USER'S GUIDE TO DATA HANDLING ON THE PDP-1

Raymond De Saussure

August 1, 1967

USER'S GUIDE TO DATA HANDLING ON THE PDP-1

Introduction The PDP-1 is a binary digital computer with a 5 microsecond memory cycle and 4096 words of 18 bit core storage. Auxiliary storage is available in the form of four potter tape units (200BPI density), and two IBM 729-IV tape units (200, 556, 800BPI density). Although an older machine -- first introduced in late 1960 -- it is particularly versatile in I/O conversion, and many forms of data handling are available here that cannot be found elsewhere at LRL.

By modern standards, the PDP-1 is slow and has minimal storage. Arithmetic handling is poor; there is no floating point hardware, and it lacks a compiler although two assemblers PAP and PAL are available. These difficulties are of less importance to data conversion and this is the prime function of the machine. Most input-output operations take in the order of milliseconds, so that a 5 microsecond cycle is not prohibitive. On the other hand, extensive arithmetic operations should be moved to the larger operating systems where they properly belong. The ideal use of the PDP-1 is as a "data-pump" such as the conversion of ASCII paper tape to high density magnetic tape without regard for format, the tape then being processed and manipulated on other machines.

Communication between the PDP-1 and other computers at LRL must be either by means of magnetic tape or else by punched cards. By far the more efficient method is magnetic tape, and punched cards are warranted only if extensive hand manipulation of the data is contemplated. Magnetic tapes are interchangeable between machines only under certain specific conditions which are well defined.

A seven channel tape --all the magnetic tapes at LRL are seven channel-- consists of a series of lines each of which contains six binary bits of information and a seventh parity bit such that the total number of "one" bits is always even or odd. Binary tapes are written in odd parity, and BCD tapes in even parity. Thus, for example, an 18 bit word from the PDP-1 is represented on magnetic tape by three parity checked lines of six bits each. Words are written continuously until a record gap is reached. From the tape alone, there is no way of determining the number of bits in the originating machine. It could as easily be two words of 18 bits each, as one word of 36 bits. Therefore, to read a tape on a different machine than written, one has only to make certain that the total number of bits in the record is evenly divisible by the number of bits in the word of the computer doing the reading. On this basis, it may be seen that a 160 word record written on the PDP-1 may be read by a maximum number of machines currently in LRL computations as long as the read and write operations are in the same density. Various machines at LRL have the following word lengths: Stretch (64), 7094 (36), 3600 (48), 6600 (60), PDP-1 (18). The Larc accepts only a BCD tape (even parity), and the 1401 has a variable word length.

I/O devices available

The following peripheral equipment is attached to the PDP-1:

Magnetic tapes (Potter) Four tape units are available to read 7-channel low density magnetic tape reels of 2400' capacity. This gives a capacity of 1.92 million PDP-1 words exclusive

of record gaps for each tape. A record gap occupies approximately the same space as fifty 18-bit words. The right hand tape unit is a read only unit utilized for systems.

Magnetic tapes (IBM) Two tape units are available to read 7-channel magnetic tapes at densities of 200, 556, or 800 bits per inch. Capacities are proportionally greater so that the high density tape will hold about 7.68 million PDP-1 words, again exclusive of record gaps.

Card reader-punch (IBM model 1402): Reading speed - 800 cards/min.; punching speed - 250 cards/min.

Line printer (Anelex): Prints at 600 or 1034+ lines/min., 120 char./line.

Typewriter (Soroban): 9.5 char./sec. input or output.

Digital plotter (Calcomp 565): X,Y plots on a 10" drum in .01" steps at 3.3 ms. per point.

Paper tape reader (Digitronics): Optical reader for 8-channel paper tape. Can handle reels up to 10.5: diameter. Fan-fold containers are 5x7" and will hold roughly 1/3 to 1/2 of a 1000' box. Tape is read at 400 lines/sec.

Paper tape punch (Teletype): Punches 8-channel fan-fold paper tape at a rate of 63.3 lines/sec. Punches from 1000' box container.

Mylar tape punch (Friden): Punches 8-channel mylar tape at a rate of about 10 char./sec. Input and output are held on spindles.

Rand tablet: Recognizes X,Y coordinates of manually held metallic pen usually operated in conjunction with the visual CRT. May

also be used with paper overlays. Resolution of 100 points/inch on a 10.24 x 10.24" metallic grid.

Visual CRT (DEC type 30): Point display only, except for horizontal and vertical grids with a 50 microsec. settling time. Vectors and characters must be constructed from points. This is a zero centered grid of size 1024 x 1024.

