PDP-1 PROGRAM LIBRARY

NUMBER: Digital - 1 - 3 - S

NAME: DEC Debugging Tape

AUTHOR: A. Kotok - DEC (MIT-1)

DATE: Writeup revised August 13, 1964

SPECS: MS, MB - SA 6000

ABSTRACT: DDT facilitates online debugging.

DDT-1 PROGRAM DESCRIPTION

Users of most computers, especially large-scale ones, are familiar with the procedure of submitting a new program for a run and receiving, along with the compilation and assembly listings, a dump and perhaps a storage map of the symbols used, together with a few remarks about the failure of the program to run properly. If the user is lucky enough to be present when his program is processed, he may get additional information from the console lights, motion of tapes, etc., but his correcting must be done away from the machine. Getting a program to work under these conditions takes a long time.

DDT helps shorten this debugging time by allowing the user to work on his program at the computer, to control its operation, and to modify the program or its data at will. Tracking down a subtle error in a complex piece of coding is a laborious and frustrating job by hand, but with DDT's breakpoint facility, the user can interrupt his program at any point and examine the state of the machine. In this way, he can quickly locate sources of trouble.

The programmer may insert corrections and patches and try them out immediately; those that work can be punched out on the spot in the form of loadable patch tapes, eliminating the necessity of creating new symbolic tapes and reassembling each time an error is found and remedied. DDT also maintains a symbol table, allowing a programmer to discuss matters with the computer in the language of his own program.

DDT occupies registers 6000 to 7750. The starting address is 6000. DDT carries with it a permanent table, starting at 5777 and extending toward 0, consisting of all standard defined PDP-1 mnemonics. This may be augmented by symbol definitions from the user's program tape or from the keyboard.

DEFINITIONS

A symbol is a string of from one to three letters or numerals. A number is a string of up to 6 digits.

An expression is a string of symbols and numbers separated by the following characters:

space	a separation character meaning arithmetic + (plus)
+	a separation character meaning arithmetic + (plus)
-	a separation character meaning arithmetic - (minus)
∨	a separation character meaning boolean inclusive or.
∧	a separation character meaning boolean and

All other characters are either used for control or are illegal.

When a register is opened, its contents are printed out and become available for modificaction.

When a register is closed, any modifications requested are made and further access to the register is denied until it is opened again.

Most DDT operations are specified by a single letter in upper case. Typing such a symbol followed by a carriage return causes DDT to perform the operation.

In the following discussion, all numbers are octal integers. In the examples, "ldc", "tab", and "beg" are symbols in the user's hypothetical program. "c(r)" means "the contents of register R". All underlined expressions in the examples are those typed by DDT; expressions without underlining are those typed by the user. DDT responds to errors by typing a "?" and ignoring the error. The user may cancel a line at any time before the carriage return, by typing × (multiplication sign).

USING DDT

The user first reads DDT into the computer. All subsequent operations, including loading the program to be debugged, are performed through DDT on the typewriter. All printed output appears on the typewriter.

Loading the Program

Z — All of memory through the highest register not used by DDT (i.e., to the bottom of the symbol table) is set to zero.

fa<laZ — Zero memory between fa and la except that part, if any occupied by DDT.

K — DDT's symbol table is restored to its initial list of permanent symbols. If any of these have been modified, the original value is not restored.

Y — A binary tape in the reader is read into memory. No new symbols will be entered into DDT's table. The tape is read into storage between the limits in register M+1 and M+2. If a checksum error is encountered, the program will stop. It is then possible to move the tape back one block, restart the reader, and press Continue to continue reading.

T — Read a Macro Symbol Table and merge it with the existing table. Definitions on tape take precedence over definitions in storage. The special symbols 1s, 2s, up to 9s, are not entered. The lowest register occupied by the symbol table is typed out upon completing reading the symbol section of the tape. This number may be found in register F. Checksum errors are handled as in Y.

Punching Operations

DDT will punch a standard Macro format tape with the title in readable format.

L — Put DDT into the title punch listen mode. Characters typed in are punched out in readable format on paper tape. Terminating characters are tab, carriage return, or back-space, which do the following:

- **tab:** Sets DDT to punch read-in-mode data blocks.
- **carriage return:** punches a standard input routine and sets DDT to punch standard checksum data blocks.
- **back space:** sets DDT to punch standard checksum data blocks but punches no input routine.

fa<laD — Punches data blocks from the first address to the last address in the format specified by L above. The first address and the last address are any symbolic expressions where fa is less than or equal to la.

adrJ — Punches the start (jmp) block to the address specified to denote the end of the binary tape.

.(center dot) — When a register is open, make the modification, if any, and punch it in the format specified by L.

Example: Punch out registers 4-10 and 100-350, in standard checksummed blocks. The program's name is MICRO and it starts at 100.

L (carriage return)
MICRO
4<10D 100<350D 100J

Program Examination

These operations allow any register in memory to be examined and modified.

/ This is the register examination character. The expression typed immediately preceding the / is the address of the register to be opened.

Example 1 When the user types

ldc 20/ DDT will immediately move to the next tab stop, print out the contents of the register ldc 20, and skip to the following tab stop. The resulting line might look like this:

ldc 20/ add 2653

If the user wishes to change the contents of the register, he types in the new information:

ldc 20/ add 2653 add 2663

Carriage Return: This causes DDT to place the new information in the open register and to close it. If no modifications were typed before the CR, the register is closed unchanged.

/ (alternate use) The slash may be used to open a register addressed by the currently opened one. If in example 1 the user wished to examine the contents of location 2653, he would have typed a "/" instead of the modifying instruction. The results may be looked like this.

Example 2

ldc 20/ add 2653 / isp 3062

Here, isp 3062 is the contents of register 2653, which is now open. The previously opened register, ldc 20, has been closed.

> The "greater than" sign works like / except any modifications to the opened register will be made before the register addressed by the new contents of the closed register is opened.

Example 3

ldc 20/ add 2653 add 2663> 7044

The contents of register 2663 is 7044.
Use of ">" does not alter the sequence of locations.