MASSACHUSETTS INSTITUTE OF TECHNOLOGY
A.I. LABORATORY

February 1972

Artificial Intelligence
Memo No. 240A (Revised 240, December 1971)

11SIM Reference Manual

Donald E. Eastlake

ABSTRACT

A program that simulates a Digital Equipment Corporation PDP-11 computer and many of its peripherals on the A.I. Laboratory Time Sharing System (ITS) is described from a user's reference point of view. This simulator has a built in DDT-like command level which provides the user with the normal range of DDT facilities but also with several special debugging features built into the simulator. The DDT command language was implemented by Richard M. Stallman while the simulator was written by the author of this memo.

Work reported herein was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under Contract Number N00014-70-A-0362-0002.

Table of Contents

I. Introduction 1

II. Processor Simulation 2

III. Command Level and Interrupt Commands 4

IV. Expressions in 11SIM DDT 7

V. Examining and Depositing 10

VI. Zeroing, Loading and Dumping 13

VII. Execution 16

VIII. Configuration Control 18

IX. Debugging Aids 19

X. Miscellaneous 22

XI. Index to Commands 24

References 26

Appendix of Device Registers 27
A. NGDIS, The Knight Display 29
B. PK, The KW11-P Programmable Clock 31
C. DC, DCR, DCT, Asyncronous Line Interface 32
D. DM, DMR, DMT, 16-line Single Speed Multiplexer 34
E. EAE, Extended Arithmetic Unit KE11-A 36
F. ET, DECtape 38
G. DF, The RF11/RS11 Disk 40
H. LP, Line Printer 42
I. LK, The KW11-L Line-Frequency Clock 42
J. PR, High Speed Paper Tape Reader 43
K. PP, High Speed Paper Tape Punch 43
L. TK, Teletype Keyboard 44
M. TP, Teletype Printer 44
N. Console Data Switches Register 45
O. Processor General Registers 45
P. I45, PDP-11/45 Features 46
Q. Processor Status Register 46

11SIM Reference Manual

1. Introduction

11SIM is a program that simulates a Digital Equipment Corporation PDP-11 computer [Ref 1]. This simulator is designed to run on the Artificial Intelligence Laboratory PDP-10 [Ref 3] system under the ITS monitor [Ref 4]. A DDT-like [Ref 5] command facility is included which makes available functions equivalent to those provided by the PDP-11 console, the normal range of DDT facilities, and the use of several special debugging features built into the simulator.

11SIM is available under ITS as a system program of name PDP11.

The PDP-11 processor simulator was written by the author of this memo. The DDT command language was designed and implemented by Richard M. Stallman.

II. Processor Simulation

The PDP-11/20 processor is simulated by 11SIM under ITS [Ref 4] with an increase in execution time of about a factor of 30. The actual simulated versus real speed ratio varied from 25. to 37. for the PDP-11/20 processor maintenance programs. Things that would tend to increase this ratio (i.e. slow down simulation) include activation of "direct to memory" input-output devices that operate in parallel with the processor on a real PDP-11 or frequent program reference to such things as the program counter as a register other than as an addressing index (i.e., BIC %7,-(3)), non-existant memory locations, or the other special register locations.

11SIM keeps track of simulated time to one tenth of a microsecond based on the nominal timing of the PDP-11/20. Delays caused by the EAE (Extended Arithmetic Unit KE11-A) are not simulated. For information on the timing treatment of other peripherals see the Appendix.

As a convience, some PDP-11/45 [Ref 2] features are also simulated but can be disabled by detaching the "I45" device [Sec VIII]. With these PDP-11/45 features off, the only known imperfection in the PDP-11/20 processor simulation is that instructions that specify a register as a source and then modify the register in the calculation of the destination address will fetch the unmodified register as data. Thus MOV %3,(3)+ does not store the incremented register 3 as on a

real PDP-11. This is because it is convenient for the simulator to fetch the source data before calculating the destination address. Digital Equipment Corporation has said it will not guarentee the compatability of this type of instruction on future models of the PDP-11 |Ref 1|.

III. Command Level and Interrupt Commands

11SIM has a DDT-like command level [Ref 5] which the user is initially typing at. Whenever command level is entered, after the first time, it prints out the location and symbolic representation of the next instruction to be executed if the simulator were proceeded. (This location is not "open" however [Sec V].) This is usually preceded by an indication of why command level was entered. For example QUIT, BREAK, and COUNT indicate command level was entered due to a [G, a breakpoint [Sec IX], or a proceed count exhaustion [Sec VII].

When first started, 11SIM prints out its version number (this should be included in the report of any bug) and initializes the PDP-11 configuration [Sec VIII]. If the user has no disk file named .PDP11 (INIT), zeroed PDP-11 core is attached to 8K from zero as with an attach command, causing a print out. Command level is then entered with zero as the location of the next instruction for the simulator to execute. The resulting initial print out is as follows:

```
11SIM.nnn
CORE = 8.K
```

If the user does have a disk file named .PDP11 (INIT), no core is initially attached by the simulator. Instead, "INIT" is typed out and after command level is first entered, the user's file will be executed as with a :XFILE [Sec X]. The user's INIT file should attach the desired amount of core.

Command level is entered if the user quits (see [G

below this section) or the simulator halts due to a HALT instruction, a buss error during a trap, or some reason related to the debugging features of the simulator. Command level types out an exclamation point as a prompt character at the beginning of each new line of input to it. Single character rub out is allowed between activation characters. Simulated time does not pass in command mode.

The following characters have effect at the interrupt level regardless of whether the simulator or command level is active. They are invisible to simulated programs and ignored by command level (but see |Q). Some help in getting back to command level rapidly while most control the destination of normal "console" or simulated teletype output.

A. |B (Begin) Turns on output to line printer (but see :WALLP |Sec X|). If not available, output will be buffered on the disk for later printing by ITS.

B. |E (End) Turns off output to line printer.

C. |G (bell) This is the "quit" character.

C.1 If typed while at command level it causes an immediate error and return to the main command loop.

C.2 If typed while simulating, it sets up a transfer to command level to occur at the end of the current instruction or simulated input-output cycle. It is possible that due to a bug in the simulator the current instruction or I-O cycle will not end soon. (This is very much more likely for a complex newly introduced I-O device than for a basic PDP-11 instruction.) In this case more |G's may be typed with the following effects:

C.2.a |G|G This has the same effect as |G|S and will allow the user to immediately see if he has quit or not despite much buffered type out.

C.2.b |G|G|G This causes command level to be entered immediately, probably from within an I-O cycle. Proceding the simulation will return to whatever was happening while a ;G |Sec VII| will start simulation at the beginning of

an instruction.

D. |Q (Quote) This character causes the immediately following character to be treated as an ordinary letter at interrupt level and also at the main program level of command level. A |Q will be invisible to a program being simulated but the following character will always be visible. It is useful for inputing any of the special characters listed in this section. It may also be used to put space, comma, colon, and semicolon into filenames. Rubbing out a character quoted by a |Q also automatically rubs out the |Q.

E. |S (Silence) This character deletes all typeout until the time it is invisibly read at the main program level.

F. |V (Voice) Turns on typeout. Typeout is also switched on by some serious errors.

G. |W (Wisper) Turns off typeout, except for most error messages which bypass this switch.

IV. Expressions in 11SIM DDT

Expressions in 11SIM DDT are composed of syllables and operators. Syllables are composed as follows:

<digit>::= 0| 1| 2| 3| 4| 5| 6| 7| 8| 9
<letter>::= A| B| C . . . X| Y| Z| .| $| %
<octal-number>::= <digit>| <digit><octal-number>
<decimal-number>::= <digit>.| <digit><decimal-number>
<ASCII-value>::= '<any-characer>| "<any-character><any-character>
<symbol>::= <letter>| <letter><octal-number>| <symbol><symbol>

<syllable>::= <octal-number>| <decimal-number>| <ASCII-value>| <symbol>

In the above, <any-character> means either any ASCII character not mentioned in section III and not rub-out, or a |Q followed by any mentioned in section III or rub-out. Rub-out is used to implement a character at a time input backup feature.

The value of a syllable (assuming no undefined symbols) is a 16. bit number and a flag that says if it is a register number. A register value can come only from a register type symbol. The only initial register symbols are %0, %1, . . . %6, %7. An expression is register valued if and only if it contains a term which is a register valued syllable.

Expressions are composed of syllables and certain operators, with the indicated precedence, as follows:

<term>::= <syllable>| <value>
<expr1>::= <term>| -<term>| +<term>
<expr2>::= <expr1>| <expr1>*<expr1>| <expr1>!<expr1>
<expr3>::= <expr2>| <expr2>+<expr2>| <expr2>-<expr2>

<expr4>::= <expr3>| <expr3>&<expr3>| <expr3>¬<expr3>| <expr3>#<expr3>

<expression>::= <expr4>| <expr4><space><expr4>

In the above "!" signifies division, "&" means logical and, "¬" (back slash) means inclusive-or, and "#" means exclusive-or. A <space> in an expression performs the operation of addition.

Since the PDP-11 has multi-word instructions, the command level DDT can handle multi-word values. These may be composed by an op-code or with angle-brakets as shown below:

<expr-sequence>::= <expression>| <expression>,<expr-sequence>

<value>::= <expression>| <op-code-with-operands>| "<"<expr-sequence>">"

For an expr-sequence at the top level, each expression is assembled as a successive word. In an embedded expr-sequence, only the first expression is used. Surrounding register value expressions with angle brakets strips off their registerness.

Op-codes may variously have zero, one, or two operands and the operands may be of various sorts. The usual PDP-11 conventions are followed |Ref 1| except for the set and clear flag instructions where SFL or CFL followed by a space, an atsign, and the appropriate flag letters is used (i.e. SFL @C for SEC or CFL @Z for CLZ). In genral, expressions of the right registerness may be used in operands. Where a register is expected and an ordinary value is supplied, it is treated

as a register value (i.e. RTS 7 is equivalent to RTS %7).

The following are some special symbols whose values are automatically set.

.	Current location [Ref 4].
%.	Location actually open [Sec V].
%	Set at entry to simulated program counter.
%OPC	Address of last or current instruction.
%Q	Has value of last quantity typed in or out.
%B	Address of last breakpoint taken [Sec IX].
%GO	Starting address for ;G with no argument.
%CORE	Number of bytes attached (first non-existant address).
%D	Last explicit destination address typed in or out.
%JPC	Sometimes set to address of last jump or trap [Sec IX].
%L	Length of last quantity typed in or out.
%S	Last explicit source address typed in or out.
%P	Last subexpression in a multiword expression, else %Q.
%CSX	Address used by ;X [Sec VII].
%PATR	Location after instruction patched [Sec VII].
PATCH	Patch space pointer [Sec VII].
%PMODE	Permanent type out mode [Sec V].
%TMODE	Temporary type out mode [Sec V].

Symbols are also defined for various device registers (including the processor "device"). They are listed in the Appendix.

V. Examining and Depositing

In examining and modifying locations, 11SIM command level follows the same general conventions as DDT [Ref 5]. "/" when preceeded by an expression types out the corresponding location in the current mode (see below this section for mode commands). This command changes the values of the symbols %. and ".". If given no argument, the value of %Q is used and only %. is changed. Open square braket is the same as "/" in all respects except that type out is in non-instruction mode.

The "=" command retypes %Q or its argument, if supplied, in numeric form. Close square braket is the same as "=" except that it types out in current mode.

The left arrow (or underbar) command reopens %. in the current mode.

The :REGS command is available to examine all of the eight procesor general registers.

Carriage return closes the currently open location and deposits its argument, if any, there. Horizontal-tab, line-feed, and up-arrow (or circumflex) do the same but then type out the address of and open a new location. For horizontal-tab, this is the location addressed by the previously open location. For line feed, this is .+1 if a byte is open or .+%L, if a word is open, where %L is the length of %Q in bytes. For up-arrow, it is .-1 if a byte is open and .-2 if a

word is open.

There are three sets of type out modes. The one-shot set are refered to during printing and are reset from the temporary set at the end of every non-type-out-mode command. The temporary set are modified to the permanent whenever the simulator types out a prompting "!". This happens when DDT is entered and after a carriage return, a colon command, and some other commands. Thus the one-shot mode affects only the next command.

Each set of type out modes is a group of bits that can be cleared (with a minus sign immediately before the mode character) or turned on for either just the one-shot set (one semicolon), or both the one-shot and temporary sets (two semicolons), or all three (three semicolons). For example, ;-D;;;N will set numeric mode for all sets but clear decimal mode in the one shot set. The following is a complete list of mode characters:

B Byte mode, when opening an even location, examine even addressed byte only. (Examining an odd location always fetches byte only.)
D Decimal mode, when printing a quantity numerically, use base 10. (Normal base is 8.)
I Instruction mode, when opening a location, print as an instruction.
N Numeric mode, overrides symbolic and ASCII modes.
R "Register" mode, stops register values form being printed symbolicly.
S Symbolic mode.
! Non-printing mode, when opening a location, don't print its contents, overrides all others.
" ASCII mode, overrides symbolic.
' Both ASCII and byte modes.

These different modes are relevant at different points in type out. Decimal mode only has effect when it has been decided to output a number. When a location is opened, "!" mode is check first. If it is off, then "I" mode is checked to see if instruction print out should be used. If "I" is off then ASCII and "N" mode are successively tested. (In examining an odd location or a register or when byte mode is on, instruction mode is supressed.) When a quantity is output, the same thing happens except the "!" and "I" modes are not examined. When a quantity is printed symbolicly, it is normally printed as the nearest symbol less than the quantity plus the difference printed numerically. Special things happen for register quantities. These are printed as a symbol only if the symbol matches exactly, "R" and "N" mode are off, and "S" mode is on. If printed numericly a % is printed in front only if the context (in an instruction) does not make it clear that the number is a register.

Numeric and ASCII mode do not affect the output of addresses printed by command level because it is about to open them (i.e., those generated by line-feed, breakpoints, etc.). Expressions typed before a mode setting command are treated as if typed afterward.

VI. Zeroing, Loading and Dumping

Although the simulated core and disk dumping and loading commands load and dump zeros, there exist separate commands to zero or set to any other value all or part of core and to zero the simulated disk. The commands are as follows:

$Z Zero all core words (except top 4K of memory).
exp$Z Set all core words (except top 4K) to exp where exp is a one word expression.
<exp1,exp2>$Z Zero all core words from exp1 through exp2.
<exp1,exp2,exp3>$Z Set all core words form exp1 through exp2 to exp3.

:ZERDF Zeros simulated disk.

Dump files in 11SIM can contain many different kinds of data. In the list below, the term "core" includes zeroes and breakpoint information |Sec IX| but not normally anything in the top 4K of memory. Dump commands write in a format known about by the load commands (farther below, this section) but not acceptable to any normal PDP-11 loader.

All dump commands but :CDMP may be followed by a file specification which consists of symbols separated by space, comma, colon, or semicolon (these characters may be included in a symbol by preceeding them with |Q) and ending with a carriage return. Symbols terminated by a colon or semicolon specify the device and system name respectively. Symbols terminated by a space, comma, or, carriage return are treated as, succesively, the first and second file names, the device, and the system name. Parts of the file specification not

given are supplied from the last load or dump command except for the system name which is taken from the last file command of any sort. The initial file name is |Ω BIN.

:DMPCOR Dump all core (includes breakpoint information |Sec IX|).
<exp1,exp2>:DMPCOR Dump core from exp1 through exp2.

:DMPSYM Dump symbol table.

:DMPTPV Dump the trap and PC change action codes |Sec IX|.

:DMPDF Dump the simulated disk contents (may also happen when detaching DF, see Appendix).

:DMP Dump core, information on amount of core attached, symbols, and TPV, but not disk.

:ODMP Open permanent dump file. After an :ODMP all ordinary dump commands, until a :CDMP, dump into the permanent file.

:CDMP Close permanent dump file (see :ODMP).

$Y Same as :DMP.

All 11SIM load commands, listed below, may be followed by file specifications of the same format as for dump commands (see above, this section). If core being loaded into isn't attached, it will be attached automatically. When core is attached in this way, a "CORE =" print out will occur. If disk information is loaded and the disk is not attached, it will be attached and a message printed. Load commands know about absolute loader format as well as those formats used by the above dump commands.

$L Zero all of core, flush all user symbols, zero cumulative virtual time, then load all items in file (Same as $Z $I :FLSYMS :LOAD but also restores %CSX (:S%CSX)

and sets PATCH to prohibit patching (1>PATCH) |Sec VII| until reset by the load or the user).

:LOAD Load all items in file (simulated disk may also be loaded by attaching it, see Appendix).

:LODCOR Load only core from file. Ignore symbols, TPV, and disk information.

:LODPTR Same as :LOAD but uses 8-bit mode for loading from the paper tape reader.

VII. Execution

This section lists commands that cause or are directly related to actual simulation of PDP-11 instructions. In the case of those commands that cause indefinitly long simulation, control can be returned to command level with a |G |Sec III|. Control will automatically revert to command level if a HALT is executed, a buss error occurs during a buss error trap, or a debugging stop is reached |Sec IX|.

$G Start indefinite simulation at location %G0.
$expG Execute exp instruction starting at %G0.
exp$G Start indefinite simulation at location exp.
exp1$exp2G Execute exp2 instructions stating at location exp1.

$N Proceed simulation for one instruction, then return to command level.
exp$N Proceed simulation for exp instructions, return to command level at end if nothing has caused an earlier return to command level.

$P Proceed simulation indefinitely.

$X insn (terminate by carriage return) Executes the instruction insn by storing it at location %CSX (initially 177720). If insn causes a transfer, simulation will continue until control returns. %7 is always restored on return.
$X Same as $X insn but executes the instruction currently at %CSX.
exp$X insn Same as $X insn except that exp is put in %1 before executing the instruction and %1 is restored on return.
exp$X Same as exp$X insn except it uses the instruction currently at %CSX.

exp:S%CSX This command allows the user to move the location used by $X commands to location exp, exp+2, and exp+4. If moved into attached memory below the top 4K, the %CSX location will be properly dumped and loaded |Sec VI|.

$I This is a reinitialization command that simulates a RESET instruction, zeros the cumulative virtual time and zeros the processor status register.

:BUSS The simulator remembers various information about the last up to three buss errors. This information, which includes the simulated time of the error, is printed along with the current time by :BUSS.

:PAT exp This command can be given only when a location is open |Sec V|. It gives the location special properties and stores exp, which must be a one word quantity, in it. Attempts to read or write the locaction by a simulated program will be treated as stopping breakpoints |Sec IX|. Fetching the location as an instruction will cause a transfer to exp. Examining the location at command level will type out ":PAT" before exp and force exp to be typed as an address. Depositing with DDT will erase the special features. The :PAT command also sets the symbol %PATR to the location of the next instruction (the location line feed would have opened). The properties of :PAT locations are correctly dumped and loaded |Sec VI|.

|\ This command (control backslash) can be given only when a location is open and the symbol PATCH has been defined to point to an even non-register location. It has the effects of :PAT PATCH but also opens location PATCH.

|] This command (control close bracket) is normally used to return to an instruction sequence that was was patched with |\. It has the same effect as :PAT %PATR except that if given an argument it first stores it and then patches after it and it also redefines the symbol PATCH instead of %PATR to point just after the patch it stores.

VIII. Configuration Control

The configuration of the simulated PDP-11 is controlled by the $A and $D commands which attach and detach devices. These commands may be preceded by an optional numeric argument and are followed by a device name. If no device name is given, either command will type a list of legal device names. For some devices, an optional file name can follow the device name. The exact effect and conventions in attaching and detaching all devices are listed in the Appendix except for the CORE "device", which is explained below.

Simulated PDP-11 core is allocated in units of 1024. words. Giving a numeric argument to an attach or detach command on the CORE device will attach or detach that number of 1024. word units. The maximum that may be attached is 28. units. After any attach or detach command on the core device the total amount of core then attached is printed out.

As it is sometimes convenient to have a way of setting the amount of core to a particular value regardless of the current amount attached, there also exists the exp:CORE command. It tries to set the number of attached blocks of core to the value of the expression exp. It then prints the amount of core attached, as it does if given no argument.

IX. Debugging Aids

The two basic bebugging features are the breakpoint feature, described immediately below, and the trap message feature described further below, this section. They have been designed to allow the user to enter command level at a point in the simulated program likely to be that at which it is going astray.

The breakpoint feature is implemented by associating a seven bit qunatity with each simulated PDP-11 word. Any reference to memory by a simulated program or direct to memory input-output device can activate a breakpoint if the appropriate bits are on in the quantity associated with the word it is referencing. These bits are as follows:

```
  1   Breaks on processor instruction fetches.
  2   Breaks on processor write references.
  4   Breaks on processor read references except
instruction fetches.
 10   Upon breaking, if this bit is set, continue
simulation, else enter command level.
 20   Breaks on direct to memory device write.
 40   Breaks on direct to memory device read.
100   Breaks on processor instruction fetch, always
enters command level, and clears itself.
```

When a break occurs, a message is typed showing various information normally including the instruction which "caused" the break. In the case of instruction fetch breakpoints that enter command level, this is the instruction fetch of which caused the break. Autoflush breakpoints, indicated by the 100 bit, always enter command mode and delete themselves. Otherwise, if the 10 bit is off command level is

entered.

The break message will also contain the code R, W, or I as the break was caused by a read, write, or instruction fetch. RW will appear if the referencing instruction was performing a read-pause-write cycle. For direct to memory references, the codes NGDIS RW for the display, DSK R and DSK W for the disk, and DM R, DM RW, and DM W for the teletype multiplexer will appear.

The commands to insert and remove breakpoints are $B, $F, $R, $W, and $U. These commands can take two expressions as arguments, one just before and one just after the "$". If no first argument is given, the command will affect all locations where breakpoints can be set. If a single word value first argument is given, the command will affect only the location with that address. If a multi-word value first argument is given, the locations from the first sub-value through the second sub-value will be affected.

With no second argument, just after the "$", $B will put in a break of type 7 (print message and enter command level on any processor reference) while $U will remove a breakpoint by making it type 0 (never break). $F, $R, and $W insert breakpoints of type 1, 4, and 2 repsectively to trap just instruction fetches, just other reads, or just write references. This is, in fact, the only difference between these commands as, with a second argument, all use the bottom

seven bits of the second argument as the break type to set.

The trap message feature allows the user to be notified and, if he desires, to enter command level on any PDP-11 trap or transfer of control. There is an array with an entry for each even word address from 0 thru 374. These are the type of locations used for "trap vectors" in the PDP-11. Each entry in this array contains one of the following trap action codes:

0 Do nothing (continue simulation).
1 Set %JPC from %OPC and continue.
2 Print message and continue.
3 Print message and enter command level.

The $T command is used for setting these codes. It can be given the location it is desired to influence traps through as a first argument (just before the "$"). If this argument is omitted, traps through all locations will be affected. A second argument (just after the "$") may be supplied which is the trap action code to be set. If this argument is omitted, 0 is assumed.

Since the PDP-11 does not trap through location 0, this slot in the trap action array is free and is used to allow any of the above trap actions on all transfer of control instructions. The inital trap vector state is 0$1T.

X. Miscellaneous

A. Symbols

At any particular time, there is symbol table in the simulator used by the command level DDT |Ref 4|. It contains op-codes, special symbols, and user defined symbols. The user defined symbols are usually initialized by a load command and are saved by some dump commands |Sec VI|. The following commands are useful in manipulating user symbols:

:LSYMS Lists all user symbols and some special symbols with their values.

:FLSYMS Flushes all user symbols.

exp>sym Defines sym with value exp (should be a one word quantity).
>SYM Defines sym with value %Q.
exp>sym|K Defines sym with the value exp but does not allow it to be used in symbolic typeout (half-kills the symbol).

|Ksym Complements half killed status of sym.

:LOOKUP Should be followed by a symbol. Valrets a string to HACTRN |Ref 5| which causes it to type out the symbol table entry for the symbol.

B. Time

A cumulative simulated run time is kept by the simulator. Simulated run time does not pass in command level.

:TIME Prints cumulative simulated run time. This is also printed by the :BUSS command |Sec VIII|.

:ZTIME Zeros the cumulative simulated run time. This is also done by $I |Sec VII| and $L |Sec VI|.

C. Other

;ALTSEM This command switches the effect of ; and $ in indicating types of commands. It prints out NORMAL or

SWITCHED as the state is the same or different from that indicated in this manaul.

:DEBUG This command complements a switch and prints out ON or OFF depending on the switch's new state. This switch is initially off. When on it enables the print out of the simulation program's program counter at certain critical points.

:LCOMS This command lists all ":" commands |Sec XI|.

:WALLP This command has the effect of |B but takes an optional file specification which controls future output to the "line printer". This file is also used by future |B's. |E terminates "console" output started by a :WALLP.

:XFILE This command takes a file specification like that used for dump and load commands |Sec VI|. It treats the file as text commands to be executed by the simulator command level. Characters in the file are echoed on the users terminal as they are read. |B, |E, |V, and |W are effective |Sec III| from the file as well as from the terminal. Any errors cause input to be taken from the termianl again (this includes |G typed on the terminal). $G, $N, and $P termiate usage of the command file.

XI. Index to Commands

|B Section III.
|E Section III.
|G Section III.
|I(tab) Section V.
|J(line-feed) Section V.
|K Section X.
|L Clear screen.
|M(carriage-return) Terminates file specifications and some commands Section VI, Section V.
|N Same as $N, Section VII.
|P Same as $P, Section VII.
|Q Section III.
|V Section III.
|W Section III.
|Z Reference 3.
|¬(control backslash) Section VII.
||(control close braket) Section VII.
$(alt-mode) Starts a class of commands indicated by the following character. See commands in this list ASCII ordered by character.
! Section IV.
;!, ;;!, ;;;! Section V.
" Section IV.
;", ;;", ;;;" Section V.
Section IV.
% Section IV.
& Section IV.
' Section IV.
;', ;;', ;;;' Section V.
* Section IV.
+ Section IV.
- Section IV.
/ Section V.
: Starts a class of commands indicated by the following symbol. See commands in this list ASCII ordered by symbol.
; Starts a class of commands indicated by the next following non-semicolon character. See commands in this list ASCII ordered by character.
< Section IV.
= Section V.
> Section IV, Section X.
$A Section VIII.
:ALTSEM Section X.
;B, ;;B, ;;;B Section V.
$B Section IX.
:BUSS Section VII.
:CDMP Section VI.

:CORE Section VIII.
;D, ;;D, ;;;D Section V.
$D Section VIII.
:DEBUG Section X.
:DMP Section VI.
:DMPCOR Section VI.
:DMPDF Section VI.
:DMPSYM Section VI.
:DMPTPV Section VI.
$F Section IX.
:FLSYMS Section X.
$G Section VII.
;I, ;;I, ;;;I Section V.
$L Section VI.
:LOAD Section VI.
:LODCOR Section VI.
:LODPTR Section VI.
:LOOKUP Section X.
:LSYMS Section X.
;N, ;;N, ;;;N Section V.
$N Section VII.
:ODMP Section VI.
$P Section VII.
:PAT Section VII.
$R Section IX.
;R, ;;R, ;;;R Section V.
:REGS Section V.
;S, ;;S, ;;;S Section V.
:S%CSX Section VII.
$T Section IX.
:TIME Section X.
$U Section IX.
$W Section IX.
:WALLP Section X.
$X Section VII.
:XFILE Section X.
$Y Section VI.
:ZERDF Section VI.
:ZTIME Section X.
|(open braket) Section V.
¬(back slash) Section IV.
|(close braket) Section V.
|(up arrow) Section V.
_ Section V.

References

1. 1969 "pdp-11 handbook" Digital Equipment Corp. |DEC-112X 01269 AJO F 11 50|

2. 1971 "PDP-11/45 Handbook" Digital Equipment Corp. |Preliminary Edition|

3. 1968 "PDP-10 System Reference Manual" Digital Equipment Corp. |DEC-HGAA-D|

4. 1969 "ITS 1.5 Reference Manual" Donald E. Eastlake et al. |AI Memo 161A|

5. 1970 "DDT Reference Manual" Eric Osman |AI Memo 147A|

Appendix of Device Registers

This appendix lists all device registers (pseudo-memory locations with magic properties) simulated by 11SIM. They appear in order of increasing address and are grouped by "device".

Each lettered section represents a device whose symbolic name or names, if any, appear in all capital letters after the section letter. Such names are the legal device names to use with the $A attach and $D detach commands |Sec VIII|. After the section heading line, a list of interrupt vectors and priorities, if any, for that device appear. Then there is an explanation of the details of the device's simulation, attachment, and detachment. In all cases except the Knight display, it is assumed that the user is familiar with the device's normal operation from manufacturer literature. For the Knight display (section A) a fuller explanation is given.

Finally, each section has a list of register locations and bits in each register. Just after the register's octal address and just after each subfield or bit will appear one of the following codes:

RO Read Only. This register or field is unaffected by writting into it. The data in it can only be read.

R/W Read Write. These bits can be read or written by the user's program.

WO Write Only. This register or field is always read as zero but has some effect on being written into.

NS Not Simulated. These bits are not simulated. For a register, this means that the location will be non-existant. For a bit or field, the bits are always read as zero and writting them has no effect.

The symbolic name of each register appears in all capital letters after the above code after the octal address of the register. This name, with a % prefixed, is a defined symbol in 11SIM's DDT whose value is the address of the register. A general description of the register is given after its symbolc name and a descrition of each used bit or field in the register is given below the main register line. All bit numbers are decimal and the notation n:m means bits n through m inclusive of bits n and m.

Unless otherwise stated, all the register locations associated with a device will be non-existant when the device is not attached and attaching a device already attached is identical to detaching it and then attaching it.

The simulation routines for these devices were written by the author of this memo except for the ET device which was implemented by Richard M. Stallman.

A. NGDIS, The Knight Display

270=interrupt vector 5=priority

The Knight display controller allows up to eight consoles to be attached to a PDP-11. In the simulator, this device is not initially attached. When attached, it uses the DEC 340 display to simulate three consoles (number 0, 1, and 2) in three quadrants of its screen. The fourth, lower right hand quandrant displays simulator information including the output from a :REGS and :BUSS command. Each word fetched for a particular display console as a display command is decoded as follows:

bit 15	14	
0	0	No-op, rest ignored.
0	1	Increment command. 13:11=direction. 10:8=count. 7:0=bits.
1	0	Pushj. 13:0=destination word address.

bit	13	12	11	
1	0	0	0	10=reset X. 9=reset Y. 8=stop. 7=pop. 6=popj.
1	0	0	1	10=intensify. 9:0=delta Y.
1	0	1	0	10=intensify. 9:0=delta X.
1	0	1	1	10=intensify. 9:0=delta X and Y.
1	1	-	-	No-op, rest ignored.

The reset X and reset Y functions set the appropriate coordinate registers to zero. The point (0,0) is at the center of the screen and the upper right quadrant is +X, +Y. Each coordinate register is eight bits.

In increment mode, the count field (in which zero implies eight) is the number of bits starting from bit 7 down that are used as follows:

dir	bit	X	Y	dir	bit	X	Y
0	0	0	+1	4	0	0	-1
0	1	+1	+1	4	1	-1	-1
1	0	+1	0	5	0	-1	0
1	1	+1	+1	5	1	-1	-1
2	0	+1	0	6	0	-1	0
2	1	+1	-1	6	1	-1	+1
3	0	0	-1	7	0	0	+1
3	1	+1	-1	7	1	-1	+1

The direction can be considered to be one of the eight compass points clockwise numbered with zero north. These points are then rotated clockwise 22 and a half degrees. Then a zero bit means the nearest of N, E, S, or W and a one bit means the nearest of NE, SE, SW, or NW. A point is displayed at the end of each increment.

In the delta X and Y commands, bit 9 and 8 have no effect currently but are reserved for future expansion of the coordinate registers. Bits 7 through 0 are 2's complement added to the appropriate coordinate register(s). A point is displayed at the new coordinates if the intensify bit is on.

The lower 14. bits of a pushj type display command are shifted left one to give the (unrelocated) byte address that is transfered to. A simple transfer is accomplished by pushjing to a pop command. The stack pointer (unrelocated) used by the display processor for each console is found in location 16. plus twice the console number.

164040 R/W NGCSR, Control and status register
0:7 R/W Run bits for each console (0 for console 0). Setting bit starts console and sets console PC (unrelocated) to the console number times two as a byte address. Clearing bit aborts output on console. Bit cleared by controller when console hits a stop command.
8 R/W Interrupt enable. Allows interrupt on run bit going to zero for any console.
9:11 RO Number of last console to get a non-existant memory error.
12 R/W NXM interrupt enable. If set, bit 15 on will interrupt and suspend all displaying even though some of bits 0:7 remain on.
13:14 R/W Relocation extension bits. See NGREL.
15 R/W Non-existant memory error. Can be set for checking purposes.

164042 R/W NGREL, Relocation pointer.
1:15 R/W Relocation for display addresses, extended by NGCSR bits 13:14. All memory locations fetched from or stored into by the Knight display have their address relocated by the addition of the relocation pointer.

B. PK, The KW11-P Programmable Clock
104=interrupt vector 6=priority

This device is not initially attached. It does not take any file name after it in an attach command but a non-zero prefix numeric argument will be taken as the "line frequency" for rate 2 operation. This is initially 60. The cumulative simulated time is used to drive the PK device so its actions correspond to simulated real time.

172540	R/W	PKCSR, Control and Status Register
0	R/W	Run, allows clock to run, cleared by underflow in Mode 0.
1:2	R/W	Rate select as follows: 1 2 rate 0 0 100,000 Hz 0 1 10,000 Hz 1 0 Line frequency. 1 1 External.
3	R/W	Mode, if zero underflow stops clock by clearing bit 0, if one clock continues counting after underflow.
4	R/W	When zero, clock counts down, if one, clock counts up.
5	R/W	Setting this bit causes clock to count by one.
6	R/W	Interrupt enable, allows bit 7 to interrupt.
7	R/W	Turned on by underflow.
15	RO	Turned on by underflow with bit 7 set, cleared by clearing bit 7.
172542	WO	PKCSB, Counter Set Buffer
0:15	WO	Sets counter, cleared by underflow in mode 0, used to restore counter on underflow in mode 1.
172544	RO	PKC, Counter
0:15	RO	Current state of counter.

C. DC, DCR, DCT, Asyncronous Line Interface
304=interrupt vector 5=priority (receive)
310=interrupt vector 5=priority (transmit)

This device is not initally attached. It may be given a file specification argument when attahced. Using the DC device attaches the same file for both receive and transmit. The DCR and DCT devices allow file specifications for just receive or transmit. All registers exist if either DCR or DCT is attached.

If a teletype is attached for input, characters will be available as 11SIM gets teletype interrupts. Characters will not be lost due to not reading them fast enough. Non-teletype input and any output is allowed every ten simulated microseconds. Characters output when ready is not on will be lost.

The DC device uses dynamically allocated input-output channels as discused under the DM device below.

174000	R/W	RCSR, Receiver Status Register
0	NS	Data terminal ready.
1	R/W	Break. In simulator, sends a zero character when set.
2	RO	Carrier detect. In simulator indicates DCR attached.
3:4	R/W	Receiver speed. No effect in simulator.
5	RO	Parity. In simulator always zero indicating even parity.
6	R/W	Interrupt enable, allows interrupt on bit 7.
7	RO	Done, indicates character available in RBUF.
8	NS	Supervisory transmit data.
9:10	R/W	Character length. No effect in simulator.

9	10	bits
0	0	8
0	1	7
1	0	6
1	1	5

12	NS	Receiver overrun.
13	NS	Ring indicator.
14	RO	Carrier transition. Cleared by reading RCSR. In simulator set when DCR attached or detached.
15	RO	Error. OR of bits 12:14.
174002	RO	RBUF, Receiver Buffer
0:7	RO	Input character. Reading clears RCSR bit 7.
174004	R/W	TSCR, Transmitter Status Register
0	R/W	Request to send. No effect in simulator.
1	RO	Clear to send. In simulator indicates that DCT attached.
2	NS	Maintenance.

3:4	R/W	Transmitter speed select. In simulator no efffect.
6	R/W	Transmitter interrupt enable.
7	RO	Ready. Indicates ready to accept character in TBUF.
8	R/W	Stop code. 0=2 bits. 1=1 bit. No effect in simulator.
15	NS	Supervisory receive data.
174006	WO	TBUF, Transmitter Buffer
0:7	WO	Transmitted character. Loading clears TSCR bit 7.

D. DM, DMR, DMT, Asyncronous 16-line Single Speed Multiplexer
314=interrupt vector 5=priority (receive)
320=interrupt vector 5=priority (transmit)

This device is not initially attached. A line number from zero to fifteen should be supplied as a prefix argument to any attach command and a file specification may be supplied after each attach command. An attach on the DM device tries to attach the same file for both receive and transmit on the specified line. The DMR and DMT devices may be used to attach receive and transmit sperarately. A line number from zero to fifteen is also usually given with a detach command. DMR and DMT may be used to detach just receive or just transmit. If no line number is supplied on an attach or detach, nothing is attached ordetached. Instead, a print out of the status of the DM lines is made.

If a teletype is attached for input, characters will be stored into memory as 11SIM gets teletype interrupts. Characters may be lost due to tumble table wrap around. Non-teletype input and any output is activated every three simulated milliseconds.

The DC and DM devices use dynamically allocated input-output channels. Each receive or transmit attachment on each line uses up one channel unless the device attached is TTY or LPT |Ref 4|. Currently there are a total of six channels available.

175000 R/W CSR, Control and Status Register
0 R/W Receiver enable. When clear disables all 16 receivers.
1 R/W Full-half duplex select. Ignored in simulator.
2 NS Maintenance.
4:5 R/W TBR extension bits. These bits also extend all other address produced by the DM. If on in the simulator they cause a timeout error.
6 R/W Receiver interrupt enable. Allows bit 7 to interrupt.
7 R/W Done. Set as each character is received.
12 R/W Transmit interrupt enable. Allows bits 13:15 to interrupt.
13 R/W System overrun. DM unable to get unibus in time. May be set for checking purposes.
14 R/W Timeout. DM non-existant memory. May be set for checking purposes.
15 R/W Transmit done. Set as any bit of BAR goes to zero.

175002 R/W BAR, Buffer Active Register
0:15 R/W One bit per line. Setting starts transmission. Clearing aborts transmission. Bit set to zero

when character count decremented to zero (see TBR).

175004 R/W BCR, Break Control Register
0:15 R/W One bit per line. Setting causes continuous break tranmission on line until bit cleared by program or reset.

175006 R/W TBR, Table Base Register
8:15 R/W This is the relocation pointer for the tables in memory used by the DM device. It is extended by CSR bits 4:5. Since the bottom eight bits are always zero, the TBR is a multiple of 400. The first sixteen locations it points to are the current byte address of characters being transmitted to each line. The next sixteen locations are the tranmission byte counts. The third set of sixteen are not used in the simulator but are normally the bit assembly words. The fourth set of sixteen words is never used but the following sixty four are the receiver tumble table. A internal pointer in the DM circles through these words and is incremented once as each character received is stored into a word. The word will have bit 15 on and will have the line number in bits 9:12. In the simulator, bit 13 will always be on and bit 14 always off. On a real DM these bits indicate parity and break respectively.

E. EAE, Extended Arithmetic Unit KE11-A

This device is initially attached. Delays caused by EAE operations are not simulated. Byte mode references can write in the simulated status register.

177300 WO DIV, Divide
0:15 WO When the divisor is moved to this address, the 32. bit dividend in the AC & MQ is divided by this number.

177302 R/W AC, Accumulator
0:15 R/W High order word of arithmetic unit. Contains low order product on multiply, remainder or high order dividend on divide.

177304 R/W MQ, Multiplier Quotient
0:15 R/W Low order word of arithmetic unit. Contains low order product or quotient on divide. When written into, its sign is extended into the AC.

177306 WO MUL, Multiply
0:15 WO When the multiplicand is loaded into this address, the EAE multiplies this number by the number in the MQ.

177310 R/W SCSR, Step Count and Status Register
0:5 R/W The step count which contains the count for long shifts and the step count following normalize.
8 R/W On shifts, set to last bit shifted out of AC & MQ.
9 RO Indicates that result is a single word in the MQ. (Or of bits 12 and 13, this word.)
10 RO Indicates that result is zero. (And of bits 11 and 12, this word.)
11 RO MQ is zero.
12 RO AC is zero.
13 RO AC is all 1's.
14 R/W Indicates result negative.
15 R/W XOR of operation causing overflow and bit 14.

177312 R/W NOR, Normalize
0:15 R/W Writing into this address results in the 32. bit number in the AC & MQ being normalized. This usually means shifting left until the top two bits are different but certain special numbers are checked for and handled differently. Reading this address fetches the shift count.

177314 WO LGS, Logical Shift
0:5 WO Output to this address results in a logical shift of the AC & MQ (filling with zeros) the specified number of bits.

177316 WO ARS, Arithmetic Shift
0:5 WO Output to this address causes an arithmetic shift of the AC & MQ (sign extension) with the shift count being the value moved to this address.

F. ET, DECtape
214=interrupt vector 6=priority
The simuator command structure for attaching this device is not yet settled.

177340	RO	TCST, Control and Status Register
0:1	NS	Extended Data.
2:4	NS	Data Tracks.
5:6	NS	Maintenance.
7	RO	Tape up to speed. Cleared when unit select or reverse bit changed.
8	RO	Non-existant memory. Cleared in the same manner as bit 15.
9	NS	Data missed.
10	RO	Block missed.
11	RO	Selection error.
12	RO	Illegal operation.
13	NS	Mark track error.
14	NS	Parity error.
15	RO	End zone. Cleared when a zero is loaded into bit 15 or TCCM.

177342	R/W	TCCM, Command Register
0	WO	Starts action, clears ready.
1:3	R/W	Command as indicated below:

#	Function
0	Stop all motion.
1	Read block number.
2	Read data.
3	Read all.
4	Stop motion in selected transport.
5	Write timing and mark track.
6	Write data.
7	Write all.

4:5	R/W	Extend bits of buss address.
6	R/W	Interrupt enable.
7	RO	Ready.
8:10	R/W	Unit select.
11	R/W	Tape direction.
12	R/W	Delay inhibit.
13	NS	Maintenance.
15	R/W	Error.

177344	R/W	TCWC, Word Count
0:15	R/W	Two's complement word count.

177346	R/W	TCBA, Buss Address
0:15	R/W	Buss address for memory transfers.

177350 NS -, Data Register

G. DF, The RF11/RS11 Disk

204=interrupt vector 5=priority

This device is not initally attached. It is the only device that may be given a file argument on attaching but acts differently if not given a file argument. All other devices that take file specifications simply use some initial file as changed by previous type-in if no file specification is supplied. This device is simulated by storing the disk contents in the simulator's core image. Two PDP-11 words are stored per PDP-10 word.

The DF device may be attached by a load command |Sec VI| which loads disk contents information. A message is printed and the information is loaded into the simulated disk. If an attach command on the DF device is given a file argument, the disk is attached and disk information only is loaded from the specified file into the simulated disk. An attach command with no file specification attaches a zeroed disk. The :ZERDF command is also available to zero the disk |Sec VI|.

Detaching the DF with a file name simulates a :DMPDF |Sec VI| and then expunges the disk contents information in core. If no file is specified and the user has not done a :DMPDF then the disk contents is lost by detaching the DF device.

Although in the simulator a disk transfer happens instantaneously in terms of cumulative simulated time, the ready flag will not normally come on (possibly causing an interrupt) until the right amount of simulated time has past. The user can activate another transfer by setting go even when ready is off.

177460 R/W DCS, Disk Control and Status

0 WO Setting causes the disk to execute the function selected by bits 1:2, clears bits 8:13, and clears bit 15 unless bit 14 is set.

1:2 R/W Select one of three functions as shown in this table:

2	1	Function
0	0	no operation
0	1	write
1	0	read
1	1	write check

3 NS Maintenance.

4:5 R/W Extended memory bits. Cause disk non-existent memory error unless both on when addressing an I-O register.

6 R/W Interrupt enable. Allows bits 7 and 15 to interrupt.

7 RO Ready.

8 WO Setting causes "power clear" of disk.

```
        9   NS   Missed transfer.
       10   NS   Write lock out.
       11   RO   Non-existent disk.  On if DAE bits 2:4 are
non-zero.
       12   NS   Data parity error.
       13   RO   Write check error.  On if word read from
memory and word read from disk differ during a write check
operation.
       14   RO   Freeze.  Or of DAE bits 10:15.
       15   RO   Error.  Or of DCS bits 9:14.

   177462   R/W  CMA, Current Memory Address
     1:15   R/W  Current memory address, initially loaded with
the starting address minus two, incremented after each word is
transfered unless DAE bit 8 is on.

   177464   R/W  WC, Word Count
     0:15   R/W  Word count, loaded with 2's complement of
block length and incremented by 1 before each transfer.
Overflow ends a function.  Limits block size to 65,536 words.

   177466   R/W  DAR, Disk Address Register
     0:10   R/W  Selects word of disk track for next transfer.
Incremented after each transfer.  Overflow carries into track
address.
    11:15   R/W  Selects track for next word to be transfered.
Carries into and is extended by DAE bits 0:1.

   177470   R/W  DAE, Disk Address Extension and Error
      0:1   R/W  Track address extension, carries into bit 2.
      2:4   R/W  Disk address, in the simulator causes non-
existant disk error if non-zero, carries into bit 5.
        5   RO   Disk address overflow, causes non-existant
disk error.
        7   NS   Data request late.
        8   R/W  CMA increment inhibit.
       10   RO   Non-existant memory on disk reference.
       12   NS   C timing track error.
       13   NS   B timing track error.
       14   NS   A timing track error.
       15   NS   Address parity error.

   177472   NS   -, Data Buffer Register

   177474   NS   -, Maintenance Register

   177476   RO   ADS, Address of Disk-Segment
     0:10   RO   Real time disk position.
```

H. LP, Line Printer
200=interrupt vector 4=priority
The device is not initially attached. It allows output to the "line printer" through the same path as that enabled by |B |Sec III|. It may be given a file specification to output to which will effect future |B or :WALLP output |Sec X|. Detaching the LP does not terminated |B or :WALLP output. Characters may be output every 6 simulated microseconds. Characters output when done is not on will be lost.

177524	R/W	LPS, Line Printer Status
6	R/W	Interrupt enable, allows bit 7 or 15 to interrupt.
7	R/W	Done. Indicates ready to accept character in LPB.
15	RO	Error on output.
177526	WO	LPB, Line Printer Buffer
0:6	WO	Character to be printed. Loading clears LPS bit 7.

I. LK, The KW11-L Line-Frequency Clock
100=interrupt vector 6=priority
This device is initially attached. It takes no file name after an attach command on it but a non-zero prefix numeric argument will set the "line frequency" used it its operation. This is initally 60. The cumulative simulated time is used to drive the LK so its actions correspond to simulate real time.

177546	RO	LKS, Line Clock Status
6	R/W	Interrupt enable, aloows bit 7 to request interupt.
7	RO	Set to one every cycle of line current, cleared by reading.

J. PR, High Speed Paper Tape Reader
70=interrupt vector 4=priority
This device is not initially attached. It takes a file specification to read from. A nonzero numeric argument will set the PR to rad in image or ASCII mode as the argument is one or greater than one. Image mode is the initial setting and must be used to read properly from the PTR [Ref 4] which is the initial file specification. A character may be read from the PR device every 6 simulated microseconds but no data is lost if characters are not read fast enough.

177550	R/W	PRS, Paper Tape Reader Status
0	WO	Reader enable, requests read of next character, can be set only if bit 15 clear, setting clears PRB and sets bit 11.
6	R/W	Interrupt enable, allows bits 7 and 15 to interrupt.
7	RO	Done, set by character available, cleared by reading PRB.
11	RO	Busy, set by setting bit 0, cleared by bit 7 coming on.
15	RO	Error, set by end of input.
177552	RO	PRB, Paper Tape Reader Buffer
0:7	RO	Reader character buffer.

K. PP, High Speed Paper Tape Punch
74=interrupt vector 4=priority
This device is not initially attached. It takes a file specifcation to write on. A non-zero prefix numeric argument will set the PP to read in image or ASCII mode as the argument is one or greater than one. Image mode is the initial setting and must be used to write properly on the PTP device [Ref 4] which is the initial file specification. Characters may be output to the PP device every 30. simulated microseconds. Data output when ready is not on is lost.

177554	R/W	PPS, Paper Tape Punch Status
6	R/W	Interrupt enable, allows bits 7 and 15 to interupt.
7	RO	Ready, cleared by loading PPB, set when ready for more data.
15	RO	Error on output.
177556	WO	PPB, Paper Tape Punch Buffer
0:7	WO	Punch character buffer.

L. TK, Teletype Keyboard
60=interrupt vector 4=priority
This device is initially attached and allows input from the user's teletype to the simulated PDP-11 program. No file specification is taken by this device but a non-zero prefix numeric argument will set the ratio between simulated real teletype speed and real teletype speed. (This is initially set to 1000.) Thus an argument of one would alllow one input character per one tenth second of simulated time or 4 seconds of real time. In the simulator, as opposed to the real PDP-11, data is not lost due to not reading it in time. However, characters buffered by the time-sharing system will be lost on transition to command level.

177560	R/W	TKS, Teletype Keyboad Status
0	NS	Reader enable.
6	R/W	Interrupt enable.
7	RO	Done, character available. Clears bit 11 when set.
11	RO	Busy, character being read.
177562	RO	TKB, Teletype Keyboard Buffer
0:7	RO	Character buffer, reading clears bit 7 in TKS.

M. TP, Teletype Printer
64=interrupt vector 4=priority
This device is initially attached and allows output to the "console" through the same path as command level type out |Sec III|. No file specification is taken by this device but a non-zero prefix numeric argument will set the ratio between simulated real teletype speed and real teletype speed. This is initially 500. (See description of TK device.) Writting in TPB when ready is off will casuse output to be lost.

177564	R/W	TPS, Teletype Printer Status
2	NS	Maintenance control.
6	R/W	Interrupt enable.
7	RO	Ready.
177566	WO	TPB, Teletype Printer Buffer
0:7	WO	Character buffer, clears bit 7 in TPS when loaded.

N. Console Data Switches Register
See references 1 and 2.

177570 RO SWR, Switch Register
0:15 RO This location represents the console data switches. Although read-only for a simulated program, it can be deposited into by command level.

O. Processor General Registers
4=interrupt vector
10=interrupt vector
20=interrupt vector
30=interrupt vector
34=interrupt vector
See references 1 and 2. The processor registers are printed by :REGS.

177700 R/W through 177716, %0 through %7
0:15 R/W %0 through %5 are the PDP-11's general purpose registers. %6 is used as the stack pointer for interrupts. %7 is the program counter.

P. 145, PDP-11/45 Features
240=interrupt vector priority variable
With this device attached, as it is initially, the following PDP-11/45 op-codes become available: XOR, MUL, DIV, ASH, ASHC, SXT, SPL, RTT, SOB, and MARK. (Estimated times are used for these operations in advancing the cumulative simulated time.) The PDP-11/45 segmentation option and floating point processor option and their associated instructions are not simulated. See reference 2.

177772	R/W	PIR, Programmed Interrupt Request
1:3	RO	Priority level of highest request (bits 9:15).
5:7	RO	Same as 1:3.
9:15	R/W	Program set requests for interrupts. Bit n is priority n minus 8.
177774	NS	-, 11/45 Stack Limit

Q. Processor Status Register
14=interrupt vector
The trace trap enable bit causes a trap through 14 at the end of most instructions. See references 1 and 2.

177776	R/W	PS, Processor Status
0	R/W	Carry flag (C).
1	R/W	Overflow flag (V).
2	R/W	Zero flag (Z).
3	R/W	Negative flag (N).
4	R/W	Trace trap enable (T).
5:7	R/W	Processor priority. Interrupts of equal or lower priority are inhibited.