Precision CRT (DEC type 31): This tube is integral to the Eyeball system, and is currently a Litton L4108. A partially reflecting membrane splits the output light into a primary and reference branch with appropriate optics into which 35mm. or 4x5" film may be placed. Light passing through this film is sensed by means of a photomultiplier tube and either returned to the computer as a yes-no pulse or else sent through an analog to digital converter for density readings. The yes-no response requires approximately 50 microseconds per point for a 4096 x 4096 matrix. Persons interested in using the Eyeball should refer to other papers on the subject, and should also hold direct discussions with both the programmers and Eyeball engineers.

Analog to Digital Converter (Redcor 632): Specified as a 40KC converter, this is used at a 25KC conversion rate (40 microsec.) and returns 10 bits + sign. It is currently operational only on the single Eyeball, and has not yet been completed in the dual system. Accuracy is in the order of 10%, although this figure must be defined carefully.

Mouse (Englebart bug) (LRL design): This is a preliminary model of

a manual device that may be rolled about to reflect an X,Y position on the visual CRT. As such, it has properties reflecting those of both the light pen and the Rand tablet. The resolution is 1024 x 1024. Sampling is accomplished by a finger-operated microswitch.

Analog to Digital Converter (LRL model): This converter handles 10 bits at a 33.3KC conversion rate. One channel is currently used with the mouse, and the other three are still available.

Eyeball Film transport (Vought): Capable of reading 35mm film from reels of 3.5" diameter (100' reels) at an actual recommended speed of about 8 frames/sec. This pin registered transport is used on the Eyeball system, usually on the primary arm.

Camera Film (Vought): A pin registered transport for unexposed 35mm film. Advances at a rate of 20 frames/sec.

Unsprocketed Eyeball transport (LRL design): Capable of moving unsprocketed film for Eyeball reading. Recently completed and designed to replace the old Vought transport, the full properties of this device are not yet known.

Light pen (LRL design operating through a DEC 370 photomultiplier): This is a light-weight tip-switch design attached to the visual CRT.

Light pen (DEC): An older model operated by a foot pedal is still available. There is little reason to use this earlier version.

Polaroid camera: Designed to lock into the reference arm of the Eyeball, this may be used to obtain immediate photographs

from the precision CRT.

Voice input (LRL design): A sound power telephone of conventional design has been attached to feed into the Redcor ADC to allow digitizing of sound signals.

Unavailable I/O

As important as the I/O that exists on the machine are those devices and features that are absent. The PDP-1 is unable to read DEC tape, opaque charts, film larger than 4x5", or disc packs. As mentioned previously, the amount of memory manipulation is limited by core size, arithmetic capability, and lack of disc storage. Faster graphics are available elsewhere in the form of the DD80, and further plans are being developed in other areas for graphics capability including opaque reading.

General I/O description

The PDP-1 has what would now be termed a single level priority interrupt and which is called the Sequence Break. When enabled, the I/O sends a completion pulse which traps the computer to location zero with appropriate register storage. The conditions relating to the various I/O devices are determined through interrogation of the status registers in which specific bits in an 18 bit field are set or cleared to indicate the exact state of the device. For example, a "one" in bit 3 of status register zero indicates that a typewriter key has been struck. The typewriter is also one of the devices connected to the sequence break so that the user has the option of either trapping to the sequence break or alternatively staying outside of the trap mode and periodically

checking bit 3. Other devices attached to the sequence break are: paper tape reader (except for the read-in mode), line printer (printing and spacing), card punch, card reader (buffer), light pen, and IBM magnetic tape units (end of word count, and job terminated).

From the user's viewpoint, the I/O may be broadly divided into three major groups as follows:

Group 1: Devices attached to a high speed channel (IBM and Potter magnetic tapes).

In this group, it is necessary to specify an address and a word count. Data is then streamed across in a block transfer mode and the computer is freed for further calculation. The channel operates on a cycle stealing basis which is, in general, unknown to the user. The Potter tapes are capable of reading or writing a non-contiguous record, which is to say that the record may be scattered through different sections of memory. The IBM units lack this property. To date this has not been very useful, and is rarely implemented.

Group 2: Devices having their own controller (Card reader-punch, printer, typewriter, visual and precision CRT, high speed paper tape reader and punch).

Into this group we have gathered the I/O that operates on a wait or proceed basis. Three command sets are generally available to these devices: a wait command, an enable and proceed, and a proceed. These commands take the form respectively of ( )W, ( )C, and ( ). The wait command causes the machine to pause

indefinitely while waiting for an I/O completion pulse. The enable command triggers a completion pulse and continues, while the proceed command simply initiates the I/O and continues leaving any timing considerations to the programmer.

Group 3: Devices attached to the spider, a general purpose, low speed control channel (Light pen interrupt, Rand tablet, Friden punch, Cal Comp, and Mouse). The IBM Selectrics formerly attached have since been removed.

The Spider is a multi-addressing device operating through a low speed channel. A word is loaded into the IO register of the PDP-1 and then sent over the low speed channel. The first two octal digits are decoded to indicate the specific device. The state of the device is indicated by a skip- no skip operation following the low speed channel call. The low speed channel command destroys the contents of the IO register so Spider programming must be carefully implemented. This approach has allowed high expansibility for minor I/O devices with a minimum of design engineering.

Fundamental Arithmetic Concepts

The PDP-1 has an 18 bit word. The first bit acts as a sign bit so that numbers from 0 to 377777 are regarded as positive, and numbers from 400000 to 777777 as negative.

The data word 777777 is equivalent to a minus zero in the PDP-1. This follows from the arithmetic which is one's complement. That is to say that minus numbers in the PDP-1 are obtained by reversing all bits in the word. Thus, to change a +3 to a -3, we have +3 (000 000 000 000 000 011) becoming 777774 (111 111 111 111 111 100).

Counting by ones from -3 to +3 would give the sequence 777774,777775,777776,0,1,2,3.

Arithmetic operations in the machine will usually reset 777777 to 0 so that one must deliberately create the number 777777, either by a negative shift operation, or more readily by the ORed operate commands CLA CMA. The result is that -0 is an excellent number to use as an error indicator or sentinel since it cannot be accidentally derived arithmetically. However, care must be used in all operations involving -0 since many commands operate in an abnormal manner with respect to this number.

One's complement arithmetic is a tricky but arithmetically sophisticated system, which is only fully appreciated after extensive use. For example, if we take a -5 (777772) and add a +2, the result is a -3 (777774). Note that the number scale has moved upward in the minus range. This becomes even more apparent if we take the number 377777 and add 1 so that it becomes 400000 (-377777). In effect, we have wrapped around the word. The overflow condition reflects this logic, for overflow is obtained on the transition from 377777 to 400000 (a change of the sign bit) rather than when 777777 goes to 0. In the ADD command, the overflow flip-flop will be set if and only if the sum of the two like-signed numbers yields a result of the opposite sign. Similarly in the subtract command SUB, the flip-flop is set, when two unlike-signed numbers are subtracted, if and only if the sign of the result does not agree with the sign of the original accumulator.

Boolean commands

The Boolean functions are obtained by the logical commands AND, XOR, IOR and the complement instruction CMA from the operate group. In theory, these commands alone would suffice for programming, but in practice they are much more limited. The primary use of the AND instruction is as a mask. An AND (7777), for example (actually programmed as AND N and N, 7777) will insure that the accumulator contains only the right 12 bits, which is to say the address field. The PDP-1 has limited byte manipulation. One may deposit either the address or instruction part of the word through the DAP or DIP instructions (the latter is almost useless). The entire word must be loaded, however, and the IO is immune to such manipulation or masking.

The exclusive OR (XOR) is used most frequently to reverse the sign bit. It also has some use for reflection in graphics work. Reversing the first bit reflects about the center line, reversing the second bit reflects about a quarter screen line, and so forth.

The inclusive OR has only limited and specific use.

The CMA is used to change sign on the entire word. Thus, if N is in the accumulator, the CMA command will result in -N in the accumulator.

Mictoinstructions

Both the skip group and the operate group on the PDP-1 belong to the class of commands known generally as microinstructions. This means that the programmer can assemble his own command within a group,

and in effect have the portions operate concurrently without loss of time. This is accomplished in either assembler simply by placing the commands next to each other with a space between. As many commands from one group as practical may be ORed in this manner. Examples are as follows:

SPA SPI	(5 microsec.)	Skip if either accumulator or IO is positive.
CLA CMA CLI	(5 microsec.)	Load 777777 into the accumulator and clear the IO.

Skip commands may <u>not</u> be ORed with operate group commands.

<u>Indirect commands</u>

The PDP-1 has multi-level indirect addressing. Care must be used not to cascade past the desired number. The following sequence will load the contents of storage location 7777 into the accumulator rather than loading the contents of A (17777) into the accumulator.

```
LAC I A                    A,   17777
```

A correct use of the indirect command is as follows where the contents of B are loaded into the accumulator:

```
LAC I A                    A,   LAC B
                           B,   777777
```

An alternate possibility is the use of the execute command, XCT, to obtain single level indirect addressing. Thus, the above could simply be written:

```
XCT A                      A,   LAC B
                           B,   777777
```

The indirect has a special meaning when used with the LAW command, namely it is used as a minus sign. By means of either assembly system, LAW I N will load the accumulator with the one's complement (minus) value of N, whereas LAW -N will load the contents of the location -N. For example, LAW I 3 loads the accumulator with 777774 while LAW -3 loads the contents of location 7774.

The indirect command is used with the skip group to indicate a reversal of the skip. SZF I 2 is interpreted as don't skip on flag 2. Any skip command may be treated in this manner.

Unlisted commands

Several commands are available in the assemblers that are not listed in the manual. Among these are the following:

NIX Y (Negative Index) op code 44 (10 microsec.) The C(Y) are replaced by C(Y)-1 which are left in the accumulator. The previous C(AC) are lost. Overflow is not indicated. If the original C(Y) equals 777777, the result is 777776. The NIX command is wired directly into the PDP-1.

SNA (Skip on non-zero accumulator) op code 650100 (10 microsec.) If the accumulator is not zero, the program counter is advanced one extra position and the next command in sequence is skipped.

SNO (Skip on non-zero overflow) op code 651000 (10 microsec.) If the overflow flip-flop is not zero, the program counter is advanced one extra position and the next instruction in sequence is skipped.

SMI (Skip on minus IO) op code 652000 (10 microsec.) If bit

0 of the IO is a one, the program counter is indexed one extra position and the next instruction in sequence is skipped.

SNS (Skip on non-zero switch) op code 6500N0 (650010,650020, ...) (10 microsec.) If the selected sense switch is non-zero, the program counter is advanced one extra position and the next instruction in the sequence will be skipped. 650070 will skip if <u>any</u> sense switch is up.

SNF (Skip on non-zero flag) op code 65000N (65001,65002, ...) (10 microsec.) If the selected program flag is non-zero, the program counter is advanced one extra position and the next instruction in the sequence will be skipped. 650007 will skip if <u>any</u> sense flag is turned on.

<u>Revised commands</u>

LAT (load accumulator from test word) op code 762200 (5 microsec.) Loads the contents of the test word switches into the accumulator. It is no longer possible to OR the accumulator with the test switches.

<u>PDP-1 Systems</u>

The systems tape 0 is customarily kept mounted on the right hand Potter tape unit, which is used as a read only unit. Depressing the read-in switch at the console causes a loop paper tape to bootstrap itself into the upper part of memory and execute. This reads the next record from the system tape 0, and from that record determines the distance to the nearest loading-routine record on the systems tape. The systems tape searches for the loading record, locates it, reads

it into memory, and executes the loader. This causes the on-line typewriter to produce the statement Type ID. At this point, the user types a three letter code followed by a slash. The loader searches its memory for this code, determines where that routine is located on the systems tape, moves to and reads that record, and begins execution of the called-for routine. The three letter codes accepted by systems are the last three alphanumerics typed prior to the slash. An Illegal ID statement can be caused either by erroneous typing or by lack of that routine on the systems tape. A list of these codes is maintained in the PDP-1 console area.

It is important that the read-in switch is not hit at any time while the systems tape is still moving. To do so may cause a runaway systems tape, in which case the machine must be stopped and the systems tape rewound at the tape unit. A few codes cause fairly complex tape movement beyond that described above, so this precaution is important.

Codes that are read into memory from systems load data over existing codes. Unless the systems programmer specifically named a zero location, unused portions are skipped over during the read. For this reason, it is usually advisable to initialize memory with a clear memory command (CLM/) from the systems tape.

Assemblers

The PDP-1 lacks a compiler primarily because of memory limitations. Programs must be written using either the old assembly language PAP, or the newer language PAL. The latter is documented in the I/O manual. Some --but not all-- of the fundamental

differences between PAP and PAL are as follows:

Feature	PAP	PAL
No. of passes in assembler	3	2
Symbolic notation for "self"	*	.
Maximum alphanumerics in name	Approx. 18	6 (first & last 3 of a longer string)
Location definition format (zero)	NAME	NAME,0
" " " (non-zero)	NAME N	NAME, N
Designation for indirect	~	I
Designation for new origin	ORG ( )	*( )
Designation for last card	END	(none required)
Decimal integer format	(none)	XXXXXX.
Card image given by listing	Yes	No (justifies into fields)

In general PAL is slightly faster and has considerably better diagnostics, but is more inconvenient to use because of the listing format.

Loop building

The most efficient loop on the PDP-1 is built with the ISP command. To flow through a loop N times, we use the following:

```
            LAW I (N)                    COUNT, 0
            DAC COUNT
ALPHA,      LAC FWA
              ⎩
            IDX ALPHA
            ISP COUNT
            JMP ALPHA
EXIT,       (as desired)
```

where (N) is the actual positive integer desired for the loop.

The negative of the ISP command does not exist. However, an equivalent may be obtained through a programming trick. For example, let us build a loop where N has a value of 3: i.e. the loop executes 3 times. Note that the order of the exits is now reversed.

```
            LAC COUNTO
            DAC COUNT
ALPHA,      ISP COUNT                    COUNTO,  -3:400000
EXIT,       (as desired)                 COUNT, 0
BETA,       LAC FWA
              S
            IDX BETA
            JMP ALPHA
```

COUNTO contains the PAL format for the exclusive OR of -3 and 400000. Note that the ISP is tested first, and then executes the loop.

The LAC ., ..., IDX portion of the above loops is not essential to the loop, itself, but rather is used to illustrate an extremely common feature of these loops without which the mechanism is usually trivial.

A slightly slower loop, but more efficient in certain cases may be built by storing a positive count in N, and comparing it with another number by means of the SAS or SAD command.

```
            DZM COUNT
            LAW (N)
            DAC COUNTO                   COUNT, 0
ALPHA,      LAC FWA                      COUNTO, 0
              S
            IDX ALPHA
            IDX COUNT
            SAS COUNTO
            JMP ALPHA
EXIT,       (as desired)
```

Subroutines (General)

Although the following conventions are not universal, they form an outline to at least many of the available subroutine packages, and are highly recommended as a guide to interchangeability.

1. Subroutine packages are given a name between 3 and 6 letters in length. All internal subroutine names begin with the same three letters, preferably the first three letters of that name.
2. Subroutines should not require preloading of special locations, but should enter by a calling sequence as will be shown later. There should be at most two entry points --one is far preferable-- and the number of exits should be held to a minimum. These exits should be of the form Exit, Exit+1, ... and should not jump elsewhere from inside the routine. Entry and exit with AC and IO in fixed format are acceptable.
3. First card of the subroutine should be a comment card giving the name of the subroutine and its date. Subsequent comment cards should suffice to define the properties of the subroutine.
4. Second to last comment card in the front should list the internal subroutines required. This may have the form:

   / SSR - (Subroutine 1), (Subroutine 2), ...

5. Last of the initial comment cards should give the calling sequence. Thus, in the subroutine example on page 19:

   / Jsp Beta/ FWA/ WC

6. Cards before and after the calling sequence card above should give conditions on the AC and IO, or the nature of the exits.

Subroutine Formats

Subroutines on the PDP-1 are connected either by a JSP command or a JDA. In both commands, the contents of the accumulator are replaced by the contents of the program counter plus one. The difference is that a JDA saves the former accumulator contents in the location of this address field, and the accumulator contents are lost on executing a JSP. The CAL instruction is completely useless and is entirely equivalent to JDA 100. Use of a subroutine call is illustrated as follows:

```
ALPHA,    JSP BETA                    BETA,   DAP EXIT
                                                 ⎫
                                                 ⎬ (subroutine)
                                                 ⎭
                                      EXIT,   JMP 0
```

The original accumulator is lost. Exit becomes changed to a JMP ALPHA+1.

```
(Accum. loaded here)                  BETA,   ..
ALPHA,    JDA BETA                            DAP EXIT
                                                 ⎫
                                                 ⎬ (subroutine)
                                                 ⎭
                                      EXIT,   JMP 0
```

The original accumulator is saved in BETA; the machine transfers to BETA+1; and EXIT becomes JMP ALPHA+1 as before.

A calling sequence may be constructed along the same lines. The following sequence would initiate a subroutine that requires

both First Word Address, and Word Count as input:

```
ALPHA,    JSP BETA
          .. FWA
          .. WC

BETA,     DAP EXIT
          LAC I EXIT
          DAP GAMMA
          IDX EXIT
          LAC I EXIT
          CMA
          DAC COUNT
          IDX EXIT
GAMMA,    LAC FWA
            ⎫
          (subroutine)
            ⎭
          IDX GAMMA
          ISP COUNT
          JMP GAMMA
EXIT,     JMP 0
COUNT,    0
```

Packaged subroutines

Numerous subroutines already exist as decimal decks. The majority of these have been written in PAP, but many are being converted to the PAL form. In general, conversion is not complex. Also available is a code which will input a PAP decimal deck and produce a packed version of PAL for decimal input. However, it should be noted that the result is not relocatable.

The following routines represent some of the more useful packages, although there are also many other more specialized subroutines. The following abbreviations are used:

FWA	First word address
WC	Word Count
TN	Tape number (logical)
RN	Number of records

I. Tape handling

A. IBM tape units

1. JSP IWRITE/ Class one command*/ FWA/ WC

   Writes IBM tape with variable WC, FWA, TN and density.

2. JSP IREAD/ Class one command*/ FWA/ WC

   Reads IBM tape as above.

3. JSP WINDI/ SFR NOO/ Sentinel

   Rewinds tape N and unloads or not depending on the sentinel status.

4. JSP IBACK/ Class one command*/ RN

   Backspaces a given number of records

5. JSP READBCD/ Class one command*/ FWA/ WC

   Reads an even parity BCD tape.

B. Potter tape units

1. JSP WRITEX/ TN / FWA/ WC

   Writes Potter tape with variable WC, FWA, and TN in low density.

2. JSP READEX/ TN/ FWA/ WC

   Reads Potter tape as above

3. JSP WINDX/ TN

   Rewinds Potter tape unit N. Mechanically incapable of unloading.

4. JSP BACKUP/ TN/ RN

   Backspaces Potter tape a given number of records.

* The class one commands control density and tape number, i.e. SHD 300 is high density on tape 3. Refer to I/O Manual for further details.

II. Printer

1. JSP POCTAL/ FWA/ WC Number of columns

   Prints octal storage of size WC starting at FWA in format with variable number of columns. Zero lines are not suppressed.

2. JSP POCKETA/ FWA/ WC/ Number of columns

   Similar to Poctal except that zero lines are suppressed.

3. JSP DOVER/ FWA/ WC/ One-third number of initial blanks

   Printer comments written in XS3 and stored in memory are picked up and printed with indicated indenting.

4. JSP PRINTOO

   Initializes printer buffer

III. Cal-Comp

1. JDA CALRYT/ N/ Size/ Q

   Writes numbers on Cal-Comp. N is the number desired; Size is an artificial scaling constant; and Q is the number of 90° counter-clockwise rotations.

2. JSP CALVEC/ X/ Y/ Z

   Plots a vector from an initial point to X,Y. Z controls pen movements.

3. JSP CALSO/ Number of times for operation

   Entered with plotting operation in IO and repeats N times.

IV. Typewriter

1. JSP KELLYG/ FWA/ WC

   Types comments from concise code in memory.

2. JSP BIGTYP

Typewriter to printer. Permits programmer to annotate printout as desired. Strikeovers and backup are permitted.

3. JSP TENTYP/ FWA/ WC

Octal storage to decimal typewriter.

4. JSP TIN/ Incorporated subroutine/ WC

Inputs decimal numbers from typewriter while simultaneously maintaining an auxiliary subroutine such as a display. Available with and without sequence break.

V. Paper tape

1. JSP PPT/ FWA

Punches six by five matrix on paper tape.

VI. Card reader-punch

1. JSP PUNCH/ FWA

Punches left 12 bits in word as a vertical column. Data destroyed as punched.

2. JSP READ

Reads cards and assigns a weight to each punched position. Useful as part of a Hollerith read.

VII. CRT and Eyeball displays

1. JSP VIEW

Fast Eyeball scan and playback.

2. JSP TOADX

Fast Analog Digital Converter display of X-sweep.

3. JSP GRIDX/ N

Displays a N x N grid.

4. JSP DCM/ FWA/ WC/ X/ Y/ Size

   Displays comments from memory. Supplementary subroutines required.

5. JDA CIRCLE

   Displays fast circle centered at origin, based on incremental algorithm, and with radius from accumulator.

VIII. Light pen

1. JSP ZAP

   Fast scan of Eyeball with light pen pickup of single point.

2. JSP LPFOL1

   Light pen tracking routine. Requires initialization and has multiple exits.

3. JSP LPS/X/Y/FWA/WC

   Light pen switch. Displays contents of a bank.

IX. Arithmetic routines

1. JDA MAX/ FWA/ WC

   Locates maximum number in bank, returning both number and position.

2. JDA MIN/ FWA/ WC

   Locates minimum number in bank, returning both number and position.

3. JSP RANDOM

   Random number generator.

4. JDA SQROOT

   Extracts square root.

5. JSP ISOMET

   Projects X,Y,Z into X,Y plane.

X. Programming aids

1. JDA SYSTEM

   An internal routine allowing the program to call directly from the systems tape without going through read-in mode or the typewriter.

2. JDA INDEXA

   Pseudo-index which is followed by command to be indexed. Not valid for skip-type commands such as DIV. Index itself is a separate memory location manipulated according to normal rules.

3. JSP CLEARX/ FWA/ WC

   Clears a bank of memory.

4. JSP BLOCK/ FWA old block/ FWA new block/ WC

   Moves a block of memory.

5. JDA DUMP

   Internal debug memory dump.

6. JDA OMD

   An internal memory dump printing eight words/line with double-spacing.

7. JDA FILLUP/ FWA/ WC of filled locations/ Delta on word/ Delta on position number. Will generate almost any form of linear format. Particularly useful for building checkout routines.

8. JDA SSB

   Displays right six bits of accumulator as flags. Useful as visual numerical counter on console that can be independent of calculations.

9. Memorize-Optior-Recall package.

This lies outside the normal format, but permits the programmer to make a sequence of decisions from the typewriter, which sequence is then held in memory for subsequent runs. Permits program adjustment to knowledgeability of user.

XI. Timing routines

1. JSP MMS(Y)

(Y) is a number divisible by 5 between 20 and 100, and represents the number of microseconds delay desired. Routine included in timing.

2. JSP TIMER/ N

N is the number of 100 microsecond delays desired including subroutine.

XII. Parity routines

1. JDA ODDPAR

Checks for odd parity, and turns on flag 1 for parity failure.

2. JDA PARADD/ Parity bit desired

Adjusts words to odd parity through adding desired bit pattern.

XIII. Conversion routines

1. JDA BCDFRID

Converts BCD even parity magnetic tape to Friden Flexowriter format. Enters and leaves with character in accumulator.

2. JSP DECOCT/ FWA/ WC

Interprets a group of BCD numbers starting at FWA as one octal word.

3. JSP XS3OCT/ FWA/ WC

   Same as above except that XS3 numbers are used.

4. JSP OCTDEC

   Enters with octal in IO; exits with decimal BCD in AC and IO.

5. JSP OCTXS3

   Converts octal to XS3. Particularly useful for print routines.

System packages

There are many routines on the systems tape that are particularly useful for general purposes. These routines are described on sheets found hanging near the machine. Among the more useful general routines are the following:

ASP/ General magnetic tape to punched card routine. Inputs magnetic tape of low or high density and with variable record lengths. Will strip a variable number of lead words per record and punch the rest with a variable number of columns per card beginning in column 1. Right 12 bits of each PDP-1 word are turned vertically and punched as a binary column of four octal digits with the most significant numbers at the top. This may be used in conjunction with JAB/ to go from paper tape to punched cards. In general, this cannot be accomplished without unscrambling on a different machine.

BTY/ Described previously under typewriter. Allows comments to be typed directly onto printouts.

CLM/ Clear memory. Recommended for use prior to any routine.

DMP/ Memory dump. The state of the console should be noted before using. In particular, one usually wishes to record the program counter, accumulator, IO, program flags and any unusual light indicators such as the defer. The locations above 7000 octal are not obtained. Note that the memory dump does not destroy lower memory, so that it is often possible to take a dump, make a manual console change and continue running. The BTY mentioned above can often be alternated into the same upper section of memory in this manner.

ITP/ May be used to print BCD tapes on the IBM tape units.

LBC/ Loads binary cards. Note that certain upper locations cannot be loaded. Binary cards lacking data will often not load correctly, and should preferably be stripped from the binary deck. Those cards lacking any punches in col. 7 or beyond may be discarded without loss. In many cases, it is then necessary to enter the starting address (SA) in the address keys, hit stop and then start.

JAB/ Punched paper tape to magnetic tape. This routine is designed as a data pump with no attempt to unscramble the user's format, but rather to give him a magnetic tape that may be transferred to another machine. Variables are tape density and record length while words may either be packed or unpacked.

OTD/ Octal tape dump may be used for printing an odd parity binary tape written in any density.

PAL/, PAP/ Assembly routines previously discussed.

PAR/ Equations input from the typewriter are displayed. Values and variables may be modified.

PBC/ Punches binary cards from memory. A given program may be stopped in mid-run, the location of the stop placed in the test word, and a PBC executed. The resulting deck will restart the code from the stopped location provided that the code is entirely contained in locations 0 to 6246.

PM8/ Assembly routine producing paper tape binary input for the PDP-8.

VAP/ General purpose assembly system.

CRT Programming

Both the visual and precision CRT have 0,0 at the center. The visual CRT has 1024 points along each axis and the precision CRT has 4096 addressable points per axis. In both cases the word is packed in the left 12 (10) bits of the PDP word. Since bits to the right of this field are ignored by the deflection registers, and since commands for the two CRTs may be ORed together, it is easier to consider the system as if both CRTs were addressable to 4096 (10,000 octal) and program in this manner, ignoring the fact that the hardware does not register this significance on the visual CRT.

The most negative X to the left is 400000_8, and the largest positive X is 377700_8. Similarly the most negative Y is 400000_8 at the bottom of the screen and the largest value at the top is 377700_8. To continue incrementing on the largest number will simply wrap around the CRT and reenter at the left (bottom) side.

To step one position we add a delta of 100_8. Notice that in crossing 0, this causes a move from 777700 to 000001. This is not objectionable, since the display is identical to that for 000000. Under certain conditions, however, this could accumulate and introduce an error. To decrement a position, it is most convenient to subtract 77 (add 777700). This has the advantage that it moves from 0 to 777700 rather than to 777677 as would have been the case had we subtracted 100. Note that the two points in question would then display as 7777 and 7776, respectively, which are not identical.

In displaying a point, we place X in the accumulator and Y in the IO, and follow with an appropriate display command. We may either use the wait form, such as a DPHW DVHW, or else we may count on a minimum time in the program of 35 microseconds before returning to our display and use the display and proceed commands such as DPH DVH. Further timing considerations apply if we intend to read a position from the precision tube. Since our registers are set up in this manner, it follows that most scans should be programmed to step X and only at the end of each line to step Y. In this manner, we are able to minimize the number of changes in the IO.

A fast scan usually requires a delta of about 32_{10} to meet timing requirements and avoid flicker on the CRT. A non-flicker on the present visual system has been observed at a value of 27 frames/ sec., but this number can vary with the individual, the lights and the phase of the moon, so that a number of about 35 or even 40 frames per sec. is probably advisable for the present system.

Although we may physically display points as rapidly as 35 microseconds, in actual practice doing useful work, 85 microseconds is a fairly tight programming display loop. If we are reading values from the Eyeball, and in particular the analog digital converter, about double this figure is necessary.

A grid command is initiated at a given X,Y and then proceeds to the right (top) side of the screen. A useful variation on this rule is obtained by sending a display grid and proceed command (a DYVH command for example), and then interrupting before full drawing time is completed. This gives a shorter vector in the desired direction of slightly unpredictable length. When displaying points, such a vector may be used to call the user's attention to the point display, which may not be obvious. Grid display commands containing Y display a vertical line; those with X display a horizontal line. All grid display commands require an appreciably longer timing than normal display commands. The X and Y grid commands may be ORed to generate a 45° line.

A few CRT tricks may accelerate displays. Changing the sign of either X or Y reflects the point about the opposite axis. For example, X,Y changed to X,-Y is mirrored about the positive X axis. As pointed out previously, this is accomplished by complementing Y. Reversing the first bit changes the point by one quadrant (half-screen shift), and changing other bits cause quarter-screen shifts, eigth-screen shifts, and so forth. Thus, to shift a picture from the origin to the lower left may be accomplished by reversing the first bits of both X and Y assuming,

needless to say, a CRT format packed in the left 12 bits. Exchanging X and Y reflects the point about a 45° axis.

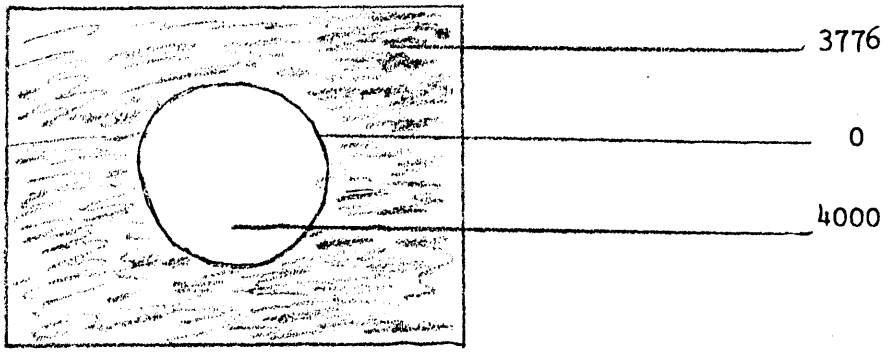
In a few cases such as a stepping scan, we perform our arithmetic with the point in the CRT format. In many cases, however, we require more extensive arithmetic operations. For this purpose, it is usually most convenient to reverse the first bit (an XOR operation against 400000) and then rotate right by six places masking out the unused six bits, if we have not already done so earlier. This converts our axial scale range from 400000,377700 to 000000, 007777, which can be manipulated with much greater ease. When we are ready to display, we simply reverse the process.

Translations of a point are readily accomplished by the methods above. Unfortunately, no convenient methods have been found for rotations, and at present the simplest approach is the brute force technique from analytic geometry -- caveat programmer.

In constructing program loops for either the yes-no Eyeball or the ADC to obtain density responses, much time could be lost in waiting for the hardware response. A display command requires 5 microseconds to execute and 50 microseconds to set the status register. From this time another 40 microseconds is required before the density is available from the low speed channel. These dead spaces should be utilized for useful work; the previous density signal may be converted during the settling time of the point, and the subsequent point coordinates may be constructed during the conversion time of the analog signal.

The manner in which the analog digital converter acts may be illustrated by the following table and illustration:

Film	Resulting signal	Voltage level
	400000	-10.5
Clear	400000	-10.0
	777600	-.009
grey	000000	0
	000200	+.009
dark	377600	+10.0
	377600	+10.5



The correspondence between the film and the signal must not be taken as absolute, since it is a relative matter that can vary from day to day and fluctuates somewhat. Relative densities are significant; absolute densities are not. Consequently, it is necessary to have programming to determine the proper ADC range in which the program shall run. A subroutine such as TOADX may be used for this purpose.

It is clear that to convert the density to an arithmetic number, we may use the same technique as applied to the CRT, namely, reverse the first bit except that now we rotate 7 places right instead of six, because the field is only 10 bits plus sign,

not 12. This results in a positive number in which 0 is clear film and 1777 is the darkest possible; in other words 1024 grey levels.

In passing, it might be mentioned, that the ADC formerly yielded a signal ranging from a few digits positive through 4000. This utilized only half of the full conversion range. It is still possible to set the level in this manner, if desired.