

DZ11

8 LINE ASYNCH MUX TESTS MD-11-DZDZA-C

EP-DZDZA-C-DL-B
COPYRIGHT © 1976
FICHE 1 OF 1

DEC 1976
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDZA-C-D
PRODUCT NAME: DZ11: 8 LINE ASYNC MUX TESTS
DATE RELEASED: DEC 1976
MAINTAINER: DIAGNOSTICS
AUTHORS: JOHN EGOLF
 JERRYL PAYNE

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OF RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

The function of the DZ11 diagnostics is to verify the option operates according to specifications. The diagnostics also verify that the DZ11 operates in its environment such as the system in which it is installed.

Parameters may be supplied to the program by either 'AUTO SIZING' or input from the user on the console by having SW00=1 at start time. Auto sizing will be done only the first time the program is started and SW07=0 and SW00=0 and SW03=0. Console input may be done at any start time if SW00=1.

Currently there is one standalone diagnostic (DZDZA), one system module for DEC X/11 (DZAA), and there are plans for an online overlay for DZITA (ITEP) - DZDZB.

DZDZA will test all parts of the DZ11 such as cables, dist pnl., and the interface module itself.

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (WITH MINIMUM 8K MEMORY)

ASR 33 (or equivalent for console)

DZ11 INTERFACE MODULE (M7B19(EIA), M7B14(20MA))

H327 Staggered turnaround connector. (if †B PARITY

and BREAK are to be tested.

H325 Cable turnaround and dist pnl testing.

H326 This may be substituted for H325.

2.2 STORAGE

Program will use all BK of memory except where ABL and BOOTSTRAP LOADER reside. Locations 1500 thru 2000 are especially to be noted and to be untouched by operator after parameters have been input from console (SW00=1); or after the 'AUTO SIZING' has been done. These locations may be changed if the user understands their meaning and different parameters are required.

3. LOADING PROCEEDURE

3.1 METHOD

All programs are in absolute format and are loaded using the ABSOLUTE LOADER. NOTE: if the diagnostics are on a media such as DISK, MAGTAPE, DECTAPE, or CASSETTE; follow instructions for the monitor which has been provided on that specific media.

ABSOLUTE LOADER starting address *500

MEMORY * SIZE

4k	17
8k	37
12k	57
16k	77
20k	117
24k	137
28k	157

- 3.1.1 Place address of ABS loader into switch register.
(also place 'HALT' SW up)
- 3.1.2 Depress 'LOAD ADDRESS' key on console and release.
- 3.1.3 Depress 'START KEY' on console and release (program should now be loading into CPU)

4. STARTING PROCEEDURE

- A. Set switch register to 000200
- B. Depress 'LOAD ADDRESS' key and release
- C. Set SWR to zero for 'AUTO SIZING' or leave or set SW00=1 for user input from console terminal.
- D. Depress 'START KEY' and release, the program will type Maindec Name and program name (if this was the first start up of the program or parameters were changed by SW00=1) and also the following:

```

'MAP OF DZ11 STATUS'
1500 160010
1502 000300
1504 000005
1506 000377
1510 017470
1512 000000

```

The above is only an example! This would indicate the status table starting at add. 1500 in the program. †THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. For information of status table see section B.4 for help.

The program will type "Running" and proceed to run the diagnostic.

4.1 CONTROL SWITCH SETTINGS

NCTE: If there is no real SWR (177570); SWR may be modified at Loc:176 or by hitting Control "G" (†G) on console terminal.

```

SW 15 Set: Halt on error
SW 14 Set: Loop on current test
SW 13 Set: Inhibit error print out
SW 12 Set: Inhibit **ALL** type out/bell on error.
SW 11 Set: Inhibit iterations. (quick pass)
SW 10 Set: Escape to next test
SW 09 Set: Loop with current data
SW 08 Set: Catch error and loop on it
SW 07 Set: NO AUTO SIZE; CLR-do AUTO SIZE. If 1st start of program
      after loading.
SW 06 Set: Reselect DZ11's desired active
SW 05 Set: Reserved
SW 04 Set: SELECT DELAY PARAMETER
SW 03 Set: Extra parameter input
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Get users parameters from console

```

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 06 RESELECT DZ11'S DESIRED ACTIVE. please note that a message is typed out for setting the switch register equal to DZ11's active. this means if the system has four DZ11s; bits 00,01,02,03 will be set in loc 'DZACTV' from the switch register. Using this switch(SW06) alters that location;therefore if four DZ11s are in the system ***DO NOT*** set switches greater than SW 03 in the up position. This would be a fatal error. do not select more active DZ11s than has been given information about in parameter input (SW00=1)

METHOD: A: Load address 200
 B: Start with SW 06=1
 C: Program will type message
 D: Set the BINARY number of DZ11s desired active EXAMPLE: 1=1 DZ11; 3=2 DZ11; 7=3 DZ11; 17=4 DZ11 37=5 DZ11 etc/aa PRESS CONTINUE.
 E: Number (IF VALID) will be in data lights (excluding 11/05)
 F: Set with any other switch settings desired. PRESS CONTINUE.

SW 01 RESTART PROGRAM AT SELECTED TEST it is strongly suggested that at least one pass has been made before trying to select a test that is not in the order of sequence the reason being is that the program has to clear areas and set up parameters. Note: if running multiple DZ11's; the DZ11 you desire to be under test must be selected by the use of SW06 before locking on the test. In other words; each time the program is started; the first DZ11 will be selected to be under test unless SW06 is used to select only one.

SW 09 LOOP ON CURRENT DATA: this switch will only work if call 'SCOPI' is in that test. The reason being that most tests deal with blocks of different data to be sent or received all at once thus in block data; one pattern can't be singled out.

SW 04 SELECT DELAY PARAMETER: THIS SWITCH SHOULD BE USED WITH CARE AS TOO SHORT A DELAY WILL CAUSE VALID TESTS TO FAIL ON CERTAIN PROCESSORS. IT IS RECOMMENDED THAT THIS SWITCH ONLY BE IN CONJUNCTION WITH SCOPE LOOPS, E.G. SW 14,9,4,1 SET; SW 9,4,2,1 SET. THE SHORTEST PARAMETER IS 1; THE LONGEST ACCEPTED IS 17777E.

4.1.3 SWITCH REGISTER PRIORITYS

ERROR SWITCHES

1. SW 12 Delete print out/bell on error.
2. SW 13 Delete error printout.
3. SW 15 Halt on the error.
4. SW 08 Goto beginning of the test(on error).
5. SW 10 Goto next test(on error).

SCOPE SWITCHES

1. SW 09 (if enabled by 'SCOPI') on an error: If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is *usually* the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not enabled; and there is a *HARD* error (constant); SW08 is best.
(SW14=1, SW10=0, SW09=0, SW08=1). for intermittent errors; SW14=1 will loop on test regardless of error or not error.
(SW14=1, SW10=0, SW09=0, SW08=1,†B0)
2. SW 14
3. SW 11

4.2 STARTING ADDRESS

SA 200 - Address 200 is for normal execution of the diagnostic. This will do the major testing necessary for verification of hardware.

SA 210 - CABLE/ECHO - Terminal Tests. Starting at address 210 will give the user the option to verify the EIA cables at the dist pnt or verify a true link to any DEC supported EIA terminal supported by the DZ11.

NOTE: If address 000042 is non-zero the program assumes it is under 'B ACT11 or XXDP control and will act accordingly after *ALL* available DZ11's are tested the program will return to 'XXDP' or 'ACT-11'.

5. OPERATING PROCEDURE

When program is initially started messages as described in section four will be printed and program will begin running the diagnostic.

5.1 NORMAL START OF DIAGNOSTIC

On the first start of the diagnostic at address 200; if auto sizing is not used or whenever SW00=1; the following questions are asked and must be answered.

"1ST CSR ADDRESS (160000:163700): "

You must type in the first DZ11 CSR in the system you wish testing to begin at. RANGE: 160000:163700

"1ST VECTOR ADDRESS (300:770): "

You must type in the vector of the first DZ11 in the system under test. RANGE 300:770

"BR LEVEL (4:6): "

type in the priority level of the DZ11 that the above information has been given about. RANGE 4 or 5 or 6.†B

"TYPE "A" FOR EIA MODULE OR "B" FOR 20MA (A:B): "

Type "A" if running a DZ11-A,B,E (EIA).
Type "B" if running a DZ11-C,D,F (20MA).
Typing a <CR> defaults to EIA MODULES.

"MAINTAINCE MODE

[EXTERNAL <H325> (E)]
[INTERNAL <DZCSR03=1>(I)]
[STAGGERED <H327> (S)]

Type "E" or "I" or "S" depending on which mode you wish to run in. If running "EXTERNAL"; all selected lines must be terminated by a H325 test connector.

"# OF DZ11'S (IN OCTAL) (1:20): "

Type total number of DZ11's to be tested in the system. RANGE
is 1 thru 20 in octal.

***** IF SW03=1 THEN *****

If SW03=1 the following will be printed.

"LINES ACTIVE BY BIT (IN OCTAL) (001:377):"

Each bit represents a line and any combination of lines may be
selected (HOWEVER IN STAGGERED MODE TWO ADJACENT LINES MUST BE
SELECTED (0-1, 2-3, 4-5, 6-7))..

"DEFAULT BAUD RATE (IN OCTAL) (00:17): "

This gives the user a chance to change the default baud rate
used in APP. 90% of the test. Normal operation is a "17"
(19.2k) or "16" (9.6k). "00" (50 baud)- Not advised.

It is important to note that all DZ11's in the system must be
CONTIGIOUS for both ADDRESS and VECTORS. also all the EXTRA
PARAMETERS other than CSR and VECTORS are given to the EXISTING
DZ11's in the system. If not all DZ11's are same priority or if
the mode of operation is different for each DZ11; THIS MUST BE
"PATCHED" INTO THE CORRECT STATUS MAP ENTRY which is printed at
start time. An alternative is to put SW00=1 at start time;
answer questions about DZ11 under test and INDICATE ONLY 1 DZ11
in the system. IF THE STATUS MAP IS TO BE "PATCHED" IT MUST BE
DONE AFTER THE QUESTIONS ARE ANSWERED OR AFTER THE AUTO SIZE.

5.2 HOW TO RUN THE "CABLE/ECHO" TESTS.

†B Normal starting for the first time would be: LOAD ADDRESS 210; START WITH THE SWR EQUAL TO 213.

NOTE: SW00=1 ASKS FOR "VECTOR" AND "CSR"
SW01=1 ASKS FOR "WHICH TEST ECHO OR CABLE", "BAUD RATE", "LINE"
UNDER TEST. Program will print out:

"VECTOR ADDRESS--"

You type vector with a 'CR'.

"CONTROL REGISTER ADDRESS--"

You type in DZCSR under test.

"WHICH TEST ? ECHO OR CABLE (E OR C)"

Lets do the CABLE TEST first. **THIS TEST IS ONLY TO BE DONE ON THE EIA VERSION OF THE DZ11 NOT THE 20MA VERSION". Type "C"
<CR>

"BAUD RATE- "

type either 50, 110, 135, 150, 300, 600, 1200 1800, 2000. 2400, 3600, 4800, 7200, 9600 followed by <CR>

"LINE: "

You type the line which has the H325 test connector. (Type either 0, 1, 2, 3, 4, 5, 6, 7) Program will then print:

"CABLE TEST"

and if everything is working; the following will be printed:

"PASS DONE."

"PASS DONE."

etc.

to change lines; HIT ANY PRINTING KEY†B ON YOUR CONSOLE TERMINAL WHILE THE PROGRAM IS RUNNING and the following will be printed:

"LINE: "

Now change the H325 test connector to another line and type the new line. Program will then print:

"CABLE TEST"

"PASS DONE."

"PASS DONE."

Continue this operation until all lines are tested.

5.3 ECHO TEST

If program has already been started at 210 and the vector and address have been typed fBin; just load address 210 and start with SWR equal to 212. program will print:

"WHICH TEST ? ECHO OR CABLE (E OR C)"

Now type an "E" to do the ECHO TEST. program will print:

"BAUD RATE--"

Type BAUD RATE at which the terminal is set that is connected to the DZ11 dist pnl. program will print:

LINE: "

Type the line the terminal is connected to at the dist pnl then the program will print:

"TERMINAL ECHO TEST"

*** AT THIS POINT THE MESSAGE:

"THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789"

SHOULD BE PRINTED ON THE TERMINAL CONNECTED TO THE DZ11. IF THIS MESSAGE IS DESIRED TO BE CONTINUOUSLY OUTPUT; SET THE SWR TO 377 (SWR=377) WHILE IT IS BEING OUTPUT OR WHEN PROGRAM IS STARTED AT 210. WHEN THIS MESSAGE IS DONE AND THE SWR IS NOT EQUAL TO 377; THE CONSOLE WILL PRINT:

"TYPE A CHAR. ON DZ11 TERMINAL"

any printable char hit on DZ11 terminal should be echoed back on the terminal. **IF YOU HIT CNTRL C (↑C) ON THE DZ11↑B TERMINAL THE PROGRAM WILL PRINT:

"PASS DONE."

on the console terminal and the "QUICK BROWN FOX" will be printed on DZ11 terminal again and the echo test will be running. TO CHANGE LINES; do like cable test. HIT PRINTABLE KEY ON CONSOLE TERMINAL. And change the line on which the terminal is connected. And enter the new line to the program.

5.4 PROGRAM AND/OR OPERATOR ACTION

†B The typical approach should be

1. Halt on error (via SW 15=1) when ever an error occurs.
2. Clear SW 15.
3. Set SW 14: (loop on this test)
4. Set SW 13: (inhibit error print out)

The TEST NUMBER and PC will be typed out and possibly an error message (this depends on the test) to give the operator an idea as to the source of the problem. If it is necessary to know more information concerning the error report; LOOK IN THE LISTING for that TEST NUMBER which was typed out and then NOTE THE PC of the ERROR REPORT this way the EXACT FUNCTIONING of the test CAN BE INTERPETED.

6. ERRORS

As described previously there will always be a TEST NUMBER and PC typed out at the time of an error (providing SW 13=0 and SW 12=0). In most cases additional information will be supplied to the the error message which is to give the operator an indication of the error.

6.2 ERROR RECOVERY

If for some reason the DZ11 should 'HANG THE BUS' (gain control of bus so that console manual functions are inhibited) an init or power down/up is necessary for operator to regain control of cpu. If this should happen; look in location 'TSTNO' (address 1216) for the number of the test that was running at the time of the catastrophic error. In this way the operator will have an idea as to what the DZ11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

See section 4. (PLEASE)
Status table should be verified regardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completely isolate problems.

7.2 OPERATING RESTRICTIONS

Parameter must be input from user OR APT if "AUTO SIZING" is not used.

8. MISCELLANEOUS

8.1 EXECUTION TIME

All DZ11 device diagnostics will give an 'END PASS' message (providing no errors and sw12=0) within 2 min. This is assuming SW11=2 (DELETE ITERATIONS) is set to give the fastest possible execution. The actual execution time depends greatly on the POP11 CPU configuration.

8.2 PASS COMPLETE

NOTE: *EVERY* time the program is started; the tests will run as if SW11 (delete iterations) was up (=1). This is to 'VERIFY NO *HARD* ERRORS' as soon as possible. Therefore the first pass -EACH TIME PROGRAM IS STARTED- will be a 'QUICK PASS' until all DZ11's in system are tested. When the diagnostic has completed a pass the following is an example of the print out to be expected.

END PASS DZDZA-C CSR: 160010 VEC: 300 PASSES: 000001 ERRORS: 000000

NOTE: The numbers for CSR and VEC are not necessarily the values for the device. They are only for this example.

8.4 KEY LOCATIONS

SLPADR (1126) Contains the address where program will return when iteration count is reached or if loop on test is asserted.

NEXT (1360) Contains the address of the next test to be performed.

STSTNM (1122) Contains the number of the test now being performed.

RUN (1406) The bit in 'RUN' always points one past the DZ11 currently being tested. EXAMPLE: (RUN)
1304/0000000001000000 Means that DZ11 no.05 is the DZ11 now running.

STATUS MAP (1500)-(2000) These locations contain the information needed to test up to 16 (decimal) DZ11s sequentially. they contain the CSR, VECTOR and STATUS concerning the configuration of each DZ11.

DZACTV (1404) Each bit set in this location indicates that the associated DZ11 will be tested in turn. EXAMPLE: (DZACTV) 1300/0000000000011111 means that DZ11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DZACTV) 1300/0000000000010001 Means that DZ11 no. 00,04 will be tested.

\$BASE (1310) Contains the receiver csr of the current DZ11 under test.

B.4A MORE ON THAT 'STATUS TABLE' (1500-2000)

```

'MAP OF [ 1 STATUS'
1500 160010
1502 1B 000300
1504 000005
1506 000377
1510 017470
1512 000000

```

The above information will be repeated for each of up to 8 DZ11's in the system (these will follow under this table). EXPLANATION:

1500	160010	This is the system control register for the 1st DZ11 in the system.
1502	000300	This is vector 'A' for the first DZ11 in the system.
1504	000005	This represents the bus interrupt priority level of the DZ11. BIT15 of this location indicates either EIA or 20MA. if BIT15=0 module should be eia; if bit15=1 module should be 20ma.
1506	000377	This is the binary representation of what lines are to be tested.
1510	017470	This is the parameter location used in most of the tests. It indicated parameters of: RX ON, SPEED SELECT 17 (19.2K BAUD) EIGHT BITS PER CHAR, AND TWO STOP BITS. The user may alter the stop bits and the speed, but the remaining parameters should be left alone.
1512	000000	This location will contain either all zeros indicating that internal loop was selected as mode of operation or it will contain 10000 indicating that "staggered mode" was selected or it will contain 000200 indicating that "external" was the mode selected.

The above is repeated for each DZ11 in the system. The table is filled by AUTO SIZING or by the manual parameter input program as described previously. Also if desired by user; the locations may be altered by hand (toggled in) to suit the specific configuration.

8.5 *** METHOD OF AUTO SIZING ***

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The program will start at address 160000 and start 'REFERENCEING' the address in the pointer. If a NON-EX MEMORY TRAP occurs, the pointer (holding 160000) is updated by 10 and the above is repeated until address 163700 is reached. If a 'SLAVE SYNC RESPONSE' was issued by the DZ11 (or any other device) (no nzm trap), "MASTER SCAN ENABLE" is attempted to be set and the "TCR" bit for line 7 is set. "TRDY" is then tested to be set and both "TCR07" AND "MASTER SCAN ENABLE" are tested to be still set. If all of this worked; then a "DEVICE CLEAR" is issued testing that the bit can be read back and that after some time it self clears. If all of the above worked; this device is assumed to be a DZ11. If any of the above failed; updating of the pointer is done and the sequence is repeated.

NOTE: If the program does not find your DZ11; something is wrong and AUTO SIZING should not be done.

8.5.2 FINDING THE VECTOR

The vector area (address 300-776) is filled with the instruction IOT and '+2' (next address). Bit14 and Bit15 (TX INTERRUPT ENABLE AND MSTSCAN ENABLE) are set into the DZCSR. "TCR07" is then set. a delay is made and if no interrupt occurs (because of a bad DZ11) the program assumes vector address 300 and the problem should be fixed in the diagnostic. Once the problem is fixed; the program should be re-setup again to get correct vector. If an interrupt occurred; the address to which the DZ11 interrupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.5.3 PARAMETER ASSUMPTIONS.

Since too much hardware would need to be turned on to SIZE the rest of the parameters; the program must assume the remaining variations. The result is not to your specific configuration may be altered by hand (toggle in) if desired. In this way 95% of the parameter setup was done by the program and 5% by you.

THEREFORE:

- 1) BUS PRIORITY IS SET TO LEVELS.
- 2) ALL EIGHT LINES ARE ASSUMED TO BE TESTED.
- 3) DEFAULT BAUD RATE IS SET TO 17 (19.2 K).
4. MODE OF OPERATION IS "INTERNAL MODE".
5. MODULE IS ASSUMED TO BE "EIA" VERSION.
SET BIT 15 IN PRIORITY ENTRY OF MAP IF YOU HAVE A ZOMA MODULE.

If all adjustments please refer to section B.4a for greater detail.

9.0 RUNNING THE DZ11 DIAGNOSTIC UNDER APT

9.1.1 THE APT INTERFACE

DZDZA has been redesigned to be compatible with the APT-Automated Product Test system. It can be run as a standalone diagnostic or in either of the APT modes. Certain variables in the original APT module were reassigned to the areas set aside for APT interfacing. These new variables generally begin with a dollar sign (\$), e.g., \$DEVN, \$BASE.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

The diagnostic uses several variables in the region subtitled 'APT Mailbox-Etable'. These variables are:

\$SWREG - used if a software switch register is desired while under APT

\$VECT1 - used to specify the interrupt level and the first vector address

\$BASE - used to indicate bottom address of DZ11 under test

\$DEVN - a bit map representing which DZ11's will be tested

\$CDW1 - used to indicate which lines to run on all DZ11's

\$DDW1 - each of the \$DDW words describes the parameters (LPR) for a particular DZ11, going up to 16 DZ11's

9.1.3 RUNNING UNDER APT

The user should be familiar with the APT system. The APT timing parameters for the DZ11 diagnostic were based on an 11/40 processor. It may be necessary to add a few more seconds if the diagnostic is out on an 11/05 processor.

All of the variables mentioned in section 9.1.2 should be set up prior to running the diagnostic under APT.

NOTE

Be sure \$BASE points to the first DZ11 before running

Based on these values, the diagnostic will set up the status table. The user is then free to monitor under APT as normal.

2 COPYRIGHT (C) 1976
 DIGITAL EQUIPMENT CORP.
 MAYNARD, MASS. 01754

PROGRAM BY JERRYL PAYNE, JOHN EGOLF

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.

23 INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***

28 MISCELLANEOUS DEFINITIONS

40 GENERAL PURPOSE REGISTER DEFINITIONS

52 PRIORITY LEVEL DEFINITIONS

62 "SWITCH REGISTER" SWITCH DEFINITIONS

9C DATA BIT DEFINITIONS (BIT00 TO BIT15)

118 BASIC "CPU" TRAP VECTOR ADDRESSES

354 THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 USED IN THE PROGRAM.

424 BITS 15-11=CPU TYPE
 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
 11/70=06, PDQ=07, Q=10
 BIT 10=REAL TIME CLOCK
 BIT 9=FLOATING POINT PROCESSOR
 BIT 8=MEMORY MANAGEMENT

432 MEM. TYPE BYTE -- (HIGH BYTE)
 900 NSEC CORE=001
 300 NSEC BIPOLAR=002
 500 NSEC MOS=003

437 MEM.LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE

475 THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

481 EM :: POINTS TO THE ERROR MESSAGE
 DH :: POINTS TO THE DATA HEADER
 DT :: POINTS TO THE DATA
 DF :: POINTS TO THE DATA FORMAT

1088 INCREMENT THE PASS NUMBER (\$PASS)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO CYCLE

1149 THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
AND LOAD THE TEST NUMBER(\$STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>
THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
SW14=1 LOOP ON TEST
SW11=1 INHIBIT ITERATIONS
CALL
SCOPE ;;SCOPE=IOT

1225 ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.

CALL:
(BI) USING A TRAP INSTRUCTION
TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
OR
TYPE
MESADR

1927 ROUTINE USED TO "AUTO SIZE" THE DZ11
CSR AND VECTOR.
NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
ADDRESS RANGE (160000:163700)
AND THE VECTOR MAY BE ANY WHERE IN THE
FLOATING VECTOR RANGE (300:770)

2044 ***** TEST 1 *****
THIS TEST PROVES THE SLAVE SYNC RESPONSE
DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
DZCSR, DZRBUF, DZTCR, DZMSR

2087 ***** TEST 2 *****
THIS TEST PROVES THAT BIT "DCLR"
CAN BE SET AND THAT IT WILL CLEAR
BY ITSELF AFTER A PERIOD OF TIME.

2117 ***** TEST 3 *****
TEST TO VERIFY THAT BIT "MAINT" CAN
BE SET. THEN VERIFY THAT BIT "MAINT" CAN
BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
CLEARED BY A "DEVICE CLEAR"

2149 ***** TEST 4 *****
 TEST TO VERIFY THAT BIT "MSENAB" CAN
 BE SET. THEN VERIFY THAT BIT "MSENAB" CAN
 BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
 VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
 CLEARED BY A "DEVICE CLEAR"

2181 ***** TEST 5 *****
 TEST TO VERIFY THAT BIT "SILOEN" CAN
 BE SET. THEN VERIFY THAT BIT "SILOEN" CAN
 BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
 VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
 CLEARED BY A "DEVICE CLEAR"

2213 ***** TEST 6 *****
 TEST TO VERIFY THAT BIT "RIE" CAN
 BE SET. THEN VERIFY THAT BIT "RIE" CAN
 BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
 VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
 CLEARED BY A "DEVICE CLEAR"

2245 ***** TEST 7 *****
 TEST TO VERIFY THAT BIT "TIE" CAN
 BE SET. THEN VERIFY THAT BIT "TIE" CAN
 BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
 VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
 CLEARED BY A "DEVICE CLEAR"

2277 ***** TEST 10 *****
 THIS TESTS THAT ALL OF THE FOLLOWING
 BITS CAN BE: SET, CLEARED, CLEARED BY "DEVICE CLEAR "
 BITS TESTED ARE:
 TCR0, TCR1, TCR2, TCR3, TCR4, TCR5, TCR6, TCR7

†B 2319 ***** TEST 11 *****
 THIS TESTS THAT ALL OF THE FOLLOWING
 BITS CAN BE: SET, CLEARED, CLEARED BY "RESET INSTR *NOT* DEVICE CLEAR "
 BITS TESTED ARE:

2323 DTR0, DTR1, DTR2, DTR3, DTR4, DTR5, DTR6, DTR7
 THIS TEST IS NOT DONE IF MODULE IS 20MA VERSION

2371 ***** TEST 12 *****
 THIS TEST PERFORMS RESET TESTING &
 TESTING OF WRITE ONLY OR READ ONLY BIT
 TEST BITS "RDONE, BIT11, BIT10, BIT9, BIT8, BIT2, BIT1
 BIT0, SILOAL" ARE READ ONLY AND THAT TRDY IS
 ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.

- 2438 ***** TEST 13 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY AND WRITE ONLY BITS
IN REGISTER DZCSR
VERIFY THAT "TIE", "SILOEN", "RIE", "MSENAB", "MAINT"
ARE THE ONLY R/W BITS IN THE DZCSR.
THEN SET "DCLR" AND VERIFY THEY ARE CLEARED
- 2437 ***** TEST 14 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZRIBBUF
AND TESTING OF WRITE ONLY REGISTER DZLPR
- 2463 ***** TEST 15 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZMSR
AND TESTING OF WRITE ONLY REGISTER DZTDR
- 2489 ***** TEST 16 *****
VERIFY THAT IF WE ARE IN "STAGGERED" MODE
- 2491 THAT SETTING "DTR" FOR A LINE WILL
BRING UP "RING" AND "CARRIER" FOR THE
ASSOCIATED LINE IN WHICH WE ARE STAGGERED!
LINE0 DTR= LINE1 RING AND CARRIER
LINE1 DTR= LINE0 RING AND CARRIER
LINE2 DTR= LINE3 RING AND CARRIER
LINE3 DTR= LINE 4+8 RING AND CARRIER
ETC...
- 2546 ***** TEST 17 *****
- 2547 TEST TO VERIFY THAT IF IN "EXTERNAL"
MODE; SETTING DTR FOR SELECTED LINES
WILL BRING UP "CARRIER" AND "RING"
FOR THAT SAME LINE. NOTE: IF YOU HAVE
SELECTED MODE AS "EXTERNAL"; THE H325 TEST CONNECTER
MUST BE USED ON ALL SPECIFIED LINES.
LINES MAY BE SPECIFIED BY SW03=1
AND SW00=1 AT START TIME OR ALTERING
STATUS MAP.
- 2593 ***** TEST 20 *****
THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
IS READY TO BE LOADED, AND THAT THE LINE SPECI-
FIED IN BITS 8-10 OF DZCSR CORRESPOND
TO THE LINE SELECTED IN DZTCR
- 2627 ***** TEST 21 *****
TEST TO TRANSMIT ONE CHAR AND
RECEIVE ONE CHAR ON ONE LINE
AT A TIME. THE CHAR IS "252" AND
ALL SELECTED LINES WILL BE TURNED ON
ONE AT A TIME. THIS IS THE FIRST TIME ANY

DATA IS CHECKED IN THE RECEIVER.
 USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
 WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.

1B

- 2711 ***** TEST 22 *****
 THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
 CHARACTERS (FLAG MODE) AND THE RECEIVER RECEIVES (FLAG MODE)
 (ONE LINE AT A TIME BASED UPON VALID LINES)
- 2715 THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
- 2792 ***** TEST 23 *****
 THIS TEST WILL PROVE THAT:
 1) THE TRANSMITTER "BREAK BIT" WORKS
 2) THE RECEIVER CAN FLAG "FRAMING ERRORS"
 3) THE RECEIVER CAN FLAG "PARITY ERRORS"
 ONLY ONE LINE AT A TIME WILL BE EXERCISED.
 THIS TEST WILL NOT BE EXERCISED UNLESS
 CONNECTED BY EXTERNAL PLUG.
- 2859 ***** TEST 24 *****
 THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
 WHILE THE PROCESSOR STATUS IS SET EXACTLY
 TO WHAT THE DZ11 PRIORITY IS SET TO.
 DEFAULT PRIORITY IS AT 5 (240).
- 2927 ***** TEST 25 *****
 THIS TEST VERIFIES THAT THE DEVICE DOES INTERRUPT
 WHILE THE PROCESSOR STATUS IS SET TO EXACTLY
 ONE LEVEL LOWER THAN THE DZ11. DZ11 PRIORITY
 DEFAULT TO LEVEL 5 MINUS ONE LEVEL IS LEVEL 4.
- 3001 ***** TEST 26 *****
 THIS TEST VERIFIES THAT THE RECEIVER WILL
 INTERRUPT BEFORE THE TRANSMITTER EVEN
 THOUGH THE TRANSMITTER WAS ENABLED
 FIRST. SET PS TO LEVEL 7;
 GET RDONE AND TRDY TO SET;
 SET TX IE AND RX IE;
 CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
- 3111 ***** TEST 27 *****
 THIS TEST VERIFIES OVERRUN AND SILO ALARM
 ONE LINE AT A TIME - BASED UPON VALID LINES
 AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
 TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
 EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
 SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
 CHAR PULLED OUT OUT THE SILO.
 USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
 ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
 USED TO SCOPE SILO ALARM PULSES, ETC.

1B

- 3246 ***** TEST 30 *****
THIS TEST THAT "SILO ENABLE" WILL INHIBIT
RECEIVER INTERRUPTS AND THAT ON THE
16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
INTERRUPT WITH "RIE" SET.
THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
- 3331 ***** TEST 31 *****
THIS TEST RUNS ALL LINES FULL BORE
BASED UPON QUALIFIED LINES
.. THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
TRANSMITTER
- 3475 ***** TEST 32 *****
DZ11 RELATIVE TIMING TEST.
EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
PARAMETERS ARE:
EIGHT BITS/PER/CHAR - TWO STOP BITS AT
50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
2400, 3600, 4800, 7200, 9600 BAUD.
19.2 K BAUD - TWO STOP BITS AT
SEVEN, SIX, FIVE BITS/PER/CHAR.
AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
THE NEXT SELECTED LINE IS THE TESTED.
- 3572 ***** TEST 33 *****
THIS TEST VERIFIES THAT EVEN PARITY WORKS
FOR ALL ODD LINES SELECTED AND THAT ODD PARITY WORKS FOR ALL
EVEN LINES SELECTED.
THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
YOU ARE IN "STAGGED" MODE.
40(8) CHARS ARE USED FOR THIS TEST.
ALL SELECTED LINES WILL BE ENABLED
AT THE SAME TIME!
- 3671 ***** TEST 34 *****
THIS TEST VERIFIES THAT ODD PARITY WORKS FOR ALL ODD LINES
SELECTED AND THAT EVEN PARITY WORKS FOR ALL EVEN LINES SELECTED
THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
YOU ARE IN "STAGGERED" MODE.
40(8) CHARS ARE USED FOR THIS TEST.
ALL SELECTED LINES WILL BE ENABLED
AT THE SAME TIME!



3855 STARTING PROCEDURE
LOAD PROGRAM
LOAD ADDRESS 000210
PRESS START
PROGRAM WILL TYPE DZ11 ECHO/CABLE TEST
PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
TYPE IN E OR C RESPECTIVELY
PROGRAM WILL TYPE "VECTOR ADDRESS-"
TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"
TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "LINE NUMBER-"
TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "BAUD RATE-"
TYPE IN THE BAUD RATE OF THE DZ11 TERMINAL
FOLLOWED BY <CARRIAGE RETURN>
THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL
50
75
110
135 (ROUNDED OFF 134.5)
150
300
600
1200
1800
2000
2400
3600
4800
7200
9600
ALL OTHERS ARE REJECTED

3892 PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTED

401B72 TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
THE DIST PNL MUST HAVE JUMPER FROM DTR TO RQST TO SEND
IN ORDER FOR THIS TEST TO WORK!

14	INTRODUCTION TO DZ11 DIAGNOSTIC
24	BASIC DEFINITIONS
134	GENERAL DEFINITIONS AND EQUIVALENCES
307	TRAPCATCHER FOR UNEXPECTED INTERRUPTS
334	ACT11 HOOKS
353	PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.
357	COMMON TAGS
411	APT MAILBOX-ETABLE
479	ERROR POINTER TABLE
546	APT PARAMETER18 BLOCK
775	PROGRAM INITIALIZATION AND START UP.
1087	END OF PASS ROUTINE
1093	END OF PASS ROUTINE
1151	GENERAL UTILITIES (TYPE OUT, ERROR, SCOPE, ETC.)
1155	SCOPE HANDLER ROUTINE
1232	TYPE ROUTINE
1311	APT COMMUNICATIONS ROUTINE
1719	POWER DOWN AND UP ROUTINES
2053	DZ11 DEVICE DIAGNOSTICS.

```

1      .TITLE MD-11-DZDZA-C
2      ;*COPYRIGHT (C) 1976
3      ;*DIGITAL EQUIPMENT CORP.
4      ;*MAYNARD, MASS. 01754
5      ;*
6      ;*PROGRAM BY JERRYL PAYNE, JOHN EGOLF
7      ;*
8      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
9      ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
10     ;*
11     000001 $TN=1
12     ; STARTING PROCEDURE
13     ; LOAD PROGRAM
14     ; LOAD ADDRESS 000200
15     ; PRESS START
16     ; PROGRAM WILL TYPE "MAINDEC-11-DZDZAC/<200>/EIGHT LINE ASYNC MUX TESTS"
17     ; PROGRAM WILL TYPE "RUNNING" TO INDICATE THAT TESTING HAS STARTED
18     ; AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
19     ; AND THEN RESUME TESTING
20
21     .SBTTL BASIC DEFINITIONS
22
23     ;*INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***
24     001120 STACK= 1120
25     .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
26     .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
27
28     ;*MISCELLANEOUS DEFINITIONS
29     000011 HT= 11          ;;CODE FOR HORIZONTAL TAB
30     000012 LF= 12          ;;CODE FOR LINE FEED
31     000015 CR= 15          ;;CODE FOR CARRIAGE RETURN
32     000200 CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
33     177776 PS= 177776     ;;PROCESSOR STATUS WORD
34     .EQUIV PS,PSW
35     177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
36     177772 PIRQ= 177772  ;;PROGRAM INTERRUPT REQUEST REGISTER
37     177570 DSWR= 177570  ;;HARDWARE SWITCH REGISTER
38     177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
39
40     ;*GENERAL PURPOSE REGISTER DEFINITIONS
41     000000 R0= %0          ;;GENERAL REGISTER
42     000001 R1= %1          ;;GENERAL REGISTER
43     000002 R2= %2          ;;GENERAL REGISTER
44     000003 R3= %3          ;;GENERAL REGISTER
45     000004 R4= %4          ;;GENERAL REGISTER
46     000005 R5= %5          ;;GENERAL REGISTER
47     000006 R6= %6          ;;GENERAL REGISTER
48     000007 R7= %7          ;;GENERAL REGISTER
49     000006 SP= %6          ;;STACK POINTER
50     000007 PC= %7          ;;PROGRAM COUNTER
51
52     ;*PRIORITY LEVEL DEFINITIONS
53     000000 PRO= 0          ;;PRIORITY LEVEL 0
54     000040 PR1= 40         ;;PRIORITY LEVEL 1
55     000100 PR2= 100       ;;PRIORITY LEVEL 2
56     000140 PR3= 140       ;;PRIORITY LEVEL 3

```

57	000200	PR4=	200	::	PRIORITY LEVEL 4
58	000240	PR5=	240	::	PRIORITY LEVEL 5
59	000300	PR6=	300	::	PRIORITY LEVEL 6
60	000340	PR7=	340	::	PRIORITY LEVEL 7

.*"SWITCH REGISTER" SWITCH DEFINITIONS

61		SW15=	100000	
62		SW14=	40000	
63	100000	SW13=	20000	
64	040000	SW12=	10000	
65	020000	SW11=	4000	
66	010000	SW10=	2000	
67	004000	SW09=	1000	
68	002000	SW08=	400	
69	001000	SW07=	200	
70	000400	SW06=	100	
71	000200	SW05=	40	
72	000100	SW04=	20	
73	000040	SW03=	10	
74	000020	SW02=	4	
75	000010	SW01=	2	
76	000004	SW00=	1	
77	000002	.EQUIV	SW09, SW9	
78	000001	.EQUIV	SW08, SW8	
79		.EQUIV	SW07, SW7	
80		.EQUIV	SW06, SW6	
81		.EQUIV	SW05, SW5	
82		.EQUIV	SW04, SW4	
83		.EQUIV	SW03, SW3	
84		.EQUIV	SW02, SW2	
85		.EQUIV	SW01, SW1	
86		.EQUIV	SW00, SW0	

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

91	100000	BIT15=	100000	
92	040000	BIT14=	40000	
93	020000	BIT13=	20000	
94	010000	BIT12=	10000	
95	004000	BIT11=	4000	
96	002000	BIT10=	2000	
97	001000	BIT09=	1000	
98	000400	BIT08=	400	
99	000200	BIT07=	200	
100	000100	BIT06=	100	
101	000040	BIT05=	40	
102	000020	BIT04=	20	
103	000010	BIT03=	10	
104	000004	BIT02=	4	
105	000002	BIT01=	2	
106	000001	BIT00=	1	
107		.EQUIV	BIT09, BIT9	
108		.EQUIV	BIT08, BIT8	
109		.EQUIV	BIT07, BIT7	
110		.EQUIV	BIT06, BIT6	
111		.EQUIV	BIT05, BIT5	
112		.EQUIV	BIT04, BIT4	

113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168

```

.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

;*BASIC "CPU" TRAP VECTOR ADDRESSES
ERRVEC= 4           ;; TIME OUT AND OTHER ERRORS
RESVEC= 10          ;; RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14          ;; "T" BIT
TRTVEC= 14          ;B. TRACE TRAP
BPTVEC= 14          ;; BREAKPOINT TRAP (BPT)
IOTVEC= 20          ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24          ;; POWER FAIL
EMTVEC= 30          ;; EMULATOR TRAP (EMT) **ERROR**
TRAPVEC=34          ;; "TRAP" TRAP
TKVEC= 60           ;; TTY KEYBOARD VECTOR
TPVEC= 64           ;; TTY PRINTER VECTOR
PIRQVEC=240         ;; PROGRAM INTERRUPT REQUEST VECTOR

```

; INSTRUCTION DEFINITIONS

```

005746 PUSH1SP=5746 ;; DECREMENT PROCESSOR STACK 1 WORD
005726 POP1SP=5726  ;; INCREMENT PROCESSOR STACK 1 WORD
010046 PUSHRO=10046 ;; SAVE RO ON STACK
012600 POPRO=12600  ;; RESTORE RO FROM STACK
024646 PUSH2SP=24646 ;; DECREMENT STACK TWICE
022626 POP2SP=22626 ;; INCREMENT STACK TWICE

```

; DZ11 CONTROL AND STATUS REGISTER DEFINITIONS
(DZCSR) BIT DEFINITIONS

```

000010 MAINT = BIT3 ;; MAINTENANCE MODE ENABLE
000020 DCLR=BIT4   ;; DEVICE CLEAR
000040 MSENAB=BIT5 ;; MASTER SCAN ENABLE
000100 RIE=BIT6    ;; RECEIVER INTERRUPT ENABLE
000200 ROONE=BIT7 ;; RECEIVER DONE
010000 SILOEN= BIT12 ;; SILO ALARM ENABLE
020000 SILOAL = BIT13 ;; SILO ALARM
040000 TIE=BIT14  ;; TRANSMITTER INTERRUPT ENABLE
100000 TROY=BIT15  ;; TRANSMITTER READY

```

; DZCSR WORD DEFINITIONS

```

000000 TLO=0           ;; TRANSMIT LINE 0
000400 TL1=BIT8      ;; TRANSMIT LINE 1
001000 TL2=BIT9      ;; TRANSMIT LINE 2
001400 TL3=BIT9!BIT8 ;; TRANSMIT LINE 3
002000 TL4=BIT10     ;; TRANSMIT LINE 4
002400 TL5=BIT10!BIT8 ;; TRANSMIT LINE 5
003000 TL6=BIT10!BIT9 ;; TRANSMIT LINE 6
003400 TL7!8=BIT10!BIT9!BIT8 ;; TRANSMIT LINE 7

```

169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224

:DZRBUF BIT DEFINITIONS

010000 PARER=BIT12 : PARITY ERROR
020000 FRMERR=BIT13 : FRAME ERROR
040000 OVRUN=BIT14 : OVERRUN ERROR
100000 DVALID=BIT15 : DATA VALID

:DZRBUF WORD DEFINITIONS

000000 RLD=0 : RECEIVER LINE 0
000400 RL1=BIT8 : RECEIVER LINE 1
001000 RL2=BIT9 : RECEIVER LINE 2
001400 RL3=BIT9:BIT8 : RECEIVER LINE 3
002000 RL4=BIT10 : RECEIVER LINE 4
002400 RL5=BIT10:BIT8 : RECEIVER LINE 5
003000 RL6=BIT10:BIT9 : RECEIVER LINE 6
003400 RL7=BIT10:BIT9:BIT8 : RECEIVER LINE 7

:DZLPR WORD DEFINITIONS

000000 LP0=0 : LINE PARAMETER 0
000001 LP1=BIT0 : LINE PARAMETER 1
000002 LP2=BIT1 : LINE PARAMETER 2
000003 LP3=BIT1:BIT0 : LINE PARAMETER 3
000004 LP4=BIT2 : LINE PARAMETER 4
000005 LP5=BIT2:BIT0 : LINE PARAMETER 5
000006 LP6=BIT2:BIT1 : LINE PARAMETER 6
000007 LP7=BIT2:BIT1:BIT0 : LINE PARAMETER 7

000000 FIVE=0 : FIVE BITS/CHAR, 1 STOP BIT
000010 SIX=BIT3 : SIX BITS/CHAR, 1 STOP BIT
000020 SEVEN=BIT4 : SEVEN BITS/CHAR, 1 STOP BIT
000030 EIGHT=BIT4:BIT3 : EIGHT BITS/CHAR, 1 STOP BIT
000040 FIVES=BIT5 : FIVE BITS/CHAR, 2 STOP BITS
000050 SIXS=BIT5:BIT3 : SIX BITS/CHAR, 2 STOP BITS
000060 SEVENS=BIT5:BIT4 : SEVEN BITS/CHAR, 2 STOP BITS
000070 EIGHTS=BIT5:BIT4:BIT3 : EIGHT BITS/CHAR, 2 STOP BITS

000100 PARITY=BIT6 : PARITY ENABLE+BD
000200 ODDPAR=BIT7 : ODD PARITY ENABLED
000000 ONESTOP=0 : ONE STOP BIT ENABLED
000040 TWOSTOP=BIT5 : TWO STOP BITS ENABLED
000000 EVEPAR=0 : EVEN PARITY ENABLED
010000 RCVON=BIT12 : ENABLE RECEIVER (RECEIVER ON)

000000 S50=0 : SPEED 50 BAUD
000400 S75=BIT8 : SPEED 75 BAUD
001000 S110=BIT9 : SPEED 110 BAUD
001400 S134=BIT9:BIT8 : SPEED 134.5 BAUD
002000 S150=BIT10 : SPEED 150 BAUD
002400 S300=BIT10:BIT8 : SPEED 300 BAUD
003000 S600=BIT10:BIT9 : SPEED 600 BAUD
003400 S1200=BIT10:BIT9:BIT8 : SPEED 1200 BAUD

GENERAL DEFINITIONS AND EQUIVALENCES

225	004000	S1800=BIT11	:SPEED 1800 BAUD
226	004400	S2000=BIT11!BIT8	:SPEED 2000 BAUD
227	005000	S2400=BIT11!BIT9	:SPEED 2400 BAUD
228	005400	S3600=BIT11!BIT9!BIT8	:SPEED 3600 BAUD
229	006000	S4800=BIT11!BIT10	:SPEED 4800 BAUD
230	006400	S7200=BIT11!BIT10!BIT8	:SPEED 7200 BAUD
231	007000	S9600=BIT11!BIT10!BIT9	:SPEED 9600 BAUD
232	007400	S19200=BIT11!BIT10!BIT9!BIT8	:SPEED 19200 BAUD

:DZTCR BIT DEFINITIONS

233			
234			
235			
236	000001	TCR0=BIT0	:TCR0
237	000002	TCR1=BIT1	:TCR1
238	000004	TCR2=BIT2	:TCR2
239	000010	TCR3=BIT3	:TCR3
240	000020	TCR4=BIT4	:TCR4
241	000040	TCR5=BIT5	:TCR5
242	000100	TCR6=BIT6	:TCR6
243	000200	TCR7=BIT7	:TCR7
244	000400	DTR0=BIT8	:DTR0
245	001000	DTR1=BIT9	:DTR1
246	002000	DTR2=BIT10	:DTR2
247	004000	DTR3=BIT11	:DTR3
248	010000	DTR4=BIT12	:DTR4
249	020000	DTR5=BIT13	:DTR5
250	040000	DTR6=BIT14	:DTR6
251	100000	DTR7=BIT15	:DTR7

:DZMSR BIT DEFINITIONS

252			
253			
254			
255	000001	RING0=BIT0	:RING INDICATED ON LINE 0
256	000002	RING1=BIT1	:RING INDICATED ON LINE 1
257	000004	RING2=BIT2	:RING INDICATED ON LINE 2
258	000010	RING3=BIT3	:RING INDICATED ON LINE 3
259	000020	RING4=BIT4	:RING INDICATED ON LINE 4
260	000040	RING5=BIT5	:RING INDICATED ON LINE 5
261	000100	RING6=BIT6	:RING INDICATED ON LINE 6
262	000200	RING7=BIT7	:RING INDICATED ON LINE 7
263	000400	C00=BIT8	:CARRIER PRESENT ON LINE 0
264	001000	C01=BIT9	:CARRIER PRESENT ON LINE 1
265	002000	C02=BIT10	:CARRIER PRESENT ON LINE 2
266	004000	C03=BIT11	:CARRIER PRESENT ON LINE 3
267	010000	C04=BIT12	:CARRIER PRESENT ON LINE 4
268	020000	C05=BIT13	:CARRIER PRESENT ON LINE 5
269	040000	C06=BIT14	:CARRIER PRESENT ON LINE 6
270	100000	C07=BIT15	:CARRIER PRESENT ON LINE 7

:DZTDR BIT DEFINITIONS

271			
272			
273			
274			
275	000400	BRK0=BIT8	:BREAK FOR LINE 0
276	001000	BRK1=BIT9	:BREAK FOR LINE 1
277	002000	BRK2=BIT10	:BREAK FOR LINE 2
278	004000	BRK3=BIT11	:BREAK FOR LINE 3
279	010000	BRK4=BIT12	:BREAK FOR LINE 4
280	020000	BRK5=BIT13	:BREAK FOR LINE 5

281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400

040000
100000

BRK6=BIT14
BRK7=BIT15

:BREAK FOR LINE 6
:BREAK FOR LINE 7

:TABLE OF LOOP AROUND FUNCTIONS (H325)

I	↑
V	↑
REC	TRANS
DATA	DATA

I	↑
V	↑
CO	RTS

I	↑↑B
V	↑
RING	DTR

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

```

303 ;*****
304 ;-----
305 ; TRAPCATCHER FOR ILLEGAL INTERRUPTS
306 ; THE STANDARD "TRAP CATCHER" IS PLACED
307 ; BETWEEN ADDRESS 0 TO ADDRESS 776.
308 ; IT LOOKS LIKE "PC+2 HALT".
309 ;-----
310 ;*****
311
312 000000 . =0
313 ; STANDARD INTERRUPT VECTORS
314 ;-----
315
316 000010 . =10
317 000010 010650 SET.PS ; FAKE "MTPS" INSTRUCTION TRAP
318 000012 000340 PR7 ; MAKE SURE PS IS PRIORITY 7
319
320 000020 . =20
321 000020 004654 .SCOPE ; SCOPE LOOP HANDLER
322 000022 000340 PR7 ; HANDLE AT PRIORITY 7
323 000024 007530 $PWDRN ; POWER FAIL HANDLER
324 000026 000340 340 ; SERVICE AT PRIORITY LEVEL 7
325 000030 006620 $ERROR ; ERROR HANDLER
326 000032 000340 340 ; SERVICE AT PRIORITY LEVEL 7
327 000034 006512 .TRPSRV ; GENERAL HANDLER DISPATCH SERVICE
328 000036 000340 340 ; SERVICE AT PRIORITY LEVEL 7
329
330 .SBTTL ACT11 HOOKS
331 ;*****
332 ; HOOKS REQUIRED BY ACT11
333 000040 $SVPC= ; SAVE PC
334 000046 . =46
335 000046 004610 $ENDAD ; ;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
336 000052 000052 . =52
337 000052 000000 .WORD 0 ; ;2)SET LOC.52 TO ZERO
338 000040 .=$SVPC ; ; RESTORE PC
339
340 . =174
341 000174 00001800 DISPREG:0 ; SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 115
342 000176 000000 SWREG: 0 ; SOFTWARE SWITCH REGISTER FOR SWITCHLESS 115
343 000200 [ 0200
344 000200 000137 002150 JMP .START ; GO TO START OF PROGRAM
345 000210 000210 . =210
346 000210 000137 023142 JMP XSTART ; GOTO CABLE TEST/ECHO TEST
347
348
349 001000 . =1000
350 001000 005200 040515 047111 MTITLE: .ASCIZ <200><12>/MAINDEC-11-DZDZAC.<200>/EIGHT LINE ASYNC MUX TESTS.<200>

```

351
352
*B 353
354
355
356
357 DC1120
358 00112C
359 001120 000000
360 001122 000
361 001123 00C
362 001124 000000
363 001126 000000
364 001130 000000
365 001132 000000
366 001134 000
367 001135 001
368 001136 000000
369 001140 000000
370 001142 000000
371 001144 000000
372 001146 000000
373 001150 000000
374 001152 000000
375 001154 000
376 001155 000
377 001156 000000
378 001160 177570
379 001162 177570
380 001164 177560
381 001166 177562
382 001170 177564
383 001172 177566
384 001174 000
385 001175 002
386 001176 012
387 001177 000
388 001200 000000
389
390 001202 000000
391 001204 000000
392 001206 000000
393 001210 000000
394 001212 000000
395 001214 000000
396 001216 000000
397 001220 000000
398 001222 000000
399 001224 000000
400 001226 000000
401 001230 077
402 001231 015
403 001232 000012
404
405
406

.SBTTL COMMON TAGS

: *THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
: *USED IN THE PROGRAM.

SCMTAG: .=1120

:: START OF COMMON TAGS

\$STNM: .WORD 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0

:: CONTAINS THE TEST NUMBER
:: CONTAINS ERROR FLAG
:: CONTAINS SUBTEST ITERATION COUNT
:: CONTAINS SCOPE LOOP ADDRESS
:: CONTAINS SCOPE RETURN FOR ERRORS
:: CONTAINS TOTAL ERRORS DETECTED
:: CONTAINS ITEM CONTROL BYTE
:: CONTAINS MAX. ERRORS PER TEST
:: CONTAINS PC OF LAST ERROR INSTRUCTION
:: CONTAINS ADDRESS OF 'GOOD' DATA
:: CONTAINS ADDRESS OF 'BAD' DATA
:: CONTAINS 'GOOD' DATA
:: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED

\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD 0
\$DISP: .WORD 0

:: AUTOMATIC MODE INDICATOR
:: INTERRUPT MODE INDICATOR

\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566

:: ADDRESS OF SWITCH REGISTER
:: ADDRESS OF DISPLAY REGISTER
:: TTY KBD STATUS
:: TTY KBD BUFFER
:: TTY PRINTER STATUS REG. ADDRESS
:: TTY PRINTER BUFFER REG. ADDRESS

\$NULL: *B .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0

:: CONTAINS NULL CHARACTER FOR FILLS
:: CONTAINS # OF FILLER CHARACTERS REQUIRED
:: "INSERT FILL CHARS. AFTER A "LINE FEED"
:: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
:: CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED

\$REGO: .WORD 0
\$REG1: .WORD 0
\$REG2: .WORD 0
\$REG3: .WORD 0
\$REG4: .WORD 0
\$REG5: .WORD 0
\$TMP0: .WORD 0
\$TMP1: .WORD 0
\$TMP2: .WORD 0
\$TMP3: .WORD 0

:: CONTAINS ((\$REGAD)+0)
:: CONTAINS ((\$REGAD)+2)
:: CONTAINS ((\$REGAD)+4)
:: CONTAINS ((\$REGAD)+6)
:: CONTAINS ((\$REGAD)+10)
:: CONTAINS ((\$REGAD)+12)
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED

\$TIMES: 0
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCII <12>

:: MAX. NUMBER OF ITERATIONS
:: QUESTION MARK
:: CARRIAGE RETURN
:: LINE FEED

: *B*****
.SBTTL APT MAILBOX-ETABLE

```

407 :*****
408 .EVEN
409 001234 $MAIL: : APT MAILBOX
410 001234 000000 $MSGTY: .WORD AMSGTY : MESSAGE TYPE CODE
411 001236 000000 $FATAL: .WORD AFATAL : FATAL ERROR NUMBER
412 001240 000000 $TESTN: .WORD ATESTN : TEST NUMBER
413 001242 000000 $PASS: .WORD APASS : PASS COUNT
414 001244 000000 $DEVCT: .WORD ADEVCT1BT : DEVICE COUNT
415 001246 000000 $UNIT: .WORD AUNIT : I/O UNIT NUMBER
416 001250 000000 $MSGAD: .WORD AMSGAD : MESSAGE ADDRESS
417 001252 000000 $MSGLG: .WORD AMSGLG : MESSAGE LENGTH
418 001254 $ETABLE: : APT ENVIRONMENT TABLE
419 001254 000 $ENV: .BYTE AENV : ENVIRONMENT BYTE
420 001255 000 $ENVM: .BYTE AENVM : ENVIRONMENT MODE BITS
421 001256 000000 $SWREG: .WORD ASWREG : APT SWITCH REGISTER
422 001260 000000 $USWR: .WORD AUSWR : USER SWITCHES
423 001262 000000 $CPUOP: .WORD ACPUOP : CPU TYPE, OPTIONS
424 :* :
425 :* :
426 :* :
427 :* :
428 :* :
429 :* :
430 001264 000 $MAMS1: .BYTE AMAMS1 : HIGH ADDRESS, M.S. BYTE
431 001265 000 $MTYP1: .BYTE AMTYP1 : MEM. TYPE, BLK#1
432 :* :
433 :* :
434 :* :
435 :* :
436 001266 000000 $MADR1: .WORD AMADR1 : HIGH ADDRESS, BLK#1
437 :* :
438 001270 000 $MAMS2: .BYTE AMAMS2 : HIGH ADDRESS, M.S. BYTE
439 001271 000 $MTYP2: .BYTE AMTYP2 : MEM. TYPE, BLK#2
440 001272 000000 $MADR2: .WORD AMADR2 : MEM. LAST ADDRESS, BLK#2
441 001274 000 $MAMS3: .BYTE AMAMS3 : HIGH ADDRESS, M.S. BYTE
442 001275 000 $MTYP3: .BYTE AMTYP3 : MEM. TYPE, BLK#3
443 001276 000000 $MADR3: .WORD AMADR3 : MEM. LAST ADDRESS, BLK#3
444 001300 000 $MAMS4: .BYTE AMAMS4 : HIGH ADDRESS, M.S. BYTE
445 001301 000 $MTYP4: .BYTE AMTYP4 : MEM. TYPE, BLK#4
446 001302 000000 $MADR4: .WORD AMADR4 : MEM. LAST ADDRESS, BLK#4
447 001304 000000 $VECT1: .WORD AVECT1BT1 : INTERRUPT VECTOR#1, BUS PRIORITY#1
448 001306 000000 $VECT2: .WORD AVECT2 : INTERRUPT VECTOR#2, BUS PRIORITY#2
449 001310 160010 $BASE: .WORD ABASE : BASE ADDRESS OF EQUIPMENT UNDER TEST
450 001312 000000 $DEVN: .WORD ADEVN : DEVICE MAP
451 001314 000000 $CDW1: .WORD ACDW1 : CONTROLLER DESCRIPTION WORD#1
452 001316 000000 $CDW2: .WORD ACDW2 : CONTROLLER DESCRIPTION WORD#2
453 001320 000000 $DDW0: .WORD ADDW0 : DEVICE DESCRIPTOR WORD#0
454 001322 000000 $DDW1: .WORD ADDW1 : DEVICE DESCRIPTOR WORD#1
455 001324 000000 $DDW2: .WORD ADDW2 : DEVICE DESCRIPTOR WORD#2
456 001326 000000 $DDW3: .WORD ADDW3 : DEVICE DESCRIPTOR WORD#3
457 001330 000000 $DDW4: .WORD ADDW4 : DEVICE DESCRIPTOR WORD#4
458 001332 000000 $DDW5: .WORD ADDW5 : DEVICE DESCRIPTOR WORD#5
459 001334 000000 $DDW6: .WORD ADDW6 : DEVICE DESCRIPTOR WORD#6
460 001336 000000 $DDW7: .WORD ADDW7 : DEVICE DESCRIPTOR WORD#7
461 001340 000000 $DDW8: .WORD ADDW8 : DEVICE DESCRIPTOR WORD#8
462 001342 000000 $DDW9: .WORD ADDW9 : DEVICE DESCRIPTOR WORD#9

```

463	001344	000000	\$DDW10:	.WORD	ADDW10	::	DEVICE	DESCRIPTOR	WORD#10
464	001346	000000	\$DDW11:	.WORD	ADDW11	::	DEVICE	DESCRIPTOR	WORD#11
465	001350	000000	\$DDW12:	.WORD	ADDW12	::	DEVICE	DESCRIPTOR	WORD#12
466	001352	000000	\$DDW13:	.WORD	ADDW13	::	DEVICE	DESCRIPTOR	WORD#13
467	001354	000000	\$DDW14:	.WORD	ADDW14	::	DEVICE	DESCRIPTOR	WORD#14
468	001356	000000	\$DDW15:	.WORD	ADDW15	::	DEVICE	DESCRIPTOR	WORD#15
469									
470									
471	001360		SETEND:						

473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ;;POINTS TO THE ERROR MESSAGE
;* DH ;;POINTS TO THE DATA HEADER
;* DT ;;POINTS TO THE DATA
;* DF ;;POINTS TO THE DATA FORMAT

\$ERRTB:

;;PROGRAM CONTROL PARAMETERS
:-----

NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA

;;PROGRAM VARIABLES
:-----

LINE: 377 ;DEFAULT ALL EIGHT LINES RUNNING
PAR: 17470 ;PARAMETERS: 8 BITS/CHAR, 2 STOP BITS, 19200 BAUD, NO PARIT
MODE: 0 ;DEFAULT MAINTENANCE MODE
SAVLIN: 0 ;LINE NUMBER
XMTLIN: 0 ;TRANSMISSION LINE NUMBER
XMTCNT: 0 ;COUNT OF WORDS IN A TRANSMISSION PATTERN
REGIST: 0 ;DEVICE ADDRESS STORAGE LOCATION
SAVPC: 0 ;PROGRAM COUNTER STORAGE
DZACTV: .BLKW 1 ;*DZ11'S SELECTED ACTIVE.
RUN: 1 ;*POINTER ONE PAST RUNNING DEVICE.
DZNUM: .BLKB 1 ;*OCTAL NUMBER OF DZ11'S.
SAVNUM: .BYTE 1 ;*WORKABLE NUMBER.
.EVEN
ACTIVE: DZ.MAP ;TABLE POINTER.

