initial)*
R. Silver
L. Lamport

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December 1966	54	

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
AF 19(628)-5165	
b. PROJECT NO. 508F	ESD-TR-66-96
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* MTR-159
d.	

10. AVAILABILITY/LIMITATION NOTICES
Distribution is unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
n/a	Engineering and Technology Division, Electronic Systems Division, L. G. Hanscom Field, Bedford, Mass.

13. ABSTRACT

This report describes the commands provided for the Phoenix time-sharing supervisor named Monitor I and gives guidelines for its use. Special attention is paid to the I/O commands and the operating conventions that are interposed between the user and the machine.

DD FORM 1 JAN 64 1473

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
COMPUTER						
TIME SHARING						
OPERATOR						
SUPERVISOR						
MONITOR						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all casea ahould be unclassified. If a meaningful title cannot be aelected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS), (S), (C), or (U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional