IDENTIFICATION

Product Code: DEC-08-COCO-D

Product Name: ODT-8

Date Created: December 30, 1967

Maintainer: Software Service Group

1   ABSTRACT

ODT (Octal Debugging Technique) is a debugging aid for the PDP-8, which facilitates communication with, and alteration of, the program being run. Communication between operator and program occurs via the Teletype, using defined commands and octal numbers. This version of ODT has been completely revised and replaces both versions of the former ODT-II program.

2   PRELIMINARY REQUIREMENTS

2.1   Equipment

Standard PDP-8 or PDP-5 with basic 4k memory and Teletype.

2.2   Storage

ODT requires 600 (octal) consecutive core locations and one location on page 0 which will be used as an intercom register. It is page relocatable.

3   LOADING OR CALLING PROCEDURE

NOTE: ODT cannot be called as a subroutine.

a. ODT is normally distributed in binary with the source available on request and is loaded with the Binary Loader.

1. Place the ODT tape in the reader.

2. Set 7777 in the SWITCH REGISTER and press LOAD ADDRESS. (If using the high-speed photoelectric reader, put switch 0 down).

3. Press START.

b. Load the binary tape of the program to be debugged in the same manner as ODT was loaded. Be sure that the two do not overlap.

4   USING THE PROGRAM OR ROUTINE

4.1   Starting Procedure

a. The starting address of ODT is the address of the symbol START. For standard library versions the high version starts at 7000 and the low at 1000.

b. Set the starting address in the SWITCH REGISTER. Press LOAD ADDRESS, and START on the console. ODT will issue a carriage return and line feed to indicate that it is now running and awaiting commands from the keyboard.

c. To restart ODT wihout clearing the checksum, set the address of START + 1 (usually 7001 high version, or 1001 low version) into the SWITCH REGISTER and press LOAD ADDRESS and START on the console.

4.2   Control Characters

a. <u>Slash (/) - Open register preceding/</u>

The register examination character / causes the register addressed by the octal number

preceding the slash to be opened and its contents typed out in octal. The open register

can then be modified by typing the desired octal number and closing the register. Any

octal number from 1 to 4 digits in length is a legal input. Typing a fifth digit is an

error and will cause the entire modification to be ignored and a question mark to be typed

back by ODT. Typing (/) with no preceding argument causes the latest named register

to be opened (again). Typing 0/ is interpreted as / with no argument.

Example: 400/6046
           400/6046 2468?
           400/6046 12345?
           /6046

b. <u>Carriage Return ( )) - Close register</u>

If the user has typed a valid octal number, after the content of a register was printed by

ODT, typing) causes the binary value of that number to replace the original contents

of the opened register and the register to be closed. If nothing has been typed by the

user, the register is closed but the content of the register is not changed.

Example: 400/6046)   Register 400 is unchanged.
           400/6046 2345)   Register 400 is changed to contain 2345.
           /2345 6046)   Replace 6046 in register 400.

Typing another command will also close an opened register.

Example: 400/6046 401/6031 2346)   Register 400 is closed and unchanged and
           400/6046 401/2346)   401 is opened and changed to 2346.

c. <u>Line Feed (↓) - Close register, open next sequential register</u>

The line feed has the same effect as the carriage return, but, in addition, the next

sequential register is opened and its contents typed.

Example: 400/6046↓   Register 400 is closed unchanged and 401
           0401/6031 1234↓   is opened. User types change, 401 is
           0402/5201)   closed containing 1234 and 402 is opened.

d. <u>Up arrow (↑) - Close register, take contents as memory reference and open same</u>

Up arrow will close an open register just as will carriage return. Further, it will inter-

pret the contents of the register as a memory reference instruction, open the register

referenced and type its contents.

2

Example: 404/3270↑   3270 symbolically is "DCA, this page, relative
0470/0212 0000 ⟩   location 70," so ODT opens register 470.
404/3270↑
0470/0000

e. Back Arrow (◄—) - Close register, open indirectly.

Back arrow will also close the currently open register and then interrupt its contents as
the address of the register whose contents it is to type and open for modification.

Example: 365/5760↑
0360/0426 ◄—
0426/5201

f. Any Illegal Character

Any character that is neither a valid control character nor an octal digit, or is the fifth
octal digit in a series, causes the current line to be ignored and a question mark typed.

Example: 4:? ⟩   ⎫
4U? ⟩   ⎬ ODT opens no register.
406/4671 67K? ⟩   ⎭
/4671 ⟩   ODT ignores modification and closes register 406.

g. xxxxG - Transfer control to user at location xxxx.

Clear the AC then go to the location specified before the G. All indicators and regis-
ters will be initialized and the break-trap, if any, will be inserted. Typing G alone is
an error but will nevertheless cause a jump to location 0.

h. xxxxB - Set breakpoint at user location xxxx.

Conditions ODT to establish a breakpoint at the location specified before the B. If
B is typed alone, ODT removes any previously established breakpoint and restores the
original contents of the break location. A breakpoint may be changed to another loca-
tion, whenever ODT is in control, by simply typing xxxxB where xxxx is the new
location. Only one breakpoint may be in effect at one time; therefore, requesting a new
breakpoint removes any previously existing one. The previous restriction on placing a
breakpoint on a JMS followed by arguments has been removed as of the June 1967 revision.
This means ODT can now be more effectively used, especially in debugging programs
which utilize floating point. The only restriction in this regard is that a breakpoint may
not be set on any of the floating point instructions which appear as arguments of a JMS.

3

Example: TAD ⎫
            DCA ⎬   Breakpoint legal here.
            JMS ⎭
            FADD   Breakpoint illegal here.

The breakpoint (B) command does not make the actual exchange of ODT instruction for user instruction, it only sets up the mechanism for doing so. The actual exchange does not occur until a "go to" or a "proceed from breakpoint" command is executed.

When, during execution, the user's program encounters the location containing the breakpoint, control passes immediately to ODT (via location 0004). The C(AC) and C(L) at the point of interruption are saved in special registers accessible to ODT. The user instruction that the breakpoint was replacing is restored, before the address of the trap and the content of the AC are typed. The restored instruction has not been executed at this time. It will not be executed until the "proceed from breakpoint" command is given. Any user register, including those containing the stored AC and Link, can now be modified in the usual manner. The breakpoint can also be moved or removed at this time.

i. A - Open register containing AC.

When the breakpoint is encountered the C(AC) and C(L) are saved for later restoration. Typing A after having encountered a breakpoint, opens for modification the register in which the AC was saved and types its contents. This register may now be modified in the normal manner (see SLASH) and the modification will be restored to the AC when the "proceed from breakpoint" is given.

↓ after A - Open register containing Link

After opening the AC storage register, typing linefeed (↓) closes the AC storage register, then opens the Link storage register for modification and types its contents. The Link register may now be modified as usual (see SLASH) and that modification will be restored to the Link when the "proceed from breakpoint" is given.

j. C - Proceed (continue) from a breakpoint.

Typing C, after having encountered a breakpoint, causes ODT to insert the latest specified breakpoint (if any), restore the contents of the AC and Link, execute the instruction trapped by the previous breakpoint, and transfer control back to the user program at the appropriate location. The user program then runs until the breakpoint is again encountered.

4

NOTE: If a trap set by ODT is not encountered while ODT is running the object (user's) program, the instruction which causes the break to occur will not be removed from the user's program.

xxxC - Continue and iterate loop xxx times before break.

The programmer may wish to establish the breakpoint at some location within a loop of his program. Since loops often run to many iterations, some means must be available to prevent a break from occuring each time the break location is encountered. This is the function of xxxC (where xxx is an octal number). After having encountered the break-point for the first time, the user specifies, with this command, how many times the loop is to be iterated before another break is to occur. The break operations have been described previously is section h.

k. M - Open search mask.

Typing M causes ODT to open for modification the register containing the current value of the search mask and type its contents. Initially the mask is set to 7777. It may be changed by opening the mask register and typing the desired value after the value typed by ODT, then closing the register.

↓ - Open lower search limit

The register immediately following the mask storage register contains the location at which the search is to begin. Typing line feed (↓) to close the mask register causes this, the lower search limit register to be opened for modification and its contents typed. Initially the lower search limit is set to 0001. It may be changed by typing the desired lower limit after that typed by ODT, then closing the register.

↓ - Open upper search limit

The next sequential register contains the location with which the search is to terminate. Typing line feed (↓) to close the lower searchlimit register causes this; the upper search limit register to be opened for modification and its contents typed. Initially, the upper search limit is the beginning of ODT itself, 7000 (1000 for low version). It may also be changed by typing the desired upper search limit after the one typed by ODT, then closing the register with a carriage return.

1. xxxxW - Word search.

The command xxxxW (where xxxx is an octal number) will cause ODT to conduct a search of a defined section of core, using the mask and the lower and upper limits which the user has specified, as indicated in section k. Word searching using ODT is similar to word

searching using DDT. The searching operations are used to determine if a given quantity is present in any of the registers of a particular section of memory.

The search is conducted as follows: ODT masks the expression xxxx which the user types preceding the W and saves the result as the quantity for which it is searching. (All masking is done by performing a Boolean AND between the contents of the mask register, C(M), and the register containing the thing to be masked.) ODT then masks each register within the user's specified limits and compares the result to the quantity for which it is searching. If the two quantities are identical, the address and the actual unmasked contents of the matching register are typed and the search continues until the upper limit is reached.

A search never alters the contents of any registers.

Example: Search locations 3000 to 4000 for all ISZ instructions, regardless of what register they refer to (i.e. search for all registers beginning with an octal 2).

M7777	7000↓	Change the mask to 7000, open lower search limit
7453/0001	3000↓	Change the lower limit to 3000, open upper limit
7454/7000	4000↓	Change the upper limit to 4000, close register
2000W		Initiate the search for ISZ instructions

2000/2467	
3057/2501	These are 4 ISZ instructions in this section
3124/2032	of core.
4000/2152	

m. T - Punch leader

ODT is capable of producing leader (code 200) on-line. This is done by typing T and then turning ON the punch. When enough leader has been punched, turn off the punch and hit STOP on the console. It is imperative that the punch be turned OFF before typing again on the keyboard, since anything typed will be punched also, if the punch is left on. To issue any further commands, reload the starting address and press START on the console.

n. xxxx; yyyyP - Punch binary

To punch a binary core image of a particular section of core, the above command is used where xxxx is the initial (octal) address and yyyy is the final (octal) address of the section of core to be punched. The computer will halt (with 7402 displayed) to allow the user to turn ON the punch. Pressing CONTINUE on the console initiates the actual punching of

the block. The punching terminates without having punched a checksum, to allow subsequent blocks to be punched and to allow an all inclusive checksum to be punched at the end by a separate command. This procedure is optional, however, and the user may punch individually checksummed blocks.

It is imperative that the punch be turned OFF before typing another command, since the keyboard and punch are linked.

o. E - Punch checksum and trailer

Given the command E, ODT will halt to allow the punch to be turned on. Pressing CONTINUE on the console will cause it to punch the accumulated checksum for the preceding block(s) of binary output followed by trailer (code 200). When a sufficient length of trailer has been output, turn OFF the punch and press STOP on the console. To continue with ODT reload the starting address and press START on the console.

The binary tape produced in this manner by ODT can now be loaded into core and run. However, the changes should be made to the symbolic source tapes as soon as possible.

4.3 Additional Techniques

a. TTY I/O-Flag

Sometimes the program being debugged may require that the TTY flag be up before it can continue output, i.e., the program output routine will be coded as follows:

```
TSF
JMP .-1
TLS
```

Since ODT normally leaves the TTY flag in an off (lowered) state, the above coding will cause the program to loop at the JMP.-1. To avoid this, ODT may be modified to leave the TTY flag in the raised (on) state when transferring control through either a "go to" or a "continue" command. This modification is accomplished by changing location XCONT-3 (normally at 7341) to a NOP (7000). To make the actual change, load ODT as usual. Open register XCONT-3 and modify it as follows:

$$7341/\underline{6042}\ 7000\ \big) \qquad (1341/\underline{6042}\ 7000\ \big)\ \text{for low version})$$

b. Current Location

The address of the current register or last register examined is remembered by ODT and remains the same, even after the commands G, C, B, T, E, and P. This location may be opened for inspection merely by typing /.

7

c. Programs Written in ODT Commands

ODT will also correctly read tapes prepared off-line (e.g., a tape punched with 1021/1157↑7775 will cause location 1021 to be opened and changed to 1157; then the memory reference address 157 will be opened and changed to 7775 (-3). This procedure will work with breakpoints, continues, punch commands, etc. Thus, debugging programs may be read into ODT to execute the program, list registers of interest, modify locations, etc.

d. Binary Tape from High Speed Punch

It is possible to obtain a binary tape from the high speed punch, instead of the Teletype, however, this requires switch manipulation. Proceed as follows:

    1. Type the punch command xxxx; yyyyP as explained in section 4.2 (n). The computer will halt.

    2. Set 7231 (1231 for low version) in the SWITCH REGISTER (SR) and press LOAD ADDRESS.

    3. Set 6026 in the SR and press DEPOSIT.

    4. Set 6021 in the SR and press DEPOSIT.

    5. Set 7225 (1225 for low version) in the SR and press LOAD ADDRESS and START on the console, and leader (code 200) will be output.

    6. When a sufficient length of leader has been produced, press STOP on the console.

    7. Set 7203 (1203 for low version) in the SR and press LOAD ADDRESS and START on the console, and the section of core specified in the punch command will be output.

    8. If another block of data is desired on the same tape, the original contents of the locations changed in steps 3, 4 and 5 must be replaced. (See step 11.) Steps 1, 2, 3, 4, and 8 must then be repeated to output the data block via the high speed punch.

    9. Set 7222 (1222 for low version) in the SR and press LOAD ADDRESS and START on the console, and the accumulated checksum will be punched followed by trailer (code 200).

    10. When a sufficient amount of trailer has been produced, press STOP on the console and press the TAPE FEED button, then remove the tape from the punch.

11. To continue using ODT, the locations changed in steps 3 and 4 must be restored as follows:

> Set 7231 (1231 for low version) in the SR and press LOAD ADDRESS.
> Set 6046 in the SR and press DEPOSIT.
> Set 6041 in the SR and press DEPOSIT.

12. Set the starting address (7000 or 1000) in the SR and press LOAD ADDRESS and START on the console, and ODT is ready to go again.

e. <u>Interrupt Program Debugging</u>

ODT executes an IOF when a breakpoint is encountered. (It does not do this when more iterations remain in an x-continue command.) This is done so that an interrupt will not occur when ODT types out the breakpoint information. It thus protects itself against spurious interrupts and may be used safely in debugging programs that turn on the interrupt mode.

However, the user must remember that there is no way in which ODT could know whether the interrupt was on when the breakpoint was encountered, and hence it does not turn on the interrupt when transferring control back to the program after receiving a "go" or a "continue" command.

f. <u>Octal Dump</u>

By setting the search mask to zero and typing W, all locations between the search limits will be printed on the Teletype.

g. <u>Indirect References</u>

When an indirect memory reference instruction is encountered, the actual address may be opened by typing ↑ and ←.

4.4   <u>Errors</u>

The only legal inputs are control characters and octal digits. Any other character will cause the character or line to be ignored and a question mark to be typed out by ODT. Typing G alone is an error. It must be preceded by an address to which control will be transferred. This will elicit no question mark also if not preceded by an address, but will cause control be transferred to location 0.

Typing any punch command with the punch ON is an error and will cause ASCII characters to be punched on the binary tape. This means the tape cannot be loaded and run properly.

4.5   Miscellaneous

If a trap set by ODT is not encountered by the user's program, the breaktrap instruction will not be removed. ODT can now be used to debug programs using floating point, since the intercom register is now register 0004, and since breaktraps may now be set on a JMS with arguments following. This version of ODT will operate on a Teletype with an ALT mode key or an ESCAPE key. To restart ODT without clearing the checksum, set the SWITCH REGISTER to the value of start + 1 (7001 or 1001 in library versions) and press LOAD ADDRESS and START on the console. The high speed punch may be used by patching three locations after typing the punch command. (See section 4.3 d.)

5   DETAILS OF OPERATION AND STORAGE

5.1   Features

ODT features include register examination and modification; binary punchouts (to the Teletype or high speed punch) of user designated blocks of memory; octal core dumps to the Teletype using the word search mechanism, as in DDT; and instruction breakpoints to return control to ODT (breakpoints). ODT makes no use of the program interrupt facility and will not operate outside of the core memory bank in which it is residing.

The breakpoint is one of ODT's most useful features. When debugging a program, it is often desirable to allow the program to run normally up to a predetermined point, at which the programmer may examine and possibly modify the contents of the accumulator (AC), the Link (L), or various instruction or storage registers within his program, depending on the results he finds. To accomplish this, ODT acts as a monitor to the user program. The user decides how far he wishes the program to run and ODT inserts an instruction in the user's program which, when encountered, causes control to transfer back to ODT. ODT immediately preserves in designated storage registers, the contents of the AC and L at the break. It then prints out the location at which the break occurred, as well as the contents of the AC at that point. ODT will then allow examination and modification of any register of the user's program (or those registers storing the AC and L). The user may also move the breakpoint, and request that ODT continue running his program. This will cause ODT to restore the AC and L, execute the trapped instruction and continue in the user's program until the breakpoint is again encountered or the program terminated normally.

5.2   Storage

ODT requires 600 (octal) locations and, as distributed by the Program Library, resides in memory between 7000 and 7577 (or 1000 and 1577 for the low version). It is, however, page relocatable.

The source tape can be re-origined to the start of any memory page except page 0 and assembled to reside in the three pages following that location, assuming they are all in the same memory bank. ODT also uses location 4 on page 0 as an intercom register between itself and the user's program when executing a breaktrap. If the user wishes to change the location of the intercom register, he may do so by changing the value of ZPAT in the source and reassembling. The intercom register must remain on page 0.

6   RESTRICTIONS

      a. ODT will not operate outside of the memory bank in which it is located.

      b. It must begin at the start of a memory page (other than page 0) and must be completely contained in one memory bank.

      c. It will not turn on the program interrupt, since it has no way of knowing if the user's program is using the interrupt. It does, however, turn off the interrupt when a breakpoint is encountered, to prevent spurious interrupts. (See 4.3 (e).)

      d. The user's program must not use or reference any core locations occupied or used by ODT, and vice versa.

      e. Register ZPAT is used as an intercom register by ODT when executing a breakpoint. In library distributed versions ZPAT = 0004. This register must be left free by the user since it is filled with an address within ODT which is used to transfer control between user program and ODT.

      f. Breakpoints are fully invisible to "open register" commands; however, breakpoints may not be placed in locations which the user program will modify in the course of execution or the breakpoint will be destroyed.

7   REFERENCES

      a. See DDT Programming Manual (Digital-8-4-S) for a full explanation of the use of debugging programs.

      b. Binary Loader (Digital-8-2-U).

8   COMMAND SUMMARY

nnnn/	Open register designated by the octal number nnnn. Reopen latest opened register.
/	Reopen latest opened register.
Carriage Return ( ))	Close previously opened register.

Line Feed (↓)	Close register and open the next sequential one for modification.
Up Arrow (↑)	Close register, take contents of that register as a memory reference and open it.
Back Arrow (←)	Close register open indirectly.
Illegal character	Current line typed by user is ignored, ODT types "? CR LF".
nnnnG	Transfer program control to location nnnn.
nnnnB	Establish a breakpoint at location nnnn.
B	Remove the breakpoint.
A	Open for modification the register in which the contents of AC were stored when the breakpoint was encountered.
C	Proceed from a breakpoint.
nnnnC	Continue from a breakpoint and iterate past the breakpoint nnnn times before interrupting the user's program at the breakpoint location.
M	Open the search mask.
(line feed)	Open lower search limit.
(line feed)	Open upper search limit.
nnnnW	Search the portion of core as defined by the upper and lower limits for the octal value nnnn.
T	Punch leader.
nnnn;mmmmP	Punch a binary core image defined by the limits nnnn and mmmm.
E	Punch checksum and trailer.

9   <u>EXAMPLES AND/OR APPLICATIONS</u>

Symbols for representing "invisible" Teletype actions:

(CR)	=	Carriage Return
(LF)	=	Line Feed
(H)	=	Computer Halts
(Cont)	=	Key Continue on Console
(PON)	=	Punch On

12

(POF)   =   Punch Off

(LEAD)   =   Production of Leader

(BIN)   =   Punching of Binary Text

(CKSMT)   =   Punching of Checksum and Trailer

   The following examples are the actual result of using ODT to run the program listed after the examples. Brackets enclose comments local to the description. Underlinings designate that produced by ODT.

M7777 7000 (LF)(CR)   [mask modified]
7473 /0001 400 (LF)(CR)   [lower search limit modified]
7474 /7000 500 (CR)(LF)   [upper search limit modified]
3000W (CR)(LF)   [quantity for which to search specified and
0404 /3272 (CR)(LF)   search begun]
0431 /3277 (CR)(LF)
0437 /3277 (CR)(LF)
0444 /3300 (CR)(LF)
0450 /3277 (CR)(LF)
0453 /3300 (CR)(LF)
0455 /3276 (CR)(LF)
0455 /3277 (CR)(LF)
(LF)   [search completed]

M7000 7777 (LF)(CR)   [change mask]
7473 /0400 300 (LF)(CR)   [change lower limit]
7474 /0500 (CR)(LF)   [upper limit is all right]
7200W (CR)(LF)   [search for all CLA instructions]
0364 /7200 (CR)(LF)   [there is only one. It is at location 364]
(LF)   [search is finished]

M7777 600 (CR)(LF)   [set mask for indirect and page bits]
4000W (CR)(LF)   [using previous limits search for all references
0377 /7400 (CR)(LF)   to page zero which occur]
0411 /7450 (CR)(LF)
0414 /7450 (CR)(LF)
0417 /7450 (CR)(LF)
0432 /7400 (CR)(LF)   [there are none, however, these microinstructions
0440 /7400 (CR)(LF)   look like indirect references to page zero since
0451 /7400 (CR)(LF)   they have a 1 in bit 3 and a Ø in bit 4]
0462 /7540 (CR)(LF)
0455 /7400 (CR)(LF)
0472 /7521 (CR)(LF)
(LF)   [search completed]

M0600 0 (LF)(CR)   [set mask to zero so that everything will match]
7473 /0360 407(LF)(CR)   [set search limits to encompass dump area]
7474 /0500 427 (CR)(LF)
W (CR)(LF)   [since W is typed alone, the word searched for
0407 /1272(CR)(LF)   is 0. The result after masking each register
0410 /1272(CR)(LF)   with 0 is, of course, 0 so all comparisons
0411 /7450(CR)(LF)   appear to the program equal and hence all
0412 /5253(CR)(LF)   unmasked contents are typed, constituting
0413 /1273(CR)(LF)   a dump]
0414 /7450(CR)(LF)
0415 /5234(CR)(LF)
0416 /1273(CR)(LF)
0417 /7450(CR)(LF)
0420 /5227(CR)(LF)
0421 /7001(CR)(LF)
0422 /7650(CR)(LF)
0423 /5242(CR)(LF)
0424 /1274(CR)(LF)
0425 /4571(CR)(LF)
0426 /5203(CR)(LF)
0427 /1275(CR)(LF)
(LF)

Examples of Register Examination & Modification

400/6046 (CR)(LF)   [Examine Only]
400/6046 2463? (CR)(LF)   [Non-octal number typed, modification ignored]
400/6046 12345?(CR)(LF)   [More than 4 digits typed, modification ignored]
/6046 2345 (CR)(LF)   [Register 400 modified to 2345]
/2345 6046(CR)(LF)   [Modified again]
/6046 401/6031 2346 (CR)(LF)   [Register closed by typing another command]
400/6046 401/2346(CR)(LF)
/2346 6031(CR)(LF)
/6031

400/6046 (LF)(CR)   [close and examine next]
0401 /6031 1234 (LF)(CR)   [modify 401, examine 402]
0402 /5201 (CR)(LF)   [close 402]
401/1234 6031 (LF)(CR)
0402 /5201 (CR)(LF)
(LF)(CR)
0403 /6036 (CR)(LF)
(LF)(CR)
0404 /3270 (CR)(LF)

14

Examples of Register Examination & Modification (continued)

404/3270 ↑ (CR)(LF)   [contents of 404 refers to "this page, loc. 70"]
0470 /0212 0000 (CR)(LF)   [ODT opens 470. User modifies 470]
404/3270 ↑ (CR)(LF)
0470 /0000 (CR)(LF)
/0000 (CR)(LF)
404/3270 3271↑ (CR)(LF)   [contents of 404 modified to refer to "this page
0471 /0360 (C R)(LF)   [ODT opens 471]   loc. 71"]
404/3271 3270↑ (CR)(LF)
0470 /0000 (CR)(LF)

365/5160 ↑ (CR)(LF)   [contents of 365 refers to "this page, loc. 160"]
0360 /0426 ↑(CR)(LF)   [ODT opens 360. Contents of 360 become
0426 /5201 (CR)(LF)   [ODT opens 426]   address]

4: ?(CR)(LF)   ⎫
4U?(CR)(LF)   ⎬ illegal character. ODT opens no register
6Q?(CR)(LF)   ⎭
406/4671 Y? (CR)(LF)   ⎫ illegal character. ODT ignores modification
406/4671 67K? (CR)(LF)   ⎬ fifth digit in series. ODT ignores modification
406/4671 67322? (CR)(LF)   ⎭ register 406 still contains original value of 4671
/4671

Examples of setting Breakpoints and Executing User's Program

475/0000 1 (LF)(CR)   ⎧user's program expects to find the numbers
0476 /0000 2 (LF)(CR)   ⎨it is to use in 475 and 476 (see listing)
0477 /0000 (CR)(LF)   ⎩answer will be stored in 477
432B (CR)(LF)   [Breakpoint is set at location 432]
400G (CR)(LF)   [user's program begins at 400, go there]
+0432 00000 (CR)(LF)   [user's program accpts input of "+". Breakpoint
477/0003   [477 contains sum of 475 & 476] encountered
                                                                          ODT types break
Registers can be changed and the same breakpoint   address & C(AC)
remains in effect.

475/0001 3 (LF)(CR)
0476 /0002 (CR)(LF)
400G (CR)(LF)
*0432 00000(CR)(LF)
477/0006 (CR)(LF)

Examples of examining and modifying AC and L after encountering a breakpoint

A0200 1 (CR)(LF)   [AC which contained Ø when breakpoint was
A0001 (CR)(LF)   encountered is modified]
/0001 (CR)(LF)
(LF)(CR)
7356 /0001 Ø (CR)(LF)   [Link which contained 1 at break is modified
/0000 (CR)(LF)   to Ø]

446B (CR)(LF)   [Destroys old breakpoint & sets one at 446]
400G (CR)(LF)
*0446 (0004 (CR)(LF)   [Breakpoint encountered]
C (CR)(LF)   [continue until ...]
0446 (0010 (CR)(LF)   [Breakpoint again encountered]
CØ (CR)(LF)
0446 (0014 (CR)(LF)

476/0003 7
/0007
446B
400G
*0446 (0004   [Breakpoint encountered]
2C   [Continue as before but pass Breakpoint twice
0446 (0020   before stopping again]
C
0446 (0024

```
      /000        START=/000
      0004        ZPAT=4

                  /THIS IS A 3-PAGE, 4K,
                  /PAGEWISE-RELOCATABLE,
                  /OCTAL DEBUGGING SYSTEM CALLED
                  /***ODT-8***

      /000        *START

/000  3075                DCA I CKSAT    /CLEAR THE CHECKSUM,
/001  0010        P10,    10             /ARBITRARY CONSTANT

/002  4557        READ,   JMS CRLF       /END LINE; SET SHUT TO -1
/003  1073                TAD I INX      /TRAP
/004  3307                DCA WORD       /GET THE TRAP ADDRESS,
/005  1074                TAD I IND      /KEEP
/006  3707                DCA I WORD     /RESTORE CONTENT,
/007  3307        READ3,  DCA WORD       /CLEAR THE INPUT,//TH INST,
/010  1203                TAD FM5        /-5
/011  3374                DCA TUTE       /SET THE LETTER COUNT,
/012  0031        REA,    KSF
/013  5212                JMP .-1        /WAIT FOR COMMAND,
/014  0036                KRB
/015  3357                DCA SCHAR
/016  1357                TAD SCHAR      /GO TYPE THE CHARACTER,
/017  4712                JMS I INY
/020  1373                TAD RETN       /INITIALIZE THE PATCH
/021  3004                DCA ZPAT       /EVERY TIME,
/022  1243                TAD BLIST      /COMPUTE ADDRESS OF COMMAND,
/023  3323                DCA SPNTR
/024  1723                TAD I SPNTR    /SEARCH FOR LEGAL CHARACTER,
/025  2323                ISZ SPNTR
/026  7510        FM2/0,  SPA            /TEST FOR END OF LIST; MINUS 5
/027  5277        QUEST,  JMP SEX        /NOT SATISFIED,
/030  7041                CIA            /COMPARE THE CHARACTER,
/031  1357                TAD SCHAR
/032  7640        FP240,  SZA CLA        /FOUND
/033  5224                JMP .-7        /NO, CONTINUE
/034  1323                TAD SPNTR
/035  1242                TAD LIABL
/036  3323                DCA SPNTR
/037  1723                TAD I SPNTR    /LOOK UP THE ADDRESS,
/040  3323                DCA SPNTR
/041  5723                JMP I SPNTR    /GO PROCESS,

/042  0514        LIABL,  TABL2-TABL1-1
/043  1044        BLIST,  TABL1

                  /ODT-8 WILL ALSO CORRECTLY READ SYMBOLIC
                  /TAPES PREPARED FOR IT: E.G. 1021/115/*77/5
```

/COMMAND LIST

```
        1044            TABLE:.
1044   0320                    320             /PUNCH
1045   0305                    305             /END
1046   0324                    324             /TRAILER
1047   0212            LP,     212             /OPEN NEXT
1050   0215            CR,     215             /CLOSE THIS ONE
1051   0237            SLA,    237             /OPEN THIS ONE
1052   0302                    302             /BREAK
1053   0307                    307             /GO
1054   0273                    273             /;
1055   0303                    303             /CONTINUE
1056   0327                    327             /WORD SEARCH
1057   0336                    336             /UP-ARROW OPENS INDIRECT(I.E. MEM REF)
1060   0315                    315             /MASK*UPPER*LOWER*
1061   0301                    301             /AC*LINK
1062   0337                    337             /BACK ARROW = OPEN INDIRECTLY
                        /TABLE MUST END WITH A NEG NUMBER
1063   1113            FMS,    -3

1064   1307            EXAM,   TAD WORD        /LOAD ADDRESS
1065   7440                    SZA             /IF ZERO, USE LAST
1066   3310                    DCA CAD
1067   1710            EX2,    TAD I CAD
1070   4711                    JMS I INO       /PNUM   (PRINT CONTENTS)
1071   3315                    DCA SHUT        /SIGNALS OPEN REG
1072   5207                    JMP READS

1073   1337            INX,    TRAD
1074   1360            INO,    KEEP
1075   7303            CRSA,   CRSA
1076   1302            INT,    FROG


        /PROCESS OCTAL DIGITS:

1077   1200            SEX,    CLA
1100   1337                    TAD SCHAR
1101   1226                    TAD FM2/0       /(=0)
1102   1500            CRNUM,  SMA
1103   5317                    JMP NU          /ILLEGAL CHAR
1104   1201                    TAD P10         /10
1105   1510                    SPA
1106   5317                    JMP NU          /ILLEGAL CHAR
1107   3323                    DCA SAD
1110   1307                    TAD WORD        /ASSEMBLE AN ADDRESS
1111   7104                    RAL CLL
1112   7006                    RTL
1113   1323                    TAD SAD
1114   3307                    DCA WORD
1115   2314                    ISZ TOTE
1116   5212                    JMP REA
```

```
                              /TYPE ERROR INDICATOR (?)
/117   /200            NO,     CLA
/120   1221                    TAD QUEST          /2//
/121   4//2                    JMS I INY          /TYPN
/122   5202                    JMP READ


                      /TO OPEN LOCATION ZERO,
                      /OPEN //// AND TYPE LINEFEED.

                      /THE ADDRESS OF THE LAST REGISTER
                      /EXAMINED REMAINS THE SAME AND MAY BE OPENED BY "/"


       /123           SPNTR=,
       /123           SAD=,

                      /ROUTINE TO HANDLE REG, MODIFICATION AND INCREMENTAL EXAMINE
/123   0000           CRL,    0
/124   1374                    TAD TUTE
/125   /041                    CIA
/126   1203                    TAD FMS            /-3
/127   /050                    SNA CLA
/130   5723                    JMP I CRL          /NO MOD, INFO AVAILABLE
/131   1367                    TAD WORD
/132   2375                    ISZ SHUT           /TEST FOR OPEN AND THEN CLOSE IT,
/133   3//0                    DCA I CAD          /MODIFY REGISTER
/134   5723                    JMP I CRL

/135   4323           CRL1,   JMS CRL            /CARRIAGE RETURN TO CLOSE
/136   4357                    JMS CRLF
/137   5207                    JMP READS

/140   1250           CRL2,   TAD CR             /SINGLE FEED+CR
/141   4//2                    JMS I INY
/142   4323                    JMS CRL
/143   4//2                    JMS I INY          /TIME FOR CAR TO RET.
/144   2370                    ISZ CAD            /LINE FEED = EXAMINE NEXT
/145   1370           UPAR3,  TAD CAD
/146   4//1                    JMS I INB          /PNUM
/147   1251                    TAD SLA
/150   4//2                    JMS I INY          /TYPN
/151   5267                    JMP EX2

/152   4323           UPIN,   JMS CRL            /CLOSE FIRST
/153   1/70                    TAD I CAD
/154   3370                    DCA CAD
/155   4357           UPAR2,  JMS CRLF
/156   5345                    JMP UPAR3
```

```
/157        SCHAR=,

            /TYPE A CAR. RET, AND LINE FEED
/157  0000  CRLF,   0
/100  1256          TAD CR          /212
/101  4112          JMS I INY       /TYPN
/102  1247          TAD LF          /212
/103  4112          JMS I INY       /TYPN
/104  7040          CMA             /MINUS ONE
/105  3315          DCA SHUT        /SIGNALS CLOSED REGISTER
/106  5757          JMP I CRLF

            /PAGE ONE PARAMETERS,
/107  0000  WORD,   0
/110  0000  CAD,    0               /CURRENT ADDRESS
/111  7446  INB,    PNUM
/112  1256  INY,    TYPN
/113  1245  REIN,   BURP
/114  0000  IOIE,   0
/115  7777  SHUT,   7777

/116  1367  PUNC,   TAD WORD
/117  3676          DCA I INY
```

```
                          /001=0, SECOND CORE PAGE

        /200              *START+200


/200  0177              SP177,  177               /FIRST IN THIS PAGE
/201  5767                      JMP I IN13         /READ3

                        /PUNCH DATA,
/202  7002              PUN1,   CLA HLT
/203  1302                      TAD FROG
/204  4765                      JMS I IN11         /PUNN (PUNCH ORIGIN)
/205  0100                      100
/206  1702              PUN2,   TAD I FROG
/207  4765                      JMS I IN11         /PUNN (PUNCH CONTENTS)
/210  0000                      0
/211  1302                      TAD FROG
/212  7041                      CIA
/213  1704                      TAD I IN10         /WORD
/214  7650                      SNA CLA
/215  5767                      JMP I IN13         /READ3
/216  2302                      ISZ FROG
/217  5206                      JMP PUN2
/220  5767                      JMP I IN13


                        /PUNCH END,
/221  7002              PUN3,   CLA HLT
/222  1303                      TAD CKSA
/223  4765                      JMS I IN11         /PUNN (PUNCH CHECKSUM)
/224  0000                      0
                        /PUNCH LEADER,
/225  1271              PUN4,   TAD SP200
/226  4230                      JMS TYPN
/227  5225                      JMP .-2


                        /TO USE THE HIGH SPEED PUNCH,
                        /TYPE "XX;YYP" THEN TOGGLE IN
                        /THE PATCHES INDICATED BELOW,
                        /THEN LOAD ADDRESS AND START:
                        /PUN4 - FOR LEADER-TRAILER,
                        /PUN1+1 = FOR DATA
                        /PUN3+1 = FOR CHECKSUM AND LEADER,
                        /RESTORE PATCHES BEFORE RESTARTING,
                        /RESTART AT START TO CLEAR CHECKSUM,
                        /RESTART AT START+1 TO RETAIN CHECKSUM,


                        /TYPE A CHARACTER
/230  0000              TYPN,   0
/231  6046                      TLS       /(0020) - FOR H,S;
/232  6041                      TSF       /(0021) - FOR H,S;
/233  5232                      JMP .-1
/234  7000              SP7000, 7000      /CLA-GROUP2
/235  5030                      JMP I TYPN
```

```
/FEATURES ADDED: INTERUPT TURNED OFF UPON HITTING BREAKPOINT; CAN USE
/HI SPEED PUNCH. BREAKPOINT CAN BE PUT ON A JMS FOLLOWED BY ARGUMENTS.
/ODT-8 IS RELOCATABLE; IF BREAKPOINT PUT ON INSTR REFERENCING AUTO-INDEX
/INDIRECTLY, IT WILL BE INCREMENTED ON CONTINUE; LINK & AC EXAMINE ON
/COMMAND; / OPENS LATEST OPENED REGISTER; CLARITY; AUTO LEADER/TRAILER;
/OPEN MEM. REF.(*); AND OPEN INDIRECT (BACK ARROW); ALSO XXX C.

                /SET A BREAK POINT.
1236   1104     TRAP,    TAD I IN10      /(WORD)=ADDRESS OF TRAP.
1237   7450              SNA
1240   1366              TAD IN12        /CRLF
1241   3357              DCA TRAD        /TRAP SET (REAL OR DUMMY)
1242   5320              JMP SPEXIT      /GO TO SECOND PAGE EXIT.

                /THE TRAP IS SPRUNG

1243   3355     BURP,    DCA SAC         /SAVE C(AC)
1244   7004              RAL
1245   3356              DCA LINK        /SAVE C(L)

1246   1360              TAD KEEP
1247   3757              DCA I TRAD      /REPLACE INSTRUCTION WHICH WAS TRAPPED

1250   7101              IAC CLL
1251   1357              TAD TRAD
1252   3361              DCA GAME        /SAVE CONTINUATION ADDRESS (BREAK ADDR+1)

1253   1360              TAD KEEP        /PICK UP TRAPPED INSTRUCTION
1254   1372              TAD SP2000      /OVERFLOW TO LINK IF IOT OR OPERATE INSTR.
1255   0271              AND SP200       /AC=0 IF PAGE 0 REFERENCE
1256   7600              SZA SNL CLA     /WAS TRAPPED INSTR AN IOT,OPER,PAGE 0 REFERENCE?
1257   5265              JMP CURPAG      /NO
1260   4322              JMS TSTJMS      /YES, SEE IF IT WAS A JMS
1261   7650              SNA CLA
1262   5267              JMP CURPAG+2    /YES, TREAT AS IF NON-PAGE-ZERO REFERENCE
1263   1360              TAD KEEP        /NO, PUT ACTUAL INSTR IN "THE" FOR EXECUTION
1264   5306              JMP LIP4

1265   1357     CURPAG,  TAD TRAD
1266   0234              AND SP7000
1267   3362              DCA FROG        /SAVE INITIAL ADDR OF PAGE REFERENCED BY TRAPPED INSTR.

1270   1360              TAD KEEP
1271   0200     SP200,   AND SP177       /GET RELATIVE ADDR REFERENCED BY TRAPPED INSTR.
1272   1362              TAD FROG        /ADD ON TOP OF PAGE
1273   3362              DCA FROG        /SAVE ABSOLUTE ADDRESS OF MEMORY REFERENCE

1274   1360              TAD KEEP
1275   0373              AND SP400
1276   7650     LPAR,    SNA CLA         /IS IT AN INDIRECT REFERENCE?
1277   5302              JMP LIP         /NO
1300   1762              TAD I FROG      /YES, GET ACTUAL REFERENCE
1301   3362              DCA FROG
```

```
/302  4322    LIP,    JMS TSTJMS    /SEE IF TRAPPED INSTR IS A JMS
/303  7450            SNA
/304  4771            JMS I IN21    /YES, IT IS A JMS (JMSER)
/305  1377            TAD IFROG     /NO    (JMS I FROG) JMS ADDS BACK 4000
/306  3351    LIP4,   DCA THE       /STORE FOR EXECUTION
/307  2765            ISZ I IN11    /TEST N-CONTINUE
/310  5344            JMP XCONT     /IGNORE THIS BREAK

/311  6002            IOF           /STOP INTERRUPTS

/312  1357            TAD TRAD      /PNUM   (PRINT TRAP ADDRESS)
/313  4770            JMS I IN14    /LEFT PAREN    (8 BITS=250=ASCII LFT PAREN)
/314  1276            TAD LPAR
/315  4230            JMS TYPN
/316  1355            TAD SAC
/317  4770            JMS I IN14    /PNUM   (PRINT C(AC))

/320  4766    SPEXIT, JMS I IN12    /CRLF
/321  5767            JMP I IN13    /READS

/322  0000    TSTJMS, 0
/323  1300            TAD KEEP      /GET TRAPPED INSTR,
/324  0374            AND SP7000    /ISOLATE OP CODE
/325  1375            TAD SP4000    /OVERFLOW TO LINK WITH AC=0 IF JMS (4000)
/326  5722            JMP I TSTJMS

              /START AT A LOCATION
/327  1764    JUMP,   TAD I IN10    /(WORD)
/330  3361            DCA GAME
/331  1352            TAD JPIGAM    /(JMP I GAME)
/332  3351            DCA THE
/333  3355            DCA SAC       /CLEAR THE AC,
/334  7410            SKP
/335  1764    CONTIN, TAD I IN10    /(WORD)
/336  7040            CMA
/337  3765            DCA I IN11    /(PUNN)=TMP COUNTER:
/340  4766            JMS I IN12    /(CRLF)
              /PATCH THE NEXT LOCATION WITH NOP(/000)
              /IF THE PROGRAM BEING DEBUGGED EXPECTS
              /THE TTY FLAG TO BE UP:
/341  6042            TCF           /CLEAR THE FLAG
/342  1757            TAD I TRAD    /SAVE TRAP CONTENTS,
/343  3300            DCA KEEP

/344  1356    XCONT,  TAD BAIT
/345  3757            DCA I TRAD    /INSERT TRAP INSTRUCTION
/346  1556            TAD LINK
/347  7110            RAR CLL       /RESTORE LINK
/350  1355            TAD SAC       /AND C(AC)
/351  7402    THE,    HLT           /OUT EXECUTION OF TRAPPED INST, AFTER PROCEED
/352  5761    JPIGAM, JMP I GAME
/353  2361            ISZ GAME      /IMITATE SKIP CONDITION,
/354  5352            JMP .-2
```

/VARIABLES MAY BE SCANNED VIA "A",

```
/355   0000        SAC,    0         /AC
/356   0000        LINK,   0         /LINK OIT
/357   /157        TRAD,   CRLF      /ADDRESS OF TRAP,
/360   0000        KEEP,   0         /CONTENT OF TRAP
/361   0000        GAME,   0         /ADDRESS FOR CONTINUE
/362   0177        FRUG,   START-1   /MEMORY REFERENCE,
/363   0000        CKSA,   0         /THE CHECKSUM TO DATE,

                   /INTER COM REGS,
/364   /167        IN10,   WORD
/365   /401        IN11,   PUNN
/366   /157        IN12,   CRLF
/367   /007        IN13,   READS
/370   /446        IN14,   PNUM
/371   7475        IN21,   JMSER              /PROCESS JMS,

                   /CONSTANTS
/372   2000        SP2000, 2000
/373   0400        SP400,  400
/374   7000        SP7000, 7000
/375   4000        SP4000, 4000
/376   5404        BAIT,   JMP I ZPAT
/377   4702        IFRUG,  JMS I FRUG
```

/001-0, THIRD CORE PAGE,

```
        1400                    *START+400


                        /PUNCH ROUTINE
1400    0177            TP177,  177       /FIRST IN THIS PAGE,

1401    0000            PUNN,   0
1402    3240                    DCA PNUM
1403    1240                    TAD PNUM
1404    7012                    RTR
1405    7012                    RTR
1406    7012                    RTR
1407    0354                    AND TP77
1410    1001                    TAD I PUNN
1411    4236                    JMS CKSM
1412    1240                    TAD PNUM
1413    0354                    AND TP77
1414    4236                    JMS CKSM
1415    5001                    JMP I PUNN
                        /MEMORY REFERENCE OPENER,
1416    4742            UPAR1,  JMS I INS0     /(CRL)="CLOSER CALL",
1417    1741                    TAD I IN27     /DAD
1420    3236                    DCA TEM
1421    1036                    TAD I TEM
1422    0200            TP200,  AND TP177
1423    3201                    DCA TEM2       /SAVE LOWER BITS,
1424    1036                    TAD I TEM
1425    0222                    AND TP200
1426    7050                    SNA CLA        /TEST FOR PAGE ZERO REF
1427    5232                    JMP .+3        /YES
1430    1741                    TAD I IN27
1431    0266                    AND TP7000
1432    1201                    TAD TEM2
1433    3741                    DCA I IN27     /DAD
1434    5035                    JMP I .+1
1435    1755                    UPAR2


                        /CHECK SUM ACCUMULATOR
1436    0000            CKSM,   0
1437    3275                    DCA CKT
1440    1746                    TAD I IN20     /CKSA
1441    1275                    TAD CKT
1442    3746                    DCA I IN20     /CKSA
1443    1275                    TAD CKT
1444    4745                    JMS I IN19     /TYPN
1445    5036                    JMP I CKSM
```

```
                              /ROUTINE TO PRINT OCTAL CONTENTS OF AC
1446   0000        PNUM,      0
1447   3201                   DCA PUNN
1450   1352                   TAD TM4
1451   3236                   DCA CKSM
1452   1201                   TAD PUNN
1453   7004                   RAL
1454   7004        PN2,       RAL
1455   7006                   RTL
1456   3201                   DCA PUNN
1457   1201                   TAD PUNN
1460   0351                   AND TP007    /ONLY 7=DIGITS GUARANTEED,
1461   1355                   TAD TP60     /IN CASE BIT 8 CAME THROUGH,
1462   4745                   JMS I INIY   /TYPN
1463   1201                   TAD PUNN
1464   2236                   ISZ CKSM
1465   5254                   JMP PN2
1466   7000        TP7000,    7000         /CLA=GROUP2
1467   1351                   TAD TP240
1470   4745                   JMS I INIY
1471   5646                   JMP I PNUM


                              /SEARCH VARIABLES,
1472   7777        MASK,      7777
1473   0001        LIMLO,     0001
1474   1000        LIMHI,     START

       1475        CRT=,
1475   0000        JMSER,     0
1476   1747                   TAD I IN22   /(PROG)=ABS MEM REF, (FINAL)
1477   3246                   DCA PNUM
1500   1750                   TAD I IN23   /GAME
1501   3646                   DCA I PNUM   /SIMULATED JMS
1502   2747                   ISZ I IN22   /PROG
1503   1353                   TAD TP1000
1504   5675                   JMP I JMSER
```

```
                              /WORD SEARCH ROUTINE
/505   4/45            WSER,   JMS I IN10        /CRLF
/506   12/3                   TAD LIMLO
/507   32/5                   DCA CKT
/510   10/5            WSER1,  TAD I CKT
/511   02/2                   AND MASK
/512   /041                   CIA
/513   1/44                   TAD I IN17        /WORD
/514   /040                   SZA CLA
/515   5525                   JMP WSER2
/516   12/5                   TAD CKT
/517   4240                   JMS PNUM
/520   135/                   TAD TP25/              /(SLASH)
/521   4/45                   JMS I IN19        /TYPN
/522   10/5                   TAD I CKT
/523   4240                   JMS PNUM
/524   4/45                   JMS I IN10        /CRLF
/525   12/5            WSER2,  TAD CKT
/526   22/5                   ISZ CKT
/527   /041                   CIA
/530   12/4                   TAD LIMHI
/531   /040            TP240,  SZA CLA
/532   5510                   JMP WSER1
/533   4/45                   JMS I IN10        /CRLF
/534   5/51                   JMP I IN25        /READ+5

                              /ROUTINES TO TYPE MASK AND LIMITS
/535   1550            AUX,    TAD CONSAU
/536   1500            MASKER, TAD CONSMS
/537   3/44                   DCA I IN17        /WORD
/540   5/00                   JMP I IN20        /EXAM
```

```
        /401            IEM2=PUNN
        /430            IEM=CRSM

/541    /170            IN27,CAD
/542    /123            IN30,CRL

                        /INTER COM REG
/543    /157            IN10,   CRLF
/544    /167            IN17,   WORD
/545    /230            IN19,   TYPN
/546    /303            IN20,   CRSA
/547    /302            IN22,   FROG
/550    /301            IN23,   GAME
        /551            TPOO/=,
/551    /007            IN25,   READ+5

                        /CONSTANTS
/552    /114            IM4,    =4
/553    1000            IP1000, 1000
/554    0077            IP77,   77
/555    0000            IPOO,   60
/556    /003            CONSAC, SAC=MASK
/557    0257            TP257,  257
/560    /412            CONSMS, MASK

        /561            TABLE=,
/561    /202                PUN1
/562    /221                PUN3
/563    /225                PUN4
/564    /140                CRL2
/565    /135                CRL1
/566    /004            IN20,   EXAM
/567    /236                TRAP
/570    /327                JUMP
/571    /170                PUNC
/572    /355                CONTIN
/573    /205                WSER
/574    /416                UPAR1
/575    /356                MASKER
/576    /355                AUX
/577    /152                UPIN    /OPEN INDIRECTLY,
```

$

THERE ARE NO ERRORS

SYMBOL TABLE

```
AUX        7355
BAIT       7370
BLIST      7043
BURP       7243
CAD        7170
CRNUM      7102
CRSA       7303
CRSAI      7073
CRSM       7430
CRT        7473
CONTIN     7333
CONSAC     7330
CONSMS     7300
CR         7030
CRL        7123
CRLF       7131
CRL1       7133
CRL2       7140
CURPAG     7263
EXAM       7004
EX2        7001
FM20       7026
FMS        7003
FP240      7032
FROG       7302
GAME       7301
IFROG      7377
INX        7073
IN0        7074
IN10       7304
IN11       7305
IN12       7306
IN13       7307
IN14       7370
IN16       7343
IN17       7344
IN19       7345
IN20       7346
IN21       7371
IN22       7347
IN23       7350
IN25       7351
IN26       7300
IN27       7341
IN30       7342
IN7        7070
IN8        7171
IN9        7172
JMSER      7475
JPIGAM     7332
JUMP       7327
KEEP       7300
LR         7047
```

SYMBOL TABLE

```
LIMHI    7474
LIMLO    7473
LINK     7550
LIP      7502
LIP4     7500
LPAK     7276
LIABL    1042
MASK     7472
MASKER   1550
NU       1117
OPIN     7152
PNUM     7446
PN2      1454
PUNC     7175
PUNN     1401
PUN1     1202
PUN2     1200
PUN3     1221
PUN4     1223
P10      7001
QUEST    7527
REA      7012
READ     7002
READS    1001
RETN     7173
SAC      7555
SAD      1123
SCHAR    1157
SEX      7077
SHUT     7175
SLA      1051
SPEXIT   7520
SPNTR    1123
SP177    7200
SP200    7271
SP2000   7512
SP400    7513
SP4000   7515
SP7000   7514
SP7000   1234
START    7000
TABL1    7044
TABL2    1551
TEM      7450
TEM2     7401
THE      1551
TM4      1552
TUTE     1174
TP007    1551
TP1000   1555
TP177    1400
TP200    1422
TP240    1551
```

SYMBOL TABLE

TP25/	7551
TP60	7555
TP7000	7400
TP7/	7554
TRAD	7551
TRAP	7230
TSTJMS	7522
TYPN	7230
UPAR1	7410
UPAR2	7155
UPAR3	7145
WORD	7107
WSER	7505
WSER1	7510
WSER2	7525
XCUNT	7544
ZPAT	0004