001360

001360 000000
001362 000000

001364 000377
001366 017470
001370 000000
001372 000000
001374 000000
001376 000000
001400 000000
001402 000000
001404 000001
001406 000001
001410 000001
001411 001
001412 001500

```

512
513
514
515
516 001414 000 EIAFLG: .BYTE 0 ;0=EIA 100000=20MA
517 001415 000 INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
518 001416 000 HDRFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
519 001417 000 MNTFLG: .BYTE 0 ;MAINTENANCE BIT SET FLAG
520 001420 000 DONFLG: .BYTE 0 ;TRANSMISSION COMPLETION FLAG
521 001422 .EVEN
522 :DATA VARIABLES
523 001422 000000 T00: .WORD 0
524 001424 000000 T01: .WORD 0
525 001426 000000 T02: .WORD 0
526 001430 000000 T03: .WORD 0
527 001432 000000 T04: .WORD 0
528 001434 000000 T05: .WORD 0
529 001436 000000 T06: .WORD 0
530 001440 000000 T07: .WORD 0
531 001442 000000 TR0: .WORD 0
532 001444 000000 TR1: .WORD 0
533 001446 000000 TR2: .WORD 0
534 001450 000000 TR3: .WORD 0
535 001452 000000 TR4: .WORD 0
536 001454 000000 TR5: .WORD 0
537 001456 000000 TR6: .WORD 0
538 001460 000000 TR7: .WORD 0
539 001462 STOP:
540 .SBTTL APT PARAMETER BLOCK
541
542
543
544
545 001462 18
546 000024 000024 .SX=. ;SAVE CURRENT LOCATION
547 000024 000200 =24 ;SET POWER FAIL TO POINT TO START OF PROGRAM
548 000044 000044 200 ;FOR APT START UP
549 000044 001462 =44 ;POINT TO APT INDIRECT ADDRESS PNTR.
550 001462 $APTHDR ;POINT TO APT HEADER BLOCK
551 .=$X ;RESET LOCATION COUNTER
552
553
554
555 001462 $APTHD:
556 00146182 000000 $SHIBTS: .WORD 0 ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
557 001464 001234 $MADDR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
558 001466 000132 $TSTM: .WORD 90. ;RUN TIM OF LONGEST TEST
559 001470 000137 $PASTM: .WORD 95. ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
560 001472 000137 $UNITM: .WORD 95. ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
561 001474 000052 .WORD $ETEND-$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)
562 ;DZ11 STATUS TABLE AND ADDRESS ASSIGNMENTS
563
564
565 001500 .=1500
566 001500 DZ.MAP:
567

```

568	001500	000001	DZCR0:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 0
569	001502	000001	DZVC0:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 0
570	001504	000001	DZLV0:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
571	001506	000001	LINE0:	.BLKW	1	;ALL LINES SELECTED
572	001510	000001	PAR0:	.BLKW	1	;PARAMETERS
573	001512	000001	MANT0:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
574						
575	001514	000001	DZCR1:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 1
576	001516	000001	DZVC1:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 1
577	001520	000001	DZLV1:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
578	001522	000001	LINE1:	.BLKW	1	;ALL LINES SELECTED
579	001524	000001	PAR1:	.BLKW	1	;PARAMETERS
580	001526	000001	MANT1:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
581						
582	001530	000001	DZCR2:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 2
583	001532	000001	DZVC2:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 2
584	001534	000001	DZLV2:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
585	001536	000001	LINE2:	.BLKW	1	;ALL LINES SELECTED
586	001540	000001	PAR2:	.BLKW	1	;PARAMETERS
587	001542	000001	MANT2:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
588						
589	001544	000001	DZCR3:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 3
590	001546	000001	DZVC3:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 3
591	001550	000001	DZLV3:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
592	001552	000001	LINE3:	.BLKW	1	;ALL LINES SELECTED
593	001554	000001	PAR3:	.BLKW	1	;PARAMETERS
594	001556	000001	MANT3:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
595						
596	001560	000001	DZCR4:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 4
597	001562	000001	DZVC4:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 4
598	001564	000001	DZLV4:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
599	001566	000001	LINE4:	.BLKW	1	;ALL LINES SELECTED
600	001570	000001	PAR4:	.BLKW	1	;PARAMETERS
601	001572	000001	MANT4:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
602						
603	001574	000001	DZCR5:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 5
604	001576	000001	DZVC5:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 5
605	001600	000001	DZLV5:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
†B 606	001602	000001	LINE5:	.BLKW	1	;ALL LINES SELECTED
607	001604	000001	PAR5:	.BLKW	1	;PARAMETERS
608	001606	000001	MANT5:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
609						
610	001610	000001	DZCR6:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 6
611	001612	000001	DZVC6:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 6
612	001614	000001	DZLV6:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
613	001616	000001	LINE6:	.BLKW	1	;ALL LINES SELECTED
614	001620	000001	PAR6:	.BLKW	1	;PARAMETERS
615	001622	000001	MANT6:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
616						
617	001624	000001	DZCR7:	.BLKW	1	;CONTROL STATUS REGISTER FOR DZ11 NUMBER 7
618	001626	000001	DZVC7:	.BLKW	1	;RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 7
619	001630	000001	DZLV7:	.BLKW	1	;PRIORITY LEVEL AND EIA FLAG SELECTOR
620	001632	000001	LINE7:	.BLKW	1	;ALL LINES SELECTED
621	001634	000001	PAR7:	.BLKW	1	;PARAMETERS
622	001636	000001	MANT7:	.BLKW	1	;MAINTENANCE MODE FOR THIS DEVICE
623						

624	001640	000001		DZCR10: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 10
625	001642	000001		DZVC10: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 10
626	001644	000001		DZLV10: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
627	001646	000001		LINE10: .BLKW	1	; ALL LINES SELECTED
628	001650	000001		PAR10: .BLKW	1	; PARAMETERS
629	001652	000001		MANT10: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
630						
631	001654	000001		DZCR11: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 11
632	001656	000001		DZVC11: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 11
633	001660	000001		DZLV11: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
634	001662	000001		LINE11: .BLKW	1	; ALL LINES SELECTED
61835	001664	000001	000001	PAR11: .BLKW	1	; PARAMETERS
636	001666	000001		MANT11: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
637						
638	001670	000001		DZCR12: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 12
639	001672	000001		DZVC12: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 12
640	001674	000001		DZLV12: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
641	001676	000001		LINE12: .BLKW	1	; ALL LINES SELECTED
642	001700	000001		PAR12: .BLKW	1	; PARAMETERS
643	001702	000001		MANT12: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
644						
645	001704	000001		DZCR13: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 13
646	001706	000001		DZVC13: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 13
647	001710	000001		DZLV13: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
648	001712	000001		LINE13: .BLKW	1	; ALL LINES SELECTED
649	001714	000001		PAR13: .BLKW	1	; PARAMETERS
650	001716	000001		MANT13: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
651						
652	001720	000001		DZCR14: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 14
653	001722	000001		DZVC14: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 14
654	001724	000001		DZLV14: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
655	001726	000001		LINE14: .BLKW	1	; ALL LINES SELECTED
656	001730	000001		PAR14: .BLKW	1	; PARAMETERS
657	001732	000001		MANT14: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
658						
659	001734	000001		DZCR15: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 15
660	001736	000001		DZVC15: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 15
661	001740	000001		DZLV15: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
662	001742	000001		LINE15: .BLKW	1	; ALL LINES SELECTED
663	001744	000001		PAR15: .BLKW	1	; PARAMETERS
664	001746	000001		MANT15: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
665						
666	001750	000001		DZCR16: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 16
667	001752	000001		DZVC16: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 16
668	001754	000001		DZLV16: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
669	001756	000001		LINE16: .BLKW	1	; ALL LINES SELECTED
670	001760	000001		PAR16: .BLKW	1	; PARAMETERS
671	001762	000001		MANT16: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
672						
673	001764	000001		DZCR17: .BLKW	1	; CONTROL STATUS REGISTER FOR DZ11 NUMBER 17
674	001766	000001		DZVC17: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 17
675	001770	000001		DZLV17: .BLKW	1	; PRIORITY LEVEL AND EIA FLAG SELECTOR
676	001772	000001		LINE17: .BLKW	1	; ALL LINES SELECTED
677	001774	000001		PAR17: .BLKW	1	; PARAMETERS
678	001776	000001		MANT17: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
679						

N03

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 16
DZDZAC.P11 21-OCT-76 13:07 APT PARAMETER BLOCK
680 002000 177777 DZ.END: 177777

PAGE: 0039

681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722

:DEFINITIONS FOR TRAP SUBROUTINE CALLS
:POINTERS TO SUBROUTINES CAN BE FOUND
:IN THE TABLE IMMEDIATELY FOLLOWING THE DEFINITIONS

::*****

```

:-----
:TRPTAB:
ADVANCE=TRAP+0 ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
      .ADVANCE
SCOPI=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
      .SCOPI
TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
      .TYPE
INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
      .INSTR
+BINSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
      .INSTER
PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
      .PARAM
SETFLG=TRAP+6 ;CALL TO SET FLAG ROUTINE
      .SETFLG
SAVOS=TRAP+7 ;CALL TO REGISTER SAVE ROUTINE
      .SAVOS
RESOS=TRAP+10 ;CALL TO REGISTER RESTORE ROUTINE
      .RESOS
CONVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE
      .CONVRT
CNVRT=TRAP+12 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
      .CNVRT
DEVICE.CLR=TRAP+13 ;CALL TO ISSUE A DEVICE CLEAR
      .DEVICE.CLR
DELAY=TRAP+14 ;CALL TO DELAY FOR FAST CPU'S
      .DELAY
PARMD=TRAP+15 ;CONVERT DECIMAL STRING TO OCTAL
      .PARMD
PAWCH=TRAP+16 ;SET FLAG ECHO OR CABLE
      .PAWCH
DCLASM=TRAP+17 ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
      .DCLASM

```

::*****

+B

```

723                                     :DZ11 VECTOR AND REGISTER INDIRECT POINTERS
724                                     :WORKING AREA
725
726 002042 160040 DZCSR: 160040 :R/W
727 002044 160041 HDZCSR: 160041 :R/W
728 002046 160042 DZRBUF: 160042 :READ ONLY
729 002050 160043 HDZRBUF: 160043 :READ ONLY
730 002052 160042 DZLPR: 160042 :WRITE ONLY
731 002054 160043 HDZLPR: 160043 :WRITE ONLY
732 002056 160044 DZTCR: 160044 :R/W
733 002060 160045 HDZTCR: 160045 :R/W
734 002062 160046 DZMSR: 160046 :READ ONLY
735 002064 160047 HDZMSR: 160047 :READ ONLY
736 002066 160046 DZTDR: 160046 :WRITE ONLY
737 002070 160047 HDZTDR: 160047 :WRITE ONLY
738                                     :DEFAULT DZ VECTORS
739 002072 000300 DZRIV: 300 :REC INTR VECTOR
740 002074 000302 DZRIIS: 302 :REC INTR STATUS
741 002076 000304 DZTIV: 304 :XMIT INTR VECTOR
742 002100 000306 DZTIS: 306 :XMIT INTR STATUS
743
744

```

745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768

002102
002102 000000
002104 000000
002106 000000
002110 000000
002112 000000
002114 000000
002116 000000
002120 000000
002122 000000
002124 000000
002126 000000
002130 000000
002132 000000
002134 000000
002136 000000
002140 000000
002142 000000
002144 000000
002146 000000

:TIME TABLE FOR RELATIVE TIMING TESTS

TMTBL:
T50: 0
T75: 0
T110: 0
T134: 0
T150: 0
T300: 0
T600: 0
T1200: 0
T1800: 0
T2000: 0
T2400: 0
T3600: 0
T4800: 0
T7200: 0
T9600: 0
TEIGHT: 0
TSEVEN: 0
TSIX: 0
TFIVE: 0

```

769
770
771
772
773
774
775
776
777 002150 .START:
778 002150 000005 RESET ;CLEAR THE WORLD. START NEW ENVIRONMENT
779 002152 012706 001120 MOV #STACK,SP ;SET UP STACK
780 002156 136427 000340 MTPS #PR7 ;LOCK OUT INTERRUPTS
781 002162 012737 007530 000024 MOV #SPWRDN,2#24 ;SET UP POWER FAIL VECTOR
782 002170 113737 001410 001411 MOV#B DZNUM, SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
783 002176 005037 001242 CLR $PASS ;CLEAR PASS COUNT
784 002202 105037 001123 CLR#B $ERFLG ;CLEAR ERROR FLAG
785 002206 012737 001500 001412 MOV #DZ.MAP, ACTIVE ;GET MAP POINTER.
786 002214 012737 000001 001406 MOV #1,RUN ;POINT POINTER TO FIRST DEVICE.
787 002222 005037 001132 CLR $ERTTL ;CLEAR ERROR COUNT
788 002226 005037 001136 CLR $ERRPC ;CLEAR LAST ERROR POINTER
789 002232 005037 001122 CLR $TSTNM ;SET UP FOR TEST 1
790 002236 012737 002150 001126 MOV #.START,$LPADR ;SET UP FOR POWER FAIL BEFORE
791 ;TESTING STARTS
792 ;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
793 002244 013746 000006 MOV 6,-(SP) ;SAVE BUS ERROR PS
794 002250 013746 000004 MOV 4,-(SP) ;SAVE BUS ERROR PC
795 002254 012737 002274 000004 MOV #20$ 4 ;SET UP TO TRAP TO THIS ROUTINE
796 002262 022777 177777 176670 CMP #-1,$SWR ;CAN 177570 BE REFERENCED?
797 002270 001402 BEQ 22$ ;IF SO AND IT IS -1, TREAT LIKE SWITCHLESS
798 002272 000407 BR 21$ ;IF YES, SKIP AROUND THE SETUP
799 002274 022626 20$: POP2SP ;REMOVE THE TRAP FROM THE STACK
800 002276 012737 000176 001160 22$: MOV #SWREG,SWR ;IF NO TRAP COMES HERE POINT TO SOFTWARE SWR
801 002304 012737 000174 001162 MOV #DISPREG,DISPLAY ;POINT TO SOFTWARE DISPLAY REGISTER
802 002312 012637 000004 21$: MOV (SP)+,4 ;RESTORE THE BUS ERROR VECTOR
803 002316 012637 000006 MOV (SP)+,6
804 002322 005737 000042 TST 42 ;WORKING UNDER A MONITOR ?
805 002326 001402 BEQ 31$ ;NO
806 002330 000137 004126 JMP 63$ ;IF YES, SKIP THE TERMINAL INTERROGATION
807 002334 105737 001415 31$: TSTB INIFLG ;HAVE WE ALREADY BEEN HERE TODAY?
808 002340 001004 BNE 29$ ;IF SO, SKIP PRINTING THE TITLE
809 002342 104402 001000 TYPE ,MTITLE ;PRINT THE DIAGNOSTIC'S TITLE
810 002346 105337 001415 DECB INIFLG ;SET THE ONCE ONLY FLAG
811 002352 105737 001255 29$: TSTB $ENVM ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
812 002356 100004 BPL 30$ ;IF NOT, GO CHECK FOR AUTO-SIZING
813 002360 004737 011310 JSR PC,SETAPT ;OTHERWISE, GO DO APT SIZING FROM ETABLE
814 002364 000137 004152 JMP 16$ ;GO PRINT DZ STATUS TABLE
815 002370 032777 000001 176562 30$: BIT #SW00,$SWR ;RESELECT ?
816 002376 001011 BNE 32$ ;IF YES, GO SET UP THE 1B INFORMATION
817 002400 122737 000377 001415 CMPB #377,INIFLG ;ON 1ST START: MUST ANSWER QUESTION
818 002406 001003 BNE .+10 ;IF NOT ANSWERING QUESTIONS
819 002410 105777 176544 TSTB $SWR ;ARE U AUTO SIZING?
820 002414 100402 BMI 32$ ;NO AUTO SIZE! NO SW00=1 ON 1ST START!
821 002416 000137 003104 JMP 73$ ;IF NO, SKIP THE INTERROGATION
822 002422 012700 001500 32$: MOV #DZ.MAP,RO ;POINT TO THE BEGINNING OF THE MAP TABLE
823 002426 105037 001416 CLR#B HDRFLG ;MAKE SURE A MAP GETS PRINTED
824 002432 005020 65$: CLR (RO)+ ;CLEAR A TABLE LOCATION

```

F04

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 21
 DZDZAC.P11 21-OCT-76 13:07 PROGRAM INITIALIZATION AND START UP.

PAGE: 0044

```

825 002434 020027 002000      CMP      RD, #DZ.END      ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
826 002440 001374              BNE      65$             ;IF NOT, CLEAR THE NEXT LOCATION IN THE TABLE
827 002442 105337 001415      DECB     IN:FLG          ;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!
828
829                               ;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
830                               ;TABLE AND SET UP THE DIAGNOSTIC.
831
832                               ;GET THE BASE ADDRESS OF THE DZ11'S
833
834                               33$:
835 002446 104403              INSTR     ;CALL THE STRING INPUT ROUTINE
836 002450 00331824          66$      ;POINTER TO MESSAGE TO BE PRINTED
837 002452 104405              PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
838 002454 160000              160000    ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
839 002456 163770              163770    ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
840 002460 001500              DZCRO    ;POINTER TO MAP LOCATION TO BE FILLED
841 002462 007                .BYTE    7      ;MASK OF INVALID BITS FOR THIS PARAMETER
842 002463 001                .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
843 002464 013737 001500 001310  MOV      DZCRO, $BASE ;COPY BASE ADDRESS TO ETABLE
844
845                               ;GET THE BASE VECTOR ADDRESS
846
847 18002472                 34$:
848 002472 104403              INSTR     ;CALL THE STRING INPUT ROUTINE
849 002474 003370              67$      ;POINTER TO MESSAGE TO BE PRINTED
850 002476 104405              PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
851 002500 000300              300      ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
852 002502 000776              776      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
853 002504 001502              DZVCO    ;POINTER TO MAP LOCATION TO BE FILLED
854 002506 003                .BYTE    3      ;MASK OF INVALID BITS FOR THIS PARAMETER
855 002507 001                .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
856 002510 013737 001502 001304  MOV      DZVCO, $VECT1 ;COPY VECTOR TO ETABLE
857
858                               ;GET THE BUS REQUEST LEVEL
859
860 002516 104403              INSTR     ;CALL THE STRING INPUT ROUTINE
861 002520 003431              68$      ;POINTER TO MESSAGE TO BE PRINTED
862 002522 104405              PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
863 002524 000004              4        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
864 002526 000007              7        ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
865 002530 001504              DZLVO    ;POINTER TO MAP LOCATION TO BE FILLED
866 002532 000                .BYTE    0      ;MASK OF INVALID BITS FOR THIS PARAMETER
867 002533 001                .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
868 002534 113737 001504 001305  MOVB     DZLVO, $VECT1+1 ;GET BUS REQUEST LEVEL INTO ETABLE
869 002542 106337 001305      ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
870 002546 106337 001305      ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
871 002552 106337 001305      ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
872 002556 106337 001305      ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
873 002562 106337 001305      ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
874
875                               ;FIND OUT IF MODULE IS EIA OR 20 MA.
876
877 002566 104402 004012      TYPE     74$           ;PRINT EIA MESSAGE
878 002572 005037 001220      CLR      $TMP1         ;USE $TMP1
879 002576 11805777 176362    B0$:     TSTB     $STKS ;IS KEYBOARD DONE?
880 002602 100375              BPL      80$           ;IF NOT, WAIT FOR IT
  
```

```

881 002604 017746 176356      MOV      2$TKB, -(SP)      ; IF YES, PUT CHARACTER ON STACK
882 002610 042716 000240      BIC      #240, (SP)       ; STRIP DOWN CHARACTER
883 002614 122726 000015      CMPB     #15, (SP)+      ; IS IT ?
884 002620 001414              BEQ      81$              ; IF SO, GET OUT
885 002622 014677 176344      MOV      -(SP), 2$TPB     ; IF NOT, PRINT CHARACTER
886 002626 042737 100000 001504  BIC      #BIT15, DZLVO    ; CLEAR EIA FLAG
887 002634 122726 000102      CMPB     #102, (SP)+     ; IS IT A B?
888 002640 001356              BNE      80$              ; IF NOT, GO BACK FOR INPUT
889 002642 052737 100000 001504  BIS      #BIT15, DZLVO    ; IF SO, SET FLAG
890 002650 000752              BR       80$              ; GET MORE INPUT
891 002652                      81$:
892
893                      ; GET THE MODE OF OPERATION (E, I, S)
894
895 002652 104403      INSTR                      ; CALL THE STRING INPUT ROUTINE
896 002654 003642      72$                        ; POINTER TO THE MESSAGE TO BE PRINTED
897 002656 104406      SETFLG                      ; CALL THE MAINTENANCE FLAG SETUP ROUTINE
898 002660 001512      MANTO                        ; THIS IS THE FLAG BEING SETUP
899
900                      ; GET THE NUMBER OF DZ11'S RUNNING
901
902 002662 104403      INSTR                      ; CALL THE STRING INPUT ROUTINE
903 002664 003609      71$                        ; POINTER TO MESSAGE TO BE PRINTED
904 002666 104405      PARAM                      ; CALL THE OCTAL TO ASCII CONVERT ROUTINE
905 002670 000001      1                          ; LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
906 002672 000020      16.                        ; HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
907 002674 001220      $TMP1                      ; POINTER TO MAP LOCATION TO BE FILLED
908 002676          000        .BYTE 0                          ; MASK OF INVALID BITS FOR THIS PARAMETER
909 002677          001        .BYTE 1                          ; NUMBER OF PARAMETERS TO STORE
910
911 002700 012737 000377 001506      MCV      #377, LINED      ; SET UP DEFAULT LINES
912 002706 012737 017470 001510      MOV      #17470, PARO     ; SET UP DEFAULT LPR PARAMETER
913
914 002714 032777 000010 176236      BIT      #5W03, 2$SWR     ; RECEIVER ON; 19.2 KBAUD; 2$STOP BITS; 8 BIT/CHAR
915 002722 001402              BEQ      40$              ; DO YOU WANT PARAMETERS?
916 002724 004737 003134              JSR      PC, 23$          ; IF NO, SKIP THE PARAMETER CALL
917 002730 012737 000001 001312 40$:      MOV      #1, $DEVM        ; GET PARAMETERS
918 002736 113737 001220 001410      MOV      $TMP1, DZNUM     ; INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
919 002744 113737 001220 001411      MOV      $TMP1, SAVNUM    ; COPY THE NUMBER OF DEVICES
920 002752 005337 001220          62$:      DEC      $TMP1           ; COPY A BACKUP NUMBER
921 002756 001404              BEQ      61$              ; $TMP1 CONTAINS THE COUNT OF UNINITIALIZED
922 002760 000261              SEC                      ; SELECTED DEVICES
923 002762 006137 001312      ROL      $DEVM           ; SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
924 002766 000771              BR       62$              ; POINT TO THE NEXT DEVICE
925 002770 013737 001312 001222 61$:      MOV      $DEVM, $TMP2     ; GO DO THIS PROCEDURE AGAIN
926 002776 013737 001312 001404      MOV      $DEVM, DZACTV    ; # OF TIMES
927 003004 012700 001500              MOV      #DZCR0, R0       ; COPY THE ACTIVE DEVICE PARAMETER
928 003010 012701 001514              MOV      #DZCR1, R1       ; SET A POINTER TO THE SPECIFIED INFORMATION
929 003014 012702 001320              MOV      #DOW0, R2        ; POINT R1 TO THE REST OF THE MAP TABLE
930 003020 000241              CLC                      ; POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
931 003022 006037 001222              ROR      $TMP2           ; INITIALIZE THE "C" BIT FOR A ROTATION
932 003026 006237 001222          64$:      ASR      $TMP2           ; SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
933 003032 103404              BCS      41$              ; ISOLATE A SELECTION FLAG IN THE "C" BIT
934 003034 012711 177777      MOV      #-1, (R1)        ; IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
935 003040 000137 004126              JMP      63$              ; TERMINATE THE LIST
936 003044 012011          41$:      MCV      (R0)+, (R1)      ; GO TO THE NEXT BLOCK
                                ; ADDRESS
    
```


PROGRAM INITIALIZATION AND START UP.

```

993 003232 104405          PARAM          ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
994 003234 000000          0              ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
995 003236 000017          17             ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
996 003240 001510          PAR0          1B      ;POINTER TO MAP LOCATION TO BE FILLED
997 003242          000          .BYTE      0          ;MASK OF INVALID BITS FOR THIS PARAMETER
998 003243          001          .BYTE      1          ;NUMBER OF PARAMETERS TO STORE
999 003244 012702 001506    MOV          #LINED,R2 ;POINT TO THE LINE SELECTION PARAMETER
1000 003250 012703 001510    MOV          #PAR0,R3  ;POINT TO THE CHOSEN PARAMETERS
1001 003254 011304          MOV          (R3),R4  ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
1002 003256 006304          ASL          R4        ;ALIGN INDEX ON WORD BOUNDARY
1003 003260 016437 030130 006604    MOV          DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
1004 003266 000313          SWAB         (R3)      ;PLACE IN HIGH BYTE
1005 003270 052713 010070    BIS          #10070,(R3) ;PLACE EXTRA PARAMETERS INTO LOC
1006 003274 011262 000014    28$:      MOV          (R2),14(R2) ;LOAD THE LINES
1007 003300 011363 000014    MOV          (R3),14(R3) ;LOAD THE PARAMETERS
1008 003304 062702 000014    ADD          #14,R2     ;POINT TO THE NEXT SET
1009 003310 062703 000014    ADD          #14,R3     ;... OF BOTH PARAMETERS
1010 003314 020327 001774    CMP          R3,#PAR17 ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
1011 003320 001365          BNE          28$      ;IF NOT, GO LOAD SOME MORE PARAMETERS
1012 003322 000207          RTS           PC      ;RETURN TO CALLING BLOCK 1B
1013 003324 030600 052123 041440 66$:      .ASCIZ      <200>/1ST CSR ADDRESS (160000:163700): /
(1) 003370 030600 052123 053040 67$:      .ASCIZ      <200>/1ST VECTOR ADDRESS (300:770): /
(1) 003431          200 051102 046040 68$:      .ASCIZ      <200>/BR LEVEL (4:6): /
(1) 003454 046200 047111 051505 69$:      .ASCIZ      <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:377): /
(1) 003530 042200 043105 052501 70$:      .ASCIZ      <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17): /
(1) 003600 021600 047440 020106 71$:      .ASCIZ      <200>/# OF DZ11'S <IN OCTAL> (1:20): /
(1) 003642 046600 044501 052116 72$:      .ASCII      <200>/MAINTENANCE MODE/
(1) 003663          200 055440 054105 1B.ASCII   <200>/ [EXTERNAL <H325> (E)]/
(1) 003717          200 055440 047111 .ASCII      <200>/ [INTERNAL <DZCSR03=1>(I)]/
(1) 003753          200 055440 052123 .ASCIZ      <200>/ [STAGGERED <H327> (S)]: /
(1) 004012 052200 050131 020105 74$:      .ASCIZ      <200>/TYPE "A" FOR EIA MODULE OR "B" FOR 20 MA (A:B): /
(1) 004074 042600 052116 051105 75$:      .ASCIZ      <200>/ENTER DELAY PARAMETER: /
(1)          004126          .EVEN
(1)          004126          63$:
1014 004126 122737 000377 001415    CMPB        #377,INIFLG ;ONLY DO AUTO SIZE ON 1ST START
1015 004134 001006          BNE          16$
1016 004136 032777 000200 175014    BIT          #BIT7,JSWR ;BIT7=1??
1017 004144 001002          BNE          16$
1018 004146 004737 011462          JSR          PC,AUTO.SIZE ;GO DO THE AUTO SIZE
1019 004152 105737 001416          16$:      TSTB        HDRFLG    ;HAS THE TABLE BEEN TYPED YET?
1020 004156 001021          BNE          1$
1021 004160 105337 001416          DECB        HDRFLG    ;INDICATE THAT THE TABLE WILL BE TYPED
1022 004164 104402 010261          TYPE        XHEAD    ;TYPE MAP HEADER
1023 004170 012700 001500          MOV          #DZ.MAP,R0 ;SET POINTER
1024 004174 010037 001220          5$:      MOV          R0,$TMP1  ;POINT TO THE MAP LOCATION
1025 004200 012037 001222          MOV          (R0)+,$TMP2 ;SET DATA
1026 004204 022737 177777 001222    CMP          #-1,$TMP2  ;END OF LIST?
1B 1027          004212 001403          BEQ          1$        ;BR IF YES
1028 004214 104411          17$:      CONVRT      XSTATQ    ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
1029 004216 010350          XSTATQ      ;CONVERT THE DATA AT THIS ADDRESS
1030 004220 000765          SR          5$        ;GO PRINT THE NEXT PARAMETER
1031 004222 005737 000042          1$:      TST          #42      ;IS PROGRAM RUNNING UNDER MONITOR
1032 004226 001026          BNE          3$
1033 004230 032777 000100 174722    BIT          #SW06,JSWR ;DESELECT SPECIFIC DEVICES??
1034 004236 001422          BEQ          3$
1035 004240 104402 010202          TYPE        .MNEW    ;TYPE THE MESSAGE.

```

1036	004244	005000				CLR	RO		:ZERO DATA DISPLAY
1037	004246	000000				HALT			:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1038	004250	027737	174704	001312		CMP	2SWR,\$DEVN		:IS THE NUMBER VALID?
1039	004256	101404				BLOS	2\$:BR IF NUMBER IS OK.
1040	004260	104402	010054			TYPE	,MERR3		:TELL USER OF INVALID NUMBER.
1041	004264	000000			9\$:	HALT			:STOP EVERY THING.
1042	004266	000776				BR	9\$:RESTART THE PROGRAM AGAIN.
1043	004270	017737	174664	001404	2\$:	MOV	2SWR,DZACTV		:GET NEW DEVICE PATTERN
1044	004276	013700	001404			MOV	DZACTV,RO		:SHOW THE USER WHAT HE SELECTED.
1045	004302	000000				HALT			:CONTINUE DYNAMIC SWITCHES.
1046	004304	032777	000020	174646	3\$:	BIT	#SW04,2SWR		:CHECK TO SEE IF DELAY COUNT CHANGES
1047	004312	001407				BEQ	18\$:IF NOT, GO CLEAR VECTOR AREA
1048	004314	104403				INSTR			:CALL THE STRING INPUT ROUTINE
1049	004316	004074				75\$:POINTER TO MESSAGE TO BE PRINTED
1050	004320	104405				PARAM			:CALL THE OCTAL TO ASCII CONVERT ROUTINE
1051	004322	000001				1			:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1052	004324	177777				177777			:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1053	004326	006604				DLYCNT			:POINTER TO MAP LOCATION TO BE FILLED
1054	004330	000				.BYTE	0		:MASK OF INVALID BITS FOR THIS PARAMETER
1055	004331	001				.BYTE	1		:NUMBER OF PARAMETERS TO STORE
1056	004332	012700	000300		18\$:	MOV	#300,RO		:PREPARE TO CLEAR THE FLOATING
1057	004336	012701	000302			MOV	#302,R1		:VECTOR AREA. 300-776
1058	004342	010120			4\$:	MOV	R1,(R0)+		:START PUTTING "PC+2 - HALT"
1059	004344	005021				CLR	(R1)+		:IN VECTOR AREA.
1060	004346	022021				CMP	(R0)+,(R1)+		:POP POINTERS
1061	004350	022700	001000			CMP	#1000,RO		:ALL DONE??
1062	004354	001372				BNE	4\$:BR IF NO.
1063									
1064									:TEST START AND RESTART
1065									:-----↑B-----
1066									
1067	004356	012706	001120			.BEGIN: MOV	#STACK,SP		:SET UP STACK
1068	004362	106427	000340			MTPS	#PR7		:LOCK OUT INTERRUPTS
1069	004366	005737	000042			TST	2#42		:IS PROGRAM UNDER MONITOR CONTROL
1070	004372	001015				BNE	2\$:BR IF YES
1071	004374	032777	000004	174556		BIT	#BIT2,2SWR		:CHECK FOR LOCK ON TEST
1072	004402	001406				BEQ	1\$:BR IF NO LOCK DESIRED.
1073	004404	104402	010100			TYPE	,MLOCK		:TYPE LOCK SELECTED.
1074	004410	012737	000240	004672		MOV	#NOP,TTST		:ADJUST SCOPE ROUTINE.
1075	004416	000403				BR	2\$:CONTINUE ALONG.
1076	004420	013737	005114	004672	1\$:	MOV	BRW,TTST		:PREPARE NORMAL SCOPE ROUTINE
1077	004426	012737	010752	001126	2\$:	MOV	#CYCLE,\$LPADR		:START AT "CYCLE" FIND WHICH DEVICE TO TEST
1078	004434	104402	007771			TYPE	,MR		:TYPE "RUNNING"
1079	004440	000177	174462			JMP	2\$LPADR		:START TESTING

```

1080 ;END OF PASS
1081 ;TYPE NAME OF TEST
1082 ;UPDATE PASS COUNT
1083 ;CHECK FOR EXIT TO ACT-11
1084 ;RESTART TEST
1085 .SBTTL END OF PASS ROUTINE
1086
1087 ;*****↑B*****
1088 ;*INCREMENT THE PASS NUMBER ($PASS)
1089 ;*IF THERES A MONITOR GO TO IT
1090 ;*IF THERE ISN'T JUMP TO CYCLE
1091
1092 $ECP:
1093 SCOPE
1094 CLR $ERRPC ;CLEAR LAST ERROR PC
1095 CLRFB $ERFLG ;CLEAR ERROR FLAG
1096 TYPE ,MEPASS ;TYPE END PASS
1097 TYPE ,MCSRX ;TYPE CSR
1098 CNVRT ,XCSR ;SHOW IT
1099 TYPE ,MVECX ;TYPE VECTOR
1100 CNVRT ,XVEC ;SHOW IT
1101 ↑B INC $PASS ;RAISE PASS COUNT
1102 TYPE ,MPASSX ;TYPE PASSES
1103 CNVRT ,XPASS ;SHOW IT
1104 DEC $PASS ;RESTORE PASS COUNT
1105 TYPE ,MERRX ;TYPE ERRORS
1106 CNVRT ,XERR ;SHOW IT
1107 DECB $AVNUM ;ARE ALL DEVICES TESTED?
1108 BNE $DOAGN ;BR IF NO.
1109 MOVFB DZNUM, SAVNUM ;RESTORE THE COUNT
1110 CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
1111 INC $PASS ;INCREMENT THE PASS NUMBER
1112 BIC #10000, $PASS ;DON'T ALLOW A NEG. NUMBER
1113 DEC (PC)+ ;LOOP?
1114 $EOPCT: .WORD 1
1115 BGT $DOAGN ;YES
1116 MOV (PC)+, 2(PC)+ ;RESTORE COUNTER
1117 $ENDCT: .WORD 1
1118 $EOPCT
1119 $GET42: MOV 2#42, RO ;GET MONITOR ADDRESS
1120 BEQ $DOAGN ;BRANCH IF NO MONITOR
1121 RESET ;CLEAR THE WORLD
1122 $ENDAD: JSR PC, (RO) ;GO TO MONITOR
1123 NOP ;SAVE ROOM
1124 ↑B NOP ;FOR
1125 NOP ;ACT11
1126 $DOAGN:
1127 JMP 2(PC)+ ;RETURN
1128 $RTNAD: .WORD CYCLE
1129
1130 XCSR: 1
1131 .BYTE 6,2
1132 DZCSR
1133 XVEC: 1
1134 .BYTE 3,2
1135 DZRIV

```

```

1136 004640 000001          XPASS: 1
1137 004642          006      002      .BYTE 6,2
1138 004644 001242          $PASS
1139 004646†B          000001      XERR: 1
1140 004650          006      002      .BYTE 6,2
1141 004652 001132          $ERTTL
1142
1143          ;SCOPE LOOP AND ITERATION HANDLER
1144          -----
1145
1146          .SBTTL SCOPE HANDLER ROUTINE
1147
1148          ;*****
1149          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1150          ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1151          ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1152          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1153          ;*SW14=1      LOOP ON TEST
1154          ;*SW11=1      INHIBIT ITERATIONS
1155          ;*CALL
1156          ;*      SCOPE          ;;SCOPE=IOT
1157
1158          $SCOPE:
1159          .SCOPE: JSR      PC,SERV.G      ;FIND OUT IF †G WAS HIT
1160                  CLR      $ERRPC      ;CLEAR LAST ERROR PC.
1161                  CMP      †TST1+2,(SP) ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
1162                  BEQ      $XTSTR      ;IF SO, DON'T LOOP ON IT
1163                  BR       1$          ;GOTO 1$ (IF LOCK SW02=1; THIS LOC =240)
1164                  TSTB     †STKS      ;KEYBOARD DONE?
1165                  BPL      $OVER      ;BR IF NO. (LOCK: HIT KE†BY TO GOTO NEXT TEST)
1166                  MOV      †STKB,-2(SP) ;CLEAR DONE BIT
1167                  BIT      †BIT14,‡SWR ;LOOP ON PRESENT TEST?
1168                  BNE      $OVER      ;YES IF SW14=1
1169          ;*****START OF CODE FOR THE XOR TESTER*****
1170          $XTSTR: BR       6$          ;IF RUNNING ON THE "XOR" TESTER CHANGE
1171                  MOV      †$ERRVEC,-(SP) ;THIS INSTRUCTION TO A "NOP" (NOP=240)
1172                  MOV      †$ERRVEC      ;SAVE THE CONTENTS OF THE ERROR VECTOR
1173                  TST      †177060      ;SET FOR TIMEOUT
1174                  MOV      (SP)+,‡$ERRVEC ;RESTORE THE ERROR VECTOR
1175          †B 1175          ;GO TO THE NEXT TEST
1176                  BR       $$VLAD      ;CLEAR THE STACK AFTER A TIME OUT
1177                  CMP      (SP)+,(SP)+ ;RESTORE THE ERROR VECTOR
1178                  MOV      (SP)+,‡$ERRVEC ;RESTORE THE ERROR VECTOR
1179                  BR       $OVER      ;LOOP ON THE PRESENT TEST
1180          6$;*****END OF CODE FOR THE XOR TESTER*****
1181          2$: TSTB     $ERFLG      ;HAS AN ERROR OCCURRED?
1182                  BEQ      3$          ;BR IF NO
1183          4$: CLRB     $ERFLG      ;ZERO THE ERROR FLAG
1184                  CLR      $TIMES     ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1185          3$: BIT      †BIT11,‡SWR ;INHIBIT ITERATIONS?
1186                  BNE      1$          ;BR IF YES
1187                  TST      $PASS      ;IF FIRST PASS OF PROGRAM
1188                  BEQ      1$          ;INHIBIT ITERATIONS
1189                  INC      $ICNT      ;INCREMENT ITERATION COUNT
1190                  CMP      $TIMES,$ICNT ;CHECK THE NUMBER OF ITERATIONS MADE
1191                  BGE      $OVER      ;BR IF MORE ITERATION REQUIRED

```

```

1192 005026 012737 000001 001124 1S:  MOV #1,SICNT      ;;REINITIALIZE THE ITERATION COUNTER
1193 005034 013737 005116 001226      MOV SMXCNT,SIMES  ;;SET NUMBER OF ITERATIONS TO DO
1194 005042 105237 001122      $SVLAD: INCB STSTNM      ;;COUNT TEST NUMBERS
1195 005046 113737 001122 001240      MOVB STSTNM,STESTN  ;;SET TEST NUMBER IN APT MAILBOX
1196 005054 011637 001126      MOV (SP),SLPADR    ;;SAVE SCOPE LOOP ADDRESS
1197 005060 013777 001122 174074 $OVER: MOV STSTNM,DISP  ;;DISPLAY TEST NUMBER
1198 005066 013716 001126      MOV $LPADR,(SP)   ;;FUDGE RETURN ADDRESS
1199 005072 105037 001417      3S:  CLRB MNTFLG    ;;CLEAR THE MAINTENANCE BIT SE↑BTTER AFTER EACH TEST
1200 005076 005737 001370      TST  MODE        ;;HAS THE MODE BEEN CHANGED?
1201 005102 001003      BNE  4S          ;;IF NOT INTERNAL, GO DO A TEST
1202 005104 112737 000010 001417      MOVB #MAINT,MNTFLG  ;;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
1203 005112 000002      4S:  RTI         ;;GO DO THE TEST
1204 005114 000406      BRW:  406
1205 005116 000005      $MXCNT: 5        ;;MAX. NUMBER OF ITERATIONS
1206
1207      ;;CHECK FOR FREEZE ON CURRENT DATA
1208      -----
1209
1210 005120 032777 001000 174032 .SCOPI: BIT #SW09,@SWR  ;;IS SW09=1(SET)?
1211 005126 001405      BEQ  1S         ;;BR IF NOT SET.
1212 005130 005737 001362      TST  LOCK      ;;IS THER A TIGHT LOOP SPECIFIED?
1213 005134 001402      BEQ  1S         ;;IF NO, RETURN
1214 005136 013716 001362      MOV  LOCK,(SP)  ;;IF YES, GOTO THE ADDRESS IN LOCK.
1215 005142 000002      1S:  RTI         ;;GO BACK.
1216
1217 005144 032777 010000 174006 .TYPE:  BIT #SW12,@SWR  ;;INHIBIT ALL PRINTOUT??
1218 005152 001403      BEQ  1S         ;;IF NOT, GO TYPE
1219 005154 062716 000002      ADD  #2,(SP)    ;;SKIP OVER MESSAGE POINTER
1220 005160 000002      RTI         ;;RETURN TO WHERE PROCEDURE WAS INVOKED
1221 005162
1222      1S:
1223      .SBTTL TYPE ROUTINE
1224
1225      ;;*****↑B*****
1226      ;;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1227      ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1228      ;;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1229      ;;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1230      ;;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1231      ;;
1232      ;;CALL:
1233      ;;*1) USING A TRAP INSTRUCTION
1234      ;;* TYPE ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1235      ;;*OR
1236      ;;* TYPE
1237      ;;* MESADR
1238      ;;*
1239 005162 105737 001177      $TYPE: TSTB $TPFLG  ;; IS THERE A TERMINAL?
1240 005166 100002      BPL  1S        ;; BR IF YES
1241 005170 000000      HALT          ;; HALT HERE IF NO TERMINAL
1242 005172 000430      BR          ;; LEAVE
1243 005174 010046      1S:  MOV  RO,-(SP)  ;; SAVE RO
1244 005176 017600 000002      MOV  @2(SP),RO  ;; GET ADDRESS OF ASCIZ STRING
1245 005202 122737 000001 001254      CMPB #APTENV,$ENV  ;; RUNNING IN APT MODE
1246 005210 001011      BNE  62S      ;; NO, GO CHECK FOR APT CONSOLE
1247 005212 132737 000100 001255      BITB #APTPOOL,$ENVM ;; SPOOL MESSAGE TO APT

```

```

1248 005220 001405          BEQ      62$      ;; NO GO CHECK FOR CONSOLE
1249 005222 010037 005232    MOV      RO,61$   ;; SETUP MESSAGE ADDRESS FOR APT
1250 005226 004737 005452    JSR     PC,$ATY3  ;; SPOOL MESSAGE TO APT
1251 005232 000000          .WORD   0          ;; MESSAGE ADDRESS
1252 005234 132737 000040 001255 61$:   BITB    #APTCSUP,$ENVM ;; APT CONSOLE SUPPRESSED
1253 005242 001003          BNE     60$      ;; YES, SKIP TYPE OUT
1254 005244 112046          MOVB   (RO)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
1255 005246 001005          BNE     4$       ;; BR IF IT ISN'T THE TERMINATOR
1256 005250 005726          TST    (SP)+     ;; IF TERMINATOR POP IT OFF THE STACK
1257 005252 012600          MOV    (SP)+,RO  ;; RESTORE RO
1258 005254 062716 000002    3$:    ADD    #2,(SP) ;; ADJUST RETURN PC
1259 005260 000002          RTI                    ;; RETURN
1260 005262 122716 000011    4$:    CMPB   #HT,(SP) ;; BRANCH IF <HT>
1261 005266 001430          BEQ     8$       ;;
1262 005270 122716 000200    CMPB   #CRLF,(SP) ;; BRANCH IF NOT <CRLF>
1263 005274 001006          BNE     5$       ;;
1264 005276 005726          TST    (SP)+     ;; POP <CR><LF> EQUIV
1265 005300 104402          TYPE                    ;; TYPE A CR AND LF
1266 005302 001231          $CRLF
1267 005304 105037 005440    CLRB   $CHARCNT   ;; CLEAR CHARACTER COUNT
1268 005310 000755          BR     2$       ;; GET NEXT CHARACTER
1269 005312 004737 005374    5$:    JSR    PC,$TYPEC ;; GO TYPE THIS CHARACTER
1270 005316 123726 001176    6$:    CMPB   $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
1271 005322 001350          BNE     2$       ;; IF NO GO GET NEXT CHAR.
1272 005324 013746 001174    MOV    $NULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
1273                                AND THE NULL CHAR.
1274 005330 105366 000001    7$:    DECB   1(SP)    ;; DOES A NULL NEED TO BE TYPED?
1275 005334 002770          BLT    6$       ;; BR IF NO--GO POP THE NULL OFF OF STACK
1276 005336 004737 005374    JSR    PC,$TYPEC ;; GO TYPE A NULL
1277 005342 105337 005440    DECB   $CHARCNT  ;; DO NOT COUNT AS A COUNT
1278 005346 000770          BR     7$       ;; LOOP
1279
1280                                ;HORIZONTAL TAB PROCESSOR
1281
1282 005350 112716 000040    8$:    MOVB   #' (SP)  ;; REPLACE TAB WITH SPACE
1283 005354 004737 005374    9$:    JSR    PC,$TYPEC ;; TYPE A SPACE
1284 005360 132737 000007 005440    BITB   #7,$CHARCNT ;; BRANCH IF NOT AT
1285 005366 001372          BNE     9$       ;; TAB STOP
1286 005370 005726          TST    (SP)+     ;; POP SPACE OFF STACK
1287 005372 000724          BR     2$       ;; GET NEXT CHARACTER
1288 005374 105777 173570    $TYPEC: TSTB   2$TPS   ;; WAIT UNTIL PRINTER IS READY
1289 005400 100375          BPL    $TYPEC
1290 005402 116677 000002 173562    MOVB   2(SP),2$TPB ;; LOAD CHAR TO BE TYPE+BD INTO DATA REG.
1291 005410 122766 000015 000002    CMPB   #CR,2(SP)  ;; IS CHARACTER A CARRIAGE RETURN?
1292 005416 001003          BNE     1$       ;; BRANCH IF NO
1293 005420 105037 005440    CLRB   $CHARCNT  ;; YES--CLEAR CHARACTER COUNT
1294 005424 000406          BR     $TYPEX   ;; EXIT
1295 005426 122766 000012 000002 1$:    CMPB   #LF,2(SP) ;; IS CHARACTER A LINE FEED?
1296 005434 001402          BEQ    $TYPEX   ;; BRANCH IF YES
1297 005436 105227          INCB   (PC)+    ;; COUNT THE CHARACTER
1298 005440 000000          $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
1299 005442 000207          $TYPEX: RTS    PC

```

1300
1301
1302
1303 ↑B .SBTTL APT COMMUNICATIONS ROUTINE

;;*****

```

1304 005444 112737 000001 005710 SATY1:  MOVB  #1, $FFLG      ;; TO REPORT FATAL ERROR
1305 005452 112737 000001 005706 SATY3:  MOVB  #1, $MFLG      ;; TO TYPE A MESSAGE
1306 005460 000403                BR      SATYC
1307 005462 112737 000001 005710 SATY4:  MOVB  #1, $FFLG      ;; TO ONLY REPORT FATAL ERROR
1308 005470                SATYC:
1309 005470 010046                MOV      RO, -(SP)      ;; PUSH RO ON STACK
1310 005472 010146                MOV      R1, -(SP)      ;; PUSH R1 ON STACK
1311 005474 105737 005706                TSTB    $MFLG          ;; SHOULD TYPE A MESSAGE?
1312 005500 001450                BEQ     55              ;; IF NOT: BR
1313 005502 122737 000001 001254                CMPB    #APTENV, $ENV  ;; OPERATING UNDER APT?
1314 005510 001031                BNE     35              ;; IF NOT: BR
1315 005512 132737 000100 001255                BITB    #APTPOOL, $ENVM ;; SHOULD SPOOL MESSAGES?
1316 005520 001425                BEQ     35              ;; IF NOT: BR
1317 005522 017600 000004                MOV      #4(SP), RO    ;; GET MESSAGE ADDR.
1318 005526 062766 000002 000004                ADD      #2, 4(SP)     ;; BUMP RETURN ADDR.
1319 005534 005737 001234                15:    TST      $MSGTYPE     ;; SEE IF DONE W/ LAST XMISSION?
1320 005540 001375                BNE     15              ;; IF NOT: WAIT
1321 005542 010037 001250                MOV      RO, $MSGAD    ;; PUT ADDR IN MAILBOX
1322 005546 105720                25:    TSTB    (RO)+       ;; FIND END OF MESSAGE
1323 005550 001376                BNE     25
1324 005552 163700 001250                SUB      $MSGAD, RO    ;; SUB START OF MESSAGE
1325 005556 006200                ASR      RO            ;; GET MESSAGE LNTH IN WORDS
1326 005560 010037 001252                MOV      RO, $MSGLEN   ;; PUT LENGTH IN MAILBOX
1327 005564 012737 000004 001234                MOV      #4, $MSGTYPE ;; TELL APT TO TAKE MSG.
1328 005572 000413                BR      55
1329 005574 017637 000004 005620 35:    MOV      #4(SP), 45    ;; PUT MSG ADDR IN JSR LINKAGE
1330 005602 062766 000002 000004                ADD      #2, 4(SP)     ;; BUMP RETURN ADDRESS
1331 005610 013746 177776                MOV      177776, -(SP) 1B ;; PUSH 177776 ON STACK
1332 005614 004737 005162                JSR     PC, $TYPE     ;; CALL TYPE MACRO
1333 005620 000000                45:    .WORD  0
1334 005622                55:
1335 005622 105737 005710                105:   TSTB    $FFLG          ;; SHOULD REPORT FATAL ERROR?
1336 005626 001416                BEQ     125            ;; IF NOT: BR
1337 005630 005737 001254                TST     $ENV           ;; RUNNING UNDER APT?
1338 005634 001413                BEQ     125            ;; IF NOT: BR
1339 005636 005737 001234                115:   TST     $MSGTYPE     ;; FINISHED LAST MESSAGE?
1340 005642 001375                BNE     115           ;; IF NOT: WAIT
1341 005644 017637 000004 001236                MOV      #4(SP), $FATAL ;; GET ERROR #
1342 005652 062766 000002 000004                ADD      #2, 4(SP)     ;; BUMP RETURN ADDR.
1343 005660 005237 001234                INC     $MSGTYPE     ;; TELL APT TO TAKE ERROR
1344 005664 105037 005710                125:   CLRB   $FFLG          ;; CLEAR FATAL FLAG
1345 005670 105037 005707                CLRB   $LFLG          ;; CLEAR LOG FLAG
1346 005674 105037 005706                CLRB   $MFLG          ;; CLEAR MESSAGE FLAG
1347 005700 012601                MOV      (SP)+, R1    ;; POP STACK INTO R1
1348 005702 012600                MOV      (SP)+, RO    ;; POP STACK INTO RO
1349 005704 000207                RTS     PC            ;; RETURN
1350 005706 000                $MFLG: .BYTE  0      ;; MESSG. FLAG
1351 005707 000                $LFLG: .BYTE  0      ;; LOG FLAG
1352 005710 000                $FFLG: .BYTE  0      ;; FATAL FLAG
1353                005712                .EVEN
1354                000200                APTSIZE=200
1355                000001                APTENV=001
1356                000100                APTPOOL=100
1357                000040                APTCSUP=040
1358
1359                ;STRING INPUT ROUTINE
    
```

```

1360
1361
1362 005712 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON STACK
1363 005714 010446 MOV R4,-(SP) ;SAVE R4 ON STACK
1364 005716 017637 000004 005734 MOV @4(SP),MSG ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
1365 005724 062766 000002 000004 ADD @2,4(SP) ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
1366 005732 104402 .INST1+8: TYPE ;PRINT THE MESSAGE
1367 005734 000000 .MSG: 0 ;MESSAGE IS POINTED TO FROM HERE
1368 005736 012704 010502 MOV #INBUF,R4 ;POINT R4 TO THE INPUT BUFFER
1369 005742 012703 000007 MOV #7,R3 ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
1370 005746 105777 173212 1S: TSTB @5TKS ;HAS A CHARACTER BEEN RECEIVED?
1371 005752 100375 BPL 1S ;IF NO, KEEP WAITING FOR IT
1372 005754 117714 173206 MOVB @5TKB,(R4) ;IF YES, SAVE IT IN THE INPUT BUFFER
1373 005760 142714 000200 BICB #200,(R4) ;KEEP ONLY THE 7-BIT ASCII INFORMATION
1374 005764 122427 000015 CMPB (R4),#15 ;IS THIS CHARACTER A LINE FEED?
1375 005770 001417 BEQ INSTR2 ;IF SO, TERMINATE THE INPUT SEQUENCE
1376 005772 105777 173172 2S: TSTB @5TPS ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
1377 005776 100375 BPL 2S ;IF WE CAN'T, WAIT UNTIL WE CAN
1378 006000 017777 173162 173164 MOV @5TKB,@5TPB ;ECHO THE CHARACTER BACK
1379 006006 005303 DEC R3 ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
1380 006010 001356 BNE 1S ;IF WE DON'T HAVE 7, GO GET SOME MORE
1381 006012 012604 MOV (SP)+,R4 ;IF WE HAVE 7, RESTORE R4
1382 006014 012603 MOV (SP)+,R3 ;RESTORE R3
1383 006016 010346 .INSTE: MOV R3,-(SP) ;SAVE R3 ON THE STACK
1384 006020 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1385 006022 104402 001230 TYPE ,QUES ;PRINT A QUESTION MARK... WHAT'S GOING ON?
1386 006026 000741 BR .INST1 ;GO PRINT THE MESSAGE AGAIN
1387 006030 012604 INSTR2: MOV (SP)+,R4 ;RESTORE R4
1388 006032 012603 MOV (SP)+,R3 ;RESTORE R3
1389 006034 000002 RTI ;RETURN TO THE MAIN PROCEDURE
1390
1391 ;CONVERT ASCII STRING TO OCTAL
1392
1393
1394 006036 010546 .PARAM: MOV R5,-(SP) ;SAVE R5 ON THE STACK
1395 006040 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1396 006042 016605 000001B4 MOV @4(SP),R5 ;GET THE SETUP INFORMATION POINTER
1397 006046 012537 006226 MOV (R5)+,LOLIM ;SET THE LOW LIMIT FOR THE INPUT
1398 006052 012537 006230 MOV (R5)+,HILIM ;SET THE HIGH LIMIT FOR THE INPUT
1399 006056 012537 006232 MOV (R5)+,DEVAOR ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
1400 006062 112537 006234 MOVB (R5)+,LOBITS ;GET THE MASK OF THE INCORRECT BITS
1401 006066 112537 006235 MOVB (R5)+,ADRCNT ;GET THE COUNT OF ITEMS TO BE STORED
1402 006072 010566 000004 MOV R5,4(SP) ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
1403 006076 005005 PARAM1: CLR R5 ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
1404 006100 012704 010502 MOV #INBUF,R4 ;POINT TO THE INPUT BUFFER
1405 006104 122714 000015 CMPB #15,(R4) ;IS THIS CHARACTER A CARRIAGE RETURN?
1406 006110 001420 BEQ PARERR ;IF SO, PRINT THE MESSAGE AGAIN
1407 006112 121427 000060 1S: CMPB (R4),#60 ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
1408 006116 002415 BLT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1409 006120 121427 000067 CMPB (R4),#67 ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
1410 006124 003012 BGT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1411 006126 142714 000060 BICB #60,(R4) ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
1412 006132 152405 BISB (R4)+,R5 ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
1413 006134 122714 000015 CMPB #15,(R4) ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
1414 006140 001406 BEQ LIMITS ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
1415 006142 006305 ASL R5 ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
    
```

```

1416 006144 006305          ASL    R5          ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
1417 006146 006305          ASL    R5          ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
;B 1418                      ;NEXT THREE BITS
1419 006150 000760          BR     15          ;GO GET THE NEXT CHARACTER
1420 006152 104404          PARERR: INSTER    ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
1421 006154 000750          BR     PARAM1     ;TRY GETTING THE PARAMETERS AGAIN
1422
1423                      ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
1424
1425                      ;-----
1426 006156 020537 006230    LIMITS: CMP    R5,HILIM ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
1427 006162 101373          BHI   PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
1428 006164 020537 006226    CMP    R5,LOLIM   ;IS THE RESULT LOWER THAN ALLOWED?
1429 006170 103770          BLO   PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
1430 006172 133705 006234    BITB  LOBITS,R5   ;ARE ANY INCORRECT BITS SET IN THE RESULT?
1431 006176 001365          BNE   PARERR      ;IF SO, GO PRINT THE MESSAGE AGAIN
1432
1433                      ;STORE NUMBER AT SPECIFIED ADDRESS
1434
1435 006200 013704 006232    15:    MOV    DEVADR,R4 ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
1436 006204 010524          MOV    R5,(R4)+  ;STORE THE RESULT
1437 006206 062705 000002    ADD    #2,R5      ;CALCULATE THE NEXT DATUM
1438 006212 105337 006235    DECB  ADRCNT      ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
1439 006216 001372          BNE   15          ;IF NOT, 1BGO STORE THE NEXT DATUM
1440 006220 012604          MOV    (SP)+,R4   ;RESTORE R4
1441 006222 012605          MOV    (SP)+,R5   ;RESTORE R5
1442 006224 000002          RTI              ;RETURN TO THE MAIN PROGRAM
1443
1444 006226 000000          LOLIM: 0          ;LOWEST ACCEPTABLE VALUE
1445 006230 000000          HILIM: 0          ;HIGHEST ACCEPTABLE
1446 006232 000000          DEVADR: 0         ;LOCATION WHERE RESULT WILL BE STORED
1447 006234 000          LOBITS: .BYTE 0   ;INCORRECT BITS MASK
1448 006235 000          ADRCNT: .BYTE 0   ;COUNT OF ITEMS TO BE STORED
1449
1450                      ;SAVE PC OF TEST THAT FAILED AND R0-R5
1451
1452                      ;-----
1453 18006236 016637 000004 001402 .SAVOS: MOV    4(SP),SAVPC ;SAVE R7 (PC)
1454
1455                      ;SAVE R0-R5
1456
1457 006244 010537 001214    SVOS:  MOV    R5,$REG5 ;SAVE R5
1458 006250 010437 001212    MOV    R4,$REG4 ;SAVE R4
1459 006254 010337 001210    MOV    R3,$REG3 ;SAVE R3
1460 006260 010237 001206    MOV    R2,$REG2 ;SAVE R2
1461 006264 010137 001204    MOV    R1,$REG1 ;SAVE R1
1462 006270 010037 001202    MOV    R0,$REG0 ;SAVE R0
1463 006274 000002          RTI              ;LEAVE.
1464
1465                      ;RESTORE R0-R5
1466
1467 006276 013700 001202    .RESC~ MOV    $REG0,R0 ;RESTORE R0
1468 006302 013701 001204    MOV    $REG1,R1 ;RESTORE R1
1469 006306 013702 001206    MOV    $REG2,R2 ;RESTORE R2
1470 006312 013703 001210    MOV    $REG3,R3 ;RESTORE R3
1471 006316 013704 001212    MOV    $REG4,R4 ;RESTORE R4
    
```

```

1472 006322 013705 001214      MOV      $REG5,R5      ;RESTORE R5
1473 006326 000002              RTI                    ;LEAVE
1474
1475                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1476
1477
1478 006330 104402 001231      .CONVR: TYPE          $CRLF          ;PRINT A CARRIAGE RETURN
1479 006334 010046      .CNVRT: MOVB          R0,-(SP)        ;SAVE R0
1480 006336 010146      MOV      R1,-(SP)        ;SAVE R1
1481 006340 010346      MOV      R3,-(SP)        ;SAVE R3
1482 006342 010446      MOV      R4,-(SP)        ;SAVE R4
1483 006344 010546      MOV      R5,-(SP)        ;SAVE R5
1484 006346 017601 000012      MOV      @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
1485 006352 062766 000002 000012      ADD      #2,12(SP)       ;POINT TO WHERE MAIN PROGRAM WILL RESUME
1486 006360 012137 006504      MOV      (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
1487 006364 112105      1$:     MOVVB        (R1)+,R5      ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
1488 006366 112100      MOVVB        (R1)+,R0      ;GET THE NUMBER OF SPACES TO PRINT
1489 01806370 013104      MOV      @2(R1)+,R4      ;COPY THE WORD TO BE CONVERTED
1490 006372 110537 006506      MOVVB       R5,CHRCNT    ;COPY THE CHARACTER COUNT
1491 006376 010403      3$:     MOV      R4,R3      ;COPY THE ARGUMENT WORD AGAIN
1492 006400 042703 177770      BIC      #1C(7),R3      ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
1493 006404 062703 000060      ADD      #060,R3        ;MAKE AN ASCII CHARACTER OUT OF THEM
1494 006410 110346      MOVVB       R3,-(SP)     ;SAVE THAT CHARACTER
1495 006412 006004      ROR      R4            ;MOVE THE NEXT THREE BITS INTO PLACE
1496 006414 006204      ASR      R4            ;MOVE THEM AGAIN
1497 006416 006204      ASR      R4            ;AND FINALLY A THIRD TIME
1498 006420 005305      DEC      R5            ;REDUCE CHARACTER COUNT. ARE ALL CHARACTERS
1499                          ;BUILT?
1500 006422 001365      BNE      3$           ;IF NO, GO BUILD THE NEXT ONE.
1501 006424 012703 010606      MOV      #MDATA,R3      ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
1502 006430 112623      4$:     MOVVB        (SP)+,(R3)+      ;STORE THE CHARACTER, STARTING WITH THE MOST
1503 006432 105337 006506      DECB     CHRCNT        ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
1504 006436 001374      BNE      4$           ;IF NO, GO TRANSFER ANOTHER
1505 006440 105700      TSTB     R0            ;ARE ANY SPACES TO BE PRINTED?
1506 006442 001404      BEQ      6$           ;IF NO, DON'T SET UP ANY
1507 006444 112723 000040      5$:     MOVVB        #01840,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
1508 006450 105300      DECB     R0            ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
1509 006452 001374      BNE      5$           ;IF YES, GO ADD ANOTHER SPACE
1510 006454 105013      6$:     CLRB      (R3)        ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
1511 006456 104402 010606      TYPE     ,MDATA        ;PRINT THE STRING WE JUST BUILT
1512 006462 005337 006504      DEC      WRDCNT        ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
1513 006466 001336      BNE      1$           ;IF YES, GO CONVERT THEM
1514 006470 012605      MOV      (SP)+,R5      ;RESTORE R5
1515 006472 012604      MOV      (SP)+,R4      ;RESTORE R4
1516 006474 012603      MOV      (SP)+,R3      ;RESTORE R3
1517 006476 006476 012601      MOV      (SP)+,R1      ;RESTORE R1
1518 006500 012600      MOV      (SP)+,R0      ;RESTORE R0
1519 006502 000002      RTI                    ;RETURN TO THE MAIN PROGRAM
1520 006504 000000      WRDCNT: 0
1521 006506 000      CHRCNT: .BYTE
1522 006507 000      SPACNT: .BYTE 0
1523
1524 006510 000000      BINWRD: 0
1525
1526
1527                          ;TRAP DISPATCH SERVICE
    
```

```

1528                                     ; ARGUMENT OF TRAP IS EXTRACTED
1529                                     ; AND USED AS OFFSET TO OBTAIN POINTER
1530                                     ; TO SELECTED SUBROUTINE
1531
1532 006512 010046 .TRPSR: MOV      RO, -(SP)      ; SAVE RO. USE RO TO FIND TRAP ROUTINE
1533 006514 016600 000002 MOV      2(SP), RO      ; GET TRAP ADDRESS
1534 006520 005740 TST      -(RO)          ; GET TRAP
1535 006522 111000 MOV      (RO), RO       ; GET RIGHT BYTE OF TRAP (TRAP OFFSET)
1536 006524 006300 ASL      RO             ; POSITION OFFSET FOR TABLE INDEXING
1537 006526 016000 002002 MOV      .TRPTAB(RO), RO ; PLACE INDEXED ADDRESS OF TABLE IN RO
1538 006532 000200 RTS      RO             ; TRANSFER TO THAT ADDRESS AND RESTORE OLD RO
1539
1540                                     ; DEVICE CLEAR ROUTINE
1541                                     ; ISSUE A DEVICE CLEAR
1542                                     ;-----↑B-----
1543 006534 .DEVICE.CLR:
1544 006534 052777 000020 173300 BIS      #DCLR, @DZCSR   ; SET DCLR
1545 006542 032777 000020 173272 1$: BIT      #DCLR, @DZCSR   ; DID IT CLEAR?
1546 006550 001374 BNE      1$            ; BR IF NO
1547 006552 000002 RTI                      ; EXIT ROUTINE
1548
1549                                     ; ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
1550                                     ;-----↑B-----
1551 006554 104413 .DCLASM: DEVICE.CLR      ; ISSUE A DEVICE CLEAR
1552 006556 153777 001417 173256 BIS      MNTFLG, @DZCSR ; LOAD THE MAINTENANCE BIT IF IT IS I MODE
1553 006564 000002 RTI                      ; RETURN TO CALLING ROUTINE
1554
1555 .DELAY:
1556 006566 010046 MOV      RO, -(SP)      ; SAVE RO
1557 006570 013700 006604 MOV      DLYCNT, RO     ; SET COUNT
1558 006574 005300 1$: DEC      RO             ; DELAY
1559 006576 001376 BNE      1$            ;
1560 006600 012600 MOV      (SP)+, RO      ; RESTORE RO
1561 006602 000002 RTI                      ; LEAVE ROUTINE
1562 006604 000001 DLYCNT: .WORD      1    ; PATCHABLE LOC FOR MORE TIME
1563
1564                                     ; ADVANCE TO NEXT TEST HANDLER
1565                                     ;-----↑B-----
1566
1567 006606 013716 001360 .ADVANCE: MOV      NEXT, (SP) ; CRUNCH STACK WITH ADDRESS OF SCOPE CALL
1568 006612 005037 001362 CLR      LOCK          ; RESET TIGHT LOOP ADDRESS
1569 006616 000002 RTI                      ; CHECK TO SEE IF OLD TEST GETS REPEATED
1570
1571                                     ; ERROR HANDLER
1572                                     ;-----↑B-----
1573
1574 006620 004737 007242 $ERROR: JSR      PC, SERV.G   ; FIND OUT IF (↑G) WAS HIT
1575 006624 032777 010000 172326 BIT      #SW12, @SWR    ; BELL ON ERROR?
1576 006632 001406 BEQ      XBX          ; BR IF NO BELL
1577 006634 105777 172330 TST      @STPS        ; TTY READY.
1578 006640 100003 BPL      XBX          ; DON'T WAIT IF TTY NOT READY.
1579 006642 112777 000207 172322 MOV      #207, @STPB   ; PUSH A BELL AT THE TTY.
1580 006650 032777 020000 172302 XBX: BIT      #SW13, @SWR    ; DELETE ERROR PRINT OUT?
1581 ↑B 006656 001113 BNE      HALTS        ; BR IF NO PRINT OUT WANTED.
1582 006660 021637 001136 CMP      (SP), $ERRPC  ; WAS THIS ERROR FOUND LAST TIME?
1583 006664 001404 BEQ      1$            ; BR IF YES
    
```

1584	006666	011637	001136		MOV	(SP), \$ERRPC		; RECORD BEING HERE
1585	006672	105037	001123		CLRB	\$SERFLG		; PREPARE HEADER
1586	006676	104407		1\$:	SAV05			; SAVE ALL PROC REGISTERS
1587	006700	011605			MOV	(SP), R5		; GET THE PC OF ERROR
1588	006702	162705	000002		SUB	#2, R5		; GET ADDRESS OF TRAP CALL
1589	006706	011504			↑BMOV	(R5), R4		; GET ERROR INSTRUCTION
1590	006710	110437	001134		MOVB	R4, \$ITEMB		; COPY TEST NUMBER FOR APT HANDLING
1591	006714	006304			ASL	R4		; MULT BY TWO
1592	006716	061504			ADD	(R5), R4		; DOUBLE IT
1593	006720	006304			ASL	R4		; MULT AGAIN
1594	006722	042704	177001		BIC	#177001, R4		; CLEAR JUNK
1595	006726	062704	026222		ADD	#.ERRTAB, R4		; GET POINTER
1596	006732	012437	007056		MOV	(R4)+, ERRMSG		; GET ERROR MESSAGE
1597	006736	012437	007070		MOV	(R4)+, DATAHD		; GET DATA HEADER
1598	006742	011437	007102		MOV	(R4), DATABP		; GET DATA TABLE
1599	006746	105737	001123		TSTB	\$SERFLG		; TYPE HEADER
1600	006752	001403			BEQ	TYPMSG		; BR IF YES
1601	006754	005737	007102		TST	DATABP		; DOES DATA TABLE EXIST?
1602	006760	001044			BNE	TYPDAT		; BR IF YES.
1603	006762	104402	001231	TYPMSG:	TYPE	, \$CRLF		; TYPE A CARRIAGE RETURN
1604	006766	104402	001231		TYPE	, \$CRLF		; AND TYPE ANOTHER
1605	006772	005737	001362		TST	LOCK		
1606	006776	001402			BEQ	1\$		
1607	007000	104402	010177		TYPE	, MASTEK		
1608	007004	104402	010165	1\$:	TYPE	, MTSTN		
1609	007010	104412	007234		CNVRT	, XTSTN		; SHOW IT
1610	007014	104402	010254		TYPE	, MERRPC		; TYPE PC.
1611	007020	104412	007226		CNVRT	, ERRTAB0		; SHOW IT
1612	007024	104402	01018127		TYPE	, MCSRX		
1613	007030	104412	004624		CNVRT	, XCSR		
1614	007034	104402	001231		TYPE	, \$CRLF		; GIVE A CR/LF
1615	007040	112737	177777	001123	MOVB	#-1, \$SERFLG		; NO MORE HEADER UNLESS NO DATA TABLE.
1616	007046	005737	007056		TST	ERRMSG		; IS THERE AN ERROR MESSAGE?
1617	007052	001402			BEQ	WTBS.FM		; BR IF NO.
1618	007054	104402			TYPE			; TYPE
1619	007056	000000		ERRMSG:	0			ERROR MESSAGE
1620	007060			WTBS.FM:				
1621	007060	005737	007070		TST	DATABP		; DATA HEADER?
1622	007064	001402			BEQ	TYPDAT		; BR IF NO
1623	007066	104402			TYPE			; TYPE
1624	007070	000000		DATABP:	0			DATABP HEADER
1625	007072	005737	007102	TYPDAT:	TST	DATABP		; DATA TABLE?
1626	007076	001402			BEQ	RESREG		; BR IF NO.
1627	007100	104411			CNVRT			; SHOW
1628	007102	000000		DATABP:	0			DATA TABLE
1629	007104	104410		RESREG:	RES05			; RESTORE PROC REGISTERS
1630	007106	122737	000001	001254	HALTS:	CMPB	#APTENV, \$ENV	; IS APT RUNNING?
1631	007114	001007			BNE	2\$; SKIP APT CALL IF NOT
1632	007116	113737	001134	007130	MOVB	\$ITEMB, 7\$; COPY ERROR NUMBER
1633	007124	004737	005462		JSR	PC, \$ATY4		; CALL APT SERVICE
1634	007130	000000		7\$:	.WORD	0		; ERROR NUMBER STUCK HERE
1635	007132	000777		8\$:	BR	8\$; LOCK UP HERE
1636	007134	022737	004610	000042	2\$:	CMP	#SENDAD, 2#42	; CHECK TO SEE IF IN ACT-11 MODE
1637	007142	001403			BEQ	1\$; IF SO, HANDLE ACCORDINGLY
1638	007144	005777	172010		TST	\$SWR		; HALT ON ERROR?
1639	007150	100004			BPL	EXITER		; BR IF NO HALT ON ERROR

1640	007152	016677	000002	172002	15:	MOV	2(SP),@DISPLAY	:SHOW ERROR PC IN DATA DISPLAY
1641	007160	000000				HALT		:HALT
1642	007162	005237	001132		EXITER:	INC	@EPTTL	:UPDATE ERROR COUNT
1643	007166	032777	000400	171764		BIT	@1BSW08,@SWR	:GOTO TOP OF TEST?
1644	007174	001007				BNE	15	:BR IF YES
1645	007176	032777	002000	171754		BIT	@SW10,@SWR	:GOTO NEXT TEST?
1646	007204	001407				BEQ	25	:BR IF NO
1647	007206	013737	00136C	001126		MOV	NEXT,@LPADR	:SET FOR NEXT TEST
1648	007214	012706	001120		15:	MOV	@STACK,SP	:RESET SP
1649	007220	000177	171702			JMP	@SLPADR	:GOTO SPECIFIED TEST
1650	007224	000002			25:	RTI		:RETURN
1651	007226	000001			ERTABO:	1		
1652	007230	006	002			.BYTE	6,2	
1653	007232	001402				SAVPC		
1654	007234	000001			XTSTN:	1		
1655	007236	002	002			.BYTE	2,2	
1656	007240	001122				\$STNM		
1657	007242	022737	177570	001160	SERV.G:	CMP	@177570,@SWR	:IS THE SWITCH REGISTER HARDWIRED?
1658	007250	001513				BEQ	65	:IF SO, IGNORE ↑G
1659	007252	017746	171710			MOV	@STKB,-(SP)	:OTHERWISE, GET THE LAST CHARACTER TYPED
1660	007256	042716	000200			BIC	@BIT7,(SP)	:STRIP PARITY(EIGHTH) BIT
1661	007262	122726	000007			CMPB	@7,(SP)+	:IS IT ↑G?
1662	007266	001104				BNE	65	:IF NOT, IGNORE INPUT
1663	007270	032777	004000	171666		BIT	@4000,@STKS	:RX BUSY?
1664	007276	001361				BNE	SERV.G	:BR IF YES
1665	007300	017737	171654	007522		MOV	@SWR,905	:SAVE (SWR).
1666	007306	013777	007522	171644	15:	MOV	905,@SWR	
1667	007314	104402	007502			TYPE	.895	
1668	007320	104412	007514			CNVRT	.885	
1669	007324	104402	007524			TYPE	.915	
1670	007330	105777	171630			TSTB	@STKS	:WAIT FOR DONE.
1671	007334	100375				BPL	.-4	
1672	007336	017746	171624			MOV	@STKB,-(SP)	
1673	007342	042716	000200			BIC	@BIT7,(SP)	
1674	007346	122726	000015			CMPB	@15,(SP)+	
1675	007352	001450				BEQ	55	
1676	007354	005077	171600			CLR	@SWR	
1677	007360	105777	171604		25:	TSTB	@STPS	
1678	007364	100375				BPL	.-4	
1679	007366	016677	177776	171576		MOV	-2(SP),@STPB	
1680	007374	000241				CLC		
1681	007376	006177	↑8171556			ROL	@SWR	
1682	007402	006177	171552			ROL	@SWR	
1683	007406	006177	171546			ROL	@SWR	
1684	007412	103735				BCS	15	:ERROR
1685	007414	026627	177776	000060		CMP	-2(SP),#60	
1686	007422	002731				BLT	15	
1687	007424	026627	177776	000067		CMP	-2(SP),#67	
1688	007432	003325				BGT	15	
1689	007434	042766	177770	177776		BIC	@C(7),-2(SP)	
1690	007442	056677	177776	171510		BIS	-2(SP),@SWR	
1691	007450	105777	171510			TSTB	@STKS	
1692	007454	100375				BPL	.-4	
1693	007456	017746	171504			MOV	@STKB,-(SP)	
1694	007462	042716	000200			BIC	@BIT7,(SP)	
1695	007466	122726	000015	↑8		CMPB	@15,(SP)+	

```

1696 007472 001332          BNE      25          :
1697 007474 104402 001231 55:      TYPE      $CRLF  :
1698 007500 000207          65:      RTS       PC      :
1699
1700 007502 020200 051450 051127 89$:      .ASCIZ  <200>? (SWR)=/?
1701 007510 036451 000057
1702          .EVEN
1703 007514 000001          88$:      1
1704 007516          006      000      .BYTE  6,0
1705 007520 007522          90$:
1706 007522 000000          91$:      .WORD  0
1707 007524 036457 000057 91$:      .ASCIZ  ?/=/?
1708          .EVEN
1709          .SBTTL  POWER DOWN AND UP ROUTINES
1710
1711          ;*****
1712          ;POWER DOWN ROUTINE
1713 007530 012737 007674 000024 $PWRDN: MOV    $SILLUP, @PWRVEC ;:SET FOR FAST UP
1714 007536 012737 000340 000026      MOV    #340, @PWRVEC+2 ;:PRIO:7
1715 007544 010046          MOV    R0, -(SP) ;:PUSH R0 ON STACK
1716 007546 010146          MOV    R1, -(SP) ;:PUSH R1 ON STACK
1717 007550 010246          MOV    R2, -(SP) ;:PUSH R2 ON STACK
1718 007552 010346          MOV    R3, -(SP) ;:PUSH R3 ON STACK
1719 007554 010446          MOV    R4, -(SP) ;:PUSH R4 ON STACK
1720 007556 010546          MOV    R5, -(SP) ;:PUSH R5 ON STACK
1721 007560 017746 171374      MOV    @SWR, -(SP) ;:PUSH @SWR ON STACK
1722 007564 010637 007700      MOV    SP, $SAVR6 ;:SAVE SP
1723 007570 012737 007602 000024      MOV    $PWRUP, @PWRVEC ;:SET UP VECTOR
1724 007576 000000          HALT
1725 007600 000776          BR     .-2          ;;HANG UP
1726
1727          ;*****
1728          ;POWER UP ROUTINE
1729 007602 012737 007674 000024 $PWRUP: MOV    $SILLUP, @PWRVEC ;:SET FOR FAST DOWN
1730 007610 013706 007700      MOV    $SAVR6, SP ;:GET SP
1731 007614 005037 007700      CLR    $SAVR6 ;:WAIT LOOP FOR THE TTY
1732 007620 005237 007700      1$:    INC    $SAVR6 ;:WAIT FOR THE INC
1733 007624      18001375      BNE    1$          ;:OF WORD
1734 007626 012677 171326      MOV    (SP)+, @SWR ;:POP STACK INTO @SWR
1735 007632 012605          MOV    (SP)+, R5 ;:POP STACK INTO R5
1736 007634 012604          MOV    (SP)+, R4 ;:POP STACK INTO R4
1737 007636 012603          MOV    (SP)+, R3 ;:POP STACK INTO R3
1738 007640 012602          MOV    (SP)+, R2 ;:POP STACK INTO R2
1739 007642 012601          MOV    (SP)+, R1 ;:POP STACK INTO R1
1740 007644 012600          MOV    (SP)+, R0 ;:POP STACK INTO R0
1741 007646 012737 007530 000024      MOV    $PWRDN, @PWRVEC ;:SET UP THE POWER DOWN VECTOR
1742 007654 012737 000340 000026      MOV    #340, @PWRVEC+2 ;:PRIO:7
1743 007662 104402          TYPE ;:REPORT THE POWER FAILURE
1744 007664 007702          SPWRMG: .WORD  MPFAIL ;:POWER FAIL MESSAGE POINTER
1745 007666 012716          MOV    (PC)+, (SP) ;:RESTART AT RESTART
1746 007670 011304          SPWRAD: .WORD  RESTART ;:RESTART ADDRESS
1747 007672 000002          RTI
1748 007674 000000          $ILLUP: HALT ;:THE POWER UP SEQUENCE WAS STARTED
1749 007676 000776          BR     .-2          ;:BEFORE THE POWER DOWN WAS COMPLETE
1750 007700 000000          $SAVR6: 0 ;:PUT THE SP HERE
1751 007702 050200 051127 043040 MPFAIL: .ASCIZ <200>/PWR FAILED. RESTART AT LAST TEST

```

```

(2) 007745 200 047105 020104 MEPASS: .ASCIZ <200>/END PASS DZDZA-C /
(2) 007771 200 052522 047116 MR: .ASCIZ <200>/RUNNING /
(2) 010005 200 051120 043517 MERR2: .ASCIZ <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 010054 044600 051516 043125 MERR3: .ASCIZ <200>/INSUFFICIENT DATA! /
(2) 010100 046200 041517 020113 MLOCK: .ASCIZ <200>/LOCK ON SELECTED TEST/
(2) 010127 103 051123 020072 MCSRX: .ASCIZ /CSR: /
(2) 010135 126 041505 020072 MVECX: .ASCIZ /VEC: /
(2) 010143 120 051501 042523 MPASSX: .ASCIZ /PASSES: /
(2) 010154 051105 047522 05152182 MERRX: .ASCIZ /ERRORS: /
(2) 010165 124 051505 020124 MTSTN: .ASCIZ /TEST NO: /
(2) 010177 052 000040 MASTEK: .ASCIZ /* /
(2) 010202 051600 052105 051440 MNEW: .ASCIZ <200>/SET SWITCH REG TO DZ11'S DESIRED ACTIVE./
(2) 010254 041520 020072 000 MERRPC: .ASCIZ /PC: /
(2) 010261 200 040515 020120 XHEAD: .ASCIZ <200>/MAP OF DZ11 STATUS/<200>
(2) 010306 044600 046114 043505 MBADLN: .ASCIZ <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>

```

```

(2) 010350 000002
1752 010352 006 003
1753 010354 001220
1754 010356 006 002
1755 010360 001222

```

```

.EVEN
XSTATQ: 2
.BYTE 6,3
$TMP1
.BYTE 6,2
$TMP2

```

.EVEN ; THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN

```

1756
1757
1758
1759
1760
1761
1762 010362 017605 000000 .SETFLG: MOV 2(SP),R5 ; PICK UP ADDRESS OF TAG
1763 010366 042737 000040 010502 BIC #40,INBUF ; STRIP LOWER CASE
1764 010374 122737 000105 010502 CMPB #'E',INBUF ; IS IT EXTERNAL LOOP BACK ?
1765 010402 001005 BNE 4$ ; NO
1766 010404 013715 010474 MOV 1$ (R5) ; YES STORE INFO
1767 010410 10501837 001417 CLRB MNTFLG ; SET MAINT BIT =0
1768 010414 000422 BR 7$ ; GET OUT
1769 010416 122737 000111 010502 4$: CMPB #'I',INBUF ; IS IT INTERNAL LOOP BACK ?
1770 010424 001006 BNE 5$ ; NO
1771 010426 013715 010476 MOV 2$ (R5) ; YES STORE INFO
1772 010432 112737 000010 001417 MOVB #MAINT,MNTFLG ; SET UP THE MAINTENANCE FLAG LOADER
1773 010440 000410 BR 7$ ; GET OUT
1774 010442 122737 000123 010502 5$: CMPB #'S',INBUF ; IS IT STAGGERED LOOP BACK ?
1775 010450 001007 BNE 6$ ; WHAT ?
1776 010452 013715 010500 MOV 3$ (R5) ; YES STORE INFO
1777 010456 105037 001417 CLRB MNTFLG ; ZERO BITS
1778 010462 062716 000002 7$: ADD #2,(SP) ; POP AROUND
1779 010466 000002 RTI
1780 010470 104404 6$: INSTER ; RETRY
1781 010472 000733 BR .SETFLG ; DITTO
1782 010474 000200 1$: .WORD 200 ; EXTERNAL = E
1783 010476 000000 2$: .WORD 0 ; INTERNAL = I
1784 010500 100000 3$: .WORD 100000 ; STAGGERED = S

```

; BUFFERS FOR INPUT-OUTPUT

```

1785
1786
1787
1788 010502 000000 INBUF: 0
1789 010544 010544 . = +40
1790 010544 000000 TEMP: 0

```

1791		010606				. = +40	
1792	010606	000000				MDATA: 0	
1793		010650				. = +40	
1794							
1795	010650	011637	010746			SET.PS: MOV	(SP), 3\$
1796	010654	162737	000002	010746		SUB	#2, 3\$
1797	010662	017737	000060	010750		MOV	23\$, 4\$
1798	010670	022737	106427	010750		CMP	#106427, 4\$
1799	010676	001003				BNE	1\$
1800	010700	011637	010746			MOV	(SP), 3\$
1801	010704	000412				BR	2\$
1802	010706	022737	106437	010750	1\$:	CMP	#106437, 4\$
1803	010714	001401				BEQ	. +4
1804	010716	000000				HALT	; RESERVED INSTRUCTION NOT "MTPS"
1805	010720	011637	010746			MOV	(SP), 3\$
1806	010724	017737	000016	010746		MOV	23\$, 3\$
1807	010732	062716	000002		2\$:	ADD	#2, (5+BP)
1808	010736	017766	000004	000002		MOV	23\$, 2(SP)
1809	010744	000002				RTI	
1810	010746	000000			3\$:	0	
1811	010750	000000			4\$:	0	

```

1812
1813
1814
1815
1816
1817
1818
1819
1820
1821 010752 005737 00141804 CYCLE: TST DZACTV ;ARE ANY DZ11'S TO BE TESTED?
1822 010756 001004 BNE 1$ ;BR IF OK.
1823 010760 104402 010005 TYPE ,MERR2 ;NO DZ11'S SELECTED!!
1824 010764 000000 HALT ;STOP THE SHOW.
1825 010766 000776 BR -2 ;DISQUALIFY CONT. SW.
1826 010770 013737 005116 001226 1$: MOV $MXCNT,$TIMES ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
1827 010776 033737 001406 001404 BIT RUN,DZACTV ;IS THIS ONE "ACTIVE"
1828 011004 001017 BNE 2$ ;BR IF GOOD ONE FOUND.
1829 011006 006137 001406 ROL RUN ;UPDATE POINTER
1830 011012 005537 001406 ADC RUN ;CATCH CARRY FROM RUN
1831 011016 062737 000014 001412 ADD #14,ACTIVE ;UPDATE ADDRESS POINTER.
1832 011024 022737 002000 001412 CMP #DZ.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
1833 011032 001356 BNE 1$ ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
1834 011034 012737 001500 001412 MOV #DZ.MAP,ACTIVE ;RESET ADDRESS POINTER.
1835 011042 000752 BR 1$ ;KEEP LOOKING FOR ACTIVE DZ11
1836 011044 006137 001406 2$: ROL RUN ;UPDATE POINTER.
1837 011050 005537 001406 ADC RUN ;CATCH CARRY.
1838 011054 013700 001412 MOV ACTIVE,RO ;GET ADDRESS POINTER.
1839 011060 062737 000014 001412 ADD #14,ACTIVE ;UPDATE.
1840 011066 022737 002000 001412 CMP#B #DZ.END,ACTIVE
1841
1842 011074 001003 BNE 3$ ;ALL DONE?
1843 011076 012737 001500 001412 MOV #DZ.MAP,ACTIVE ;BR IF NO.
1844 011104 012037 001310 3$: MOV (RO)+,$BASE ;RESTORE POINTER.
1845 011110 012037 002072 MOV (RO)+,DZRIV ;LOAD SYSTEM CTRL. REG
1846 011114 012037 026216 MOV (RO)+,DZPRT ;LOAD VECTOR
1847 011120 113737 026217 001414 MOV#B DZPRT+1,EIAFLG ;LOAD PRIORITY
1848 011126 042737 100000 026216 BIC #BIT15,DZPRT ;EIA OR ZOMA
1849 011134 012037 001364 MOV (RO)+,LINE ;CLEAR FLAG
1850 011140 012037 001366 MOV (RO)+,PAR ;SET UP LINE DZ LINES ACTIVE
+1851 011144 012037 001370 MOV (RO)+,MODE ;SET UP PARAMETERIZATION
1852 011150 004737 026010 JSR PC,DZLEV ;SET UP MAINTENANCE MODE
1853 011154 005737 000042 TST #42 ;SET UP
1854 011160 001046 BNE 4$ ;ARE WE UNDER MONITOR CONTROL?
1855 011162 032777 000002 167770 BIT #SW01,$SWR ;IF YES, SKIP THIS SETUP
1856 011170 001442 BEQ 4$ ;IF SW01=1, GET STARTING TEST #
1857 011172 104402 001231 7$: TYPE ,$CRLF ;BR IF NO TEST IS TO BE INPUTTED
1858 011176 104403 INSTR ;CALL THE STRING INPUT ROUTINE
1859 011200 010165 MTSTN ;POINTER TO MESSAGE TO BE PRINTED
1860 011202 104405 PARAM ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1861 011204 000001 1 ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1862 011206 001000 1000 ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1863 011210 001122 $STNM ;POINTER TO MAP LOCATION TO BE FILLED
1864 011212 000 .BYTE 0 ;MASK OF INVALID BITS FOR THIS PARAMETER
1865 011213 001 .BYTE 1 ;NUMBER OF PARAMETERS TO STORE
1866 011214 012700 012216 MOV #TST1,RO
1867 011220 022710 000004 5$: CMP #4,(RO)
    
```

M05

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 41
 DZDZAC.P11 21-OCT-76 13:07 POWER DOWN AND UP ROUTINES

PAGE: 0064

```

1868 011224 001015          BNE      6$
1869 011226 02276180      012737 000002  CMP      #12737,2(R0)
1870 011234 001011          BNE      6$
1871 011236 023760 001122 000004  CMP      $TSTNM,4(R0) ; IS THIS THE TEST ?
1872 011244 001005          BNE      6$ ; IF NOT, DON'T PROCESS NUMBER
1873 011246 010037 001126  MOV      R0,$LPADR ; SAVE PC
1874 011252 104402 001231  TYPE     $CRLF
1875 011256 000412          BR       8$
1876 011260 005720          6$:     TST      (R0)+
1877 011262 020027 022156  CMP      R0,#TLAST+10
1878 011266 001354          BNE      5$
1879 011270 104402 001230  TYPE     $QUES
1880 011274 000736          BR       7$
1881 011276 012737 012216 001126 4$:     MOV      #TST1,$LPADR ; PREPARE TEST ADDRESS
1882 011304          8$:
1883 011304 01800177      167616  RESTART:JMP  2$LPADR ; GO START TESTING.***WARNING!***
1884                                     ; THIS JUMP IS USED BY POWER UP ROUTINE!!!!
1885
  
```

```

1886 ; -ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
1887 ; IF BIT7 IN THE ENVIRONMENT MODE ($ENVM) BYTE IS SET.
1888 ; THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
1889
1890 011310 012700 001500 SETAPT: MOV #DZ.MAP,R0 ; POINT TO THE DEVICE MAP TABLE
1891 011314 013701 001310 MOV $BASE,R1 ; BUILD DEVICE ADDRESSES IN R1
1892 011320 013702 001304 MOV $VECT1,R2 ; BUILD DEVICE VECTORS IN R2
1893 011324 042702 177007 BIC #C<770>,R2 ; STRIP AWAY OTHER INFORMATION
1894
1895 011330 113703 001305 MOVB $VECT1+1,R3 ; LOAD THE INTERRUPT PRIORITY FROM R3
1896 011334 106003 RORB R3 ; ALIGN THE NUMBER
1897 011336 106003 RORB R3 ; ALIGN THE NUMBER
1898 011340 106003 RORB R3 ; ALIGN THE NUMBER
1899 011342 106003 RORB R3 ; ALIGN THE NUMBER
1900 011344 106003 RORB R3 ; ALIGN THE NUMBER
1901 011346 042703 177770 BIC #C<7>,R3 ; REMOVE ALL BUT BUS LEVEL NUMBER
1902 011352 012704 001320 MOV #SDDWO,R4 ; POINT TO THE BEGINNING OF DEVICE+8 PARAMETERS
1903 011356 013705 001312 MOV $DEVM,R5 ; GET THE MAP OF ACTIVE DEVICES
1904 011362 010537 001404 MOV R5,DZACTV ; SAVE THE BIT MAP
1905 011366 006005 1S: ROR R5 ; GET A DEVICE SELECTION BIT
1906 011370 103407 BCS 3S ; IF IT IS SELECTED, GO SET UP A MAP
1907 011372 001425 BEQ 5S ; IF NO MORE ARE SELECTED, GET OUT OF SETUP
1908 011374 005724 TST (R4)+ ; POINT TO NEXT DEVICE DESCRIPTOR
1909 011376 062701 000010 2S: ADD #10,R1 ; SET UP THE NEXT ADDRESS
1910 011402 062702 000010 ADD #10,R2 ; SET UP THE NEXT VECTOR GROUP
1911 011406 000767 BR 1S ; GO SEE IF MORE DEVICES REMAIN
1912 011410 010120 3S: MOV R1,(R0)+ ; LOAD DEVICE ADDRESS
1913 011412 010220 MOV R2,(R0)+ ; LOAD THE VECTOR ADDRESS
1914 011414 010320 MOV R3,(R0)+ ; LOAD THE INTERRUPT PRIORITY LEVEL
1915 011416 013720 001314 MOV $CDW1,(R0)+ ; GET THE NUMBER OF LINES IN OPERATION
1916 011422 012420 MOV (R4)+,(R0)+ ; LOAD DEVICE PARAMETERS
1917 011424 100406 BMI 4S ; IF 20MA MODE SELECTED, SET IT UP
1918 011426 052760 100000 177772 BIS #100000,-6(R0) ; SET THE 20MA FLAG IN DZLVN
1919 011434 042760 100000 177776 BIC #100000,-2(R0) ; CLEAR THE FLAG IN DZPARN
1920 011442 005020 4S: CLR (R0)+ ; DEFAULT OPERATION TO INTERNAL MAINTENANCE MODE
1921 011444 000754 BR 2S ; GO BUILD THE NEXT ADDRESS
1922 011446 012710 177777 5S: MOV #-1,(R0) ; TERMINATE THE DEVICE MAP
1923 011452 012737 001256 001160 MOV #SSWREG,SWR ; SET TO SOFTWARE APT SWITCH REGISTER
1924 011460 000207 RTS PC ; RETURN TO PRINT STATUS TABLE
1925
1926
1927 ; *ROUTINE USED TO "AUTO SIZE" THE DZ11
1928 ; *CSR AND VECTOR.
1929 ; *NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1930 ; * ADDRESS RANGE (160000:163700)
1931 ; * AND THE VECTOR MAY BE ANY WHERE IN THE
1932 ; * FLOATING VECTOR RANGE (300:770)
1933 ; *
1934
1935 011462 AUTO.SIZE:
1936 011462 000005 RESET ; INSURE A BUS INIT.
1937 011464 105337 001415 DECB INIFLG ; SHOW THAT I WAS HERE
1938 011470 012702 001500 CSRMAP: MOV #DZ.MAP,R2 ; LOAD MAP POINTER.
1939 011474 012703 001320 MOV #SDDWO,R3 ; POINT TO ETABLE DEVICE DESCRIPTOR WORDS
1940 011500 005022 1S: CLR (R2)+ ; ZERO ENTIRE MAP
1941 011502 022702 002000 CMP #DZ.END,R2 ; ALL DONE?
    
```

1942	011506	001374		BNE	15		:BR IF NO
1943	011510	105037	001410	CLRB	DZNUM		:SET OCTAL NUMBER OF DZ11'S TO 0
1944	011514	012702	001500	MOV	#DZ.MAP,R2		
1945	011520	012701	160000	MOV	#160000,R1		:SET FOR FIRST ADDRESS TO BE TESTED
1946	011524	012737	012020	MOV	#65,2#4		:SET FOR NON-EXISTENT DEVICE TIME OUT
1947	011532	052711	000040	BIS	#BITS,(R1)	25:	:TRY TO SET MASTER SCAN ENABLE
1948	011536	052761	000200	BIS	#BIT7,4(R1)		:TRY TO TRANSMIT ON LINE 7
1949	011544	005000		CLR	RO		:USE RO AS A COUNTER
1950	011546	005711		TST	(R1)	75:	:HAS TRANSMITTER READY COME UP?
1951	011550	100403		BMI	85		:IF SO, GO GET A FINAL CHECK
1952	011552	005300		DEC	RO		:REDUCE COUNT. TIME UP?
1953	011554	001374		BNE	75		:IF NOT, KEEP WAITING
1954	011556	000451		BR	35		:ASSUME IT'S NOT A DZ11
1955	011560	032761	000200	BIT	#BIT7,4(R1)	85:	:IS LINE 7 ENABLE STILL SET? IT SHOULD BE
1956	011566	001445		BEQ	35		:IF IT'S NOT, ASSUME IT'S NOT A DZ11
1957	011570	032711	000040	BIT	#BITS,(R1)		:IS MASTER SCAN ENABLE STILL SET?
1958	011574	001442		BEQ	35		:IF NOT, ASSUME IT'S NOT A DZ11
1959	011576	005000		CLR	RO		
1960	011600	052711	000020	BIS	#20,(R1)		:SET DEVICE CLEAR
1961	011604	032711	000020	BIT	#20,(R1)		:SHOULD STAY SET FOR A WHILE IF DZ
1962	011610	001434		BEQ	35		:BR IF NOT +BDZ11
1963	011612	032711	000020	BIT	#20,(R1)		:WAIT FOR BIT TO CLEAR
1964	011616	001404		BEQ	.+12		:BR WHEN CLEARED
1965	011620	104414		DELAY			
1966	011622	005200		INC	RO		
1967	011624	001372		BNE	.-12		
1968	011626	000425		BR	35		:BIT NOT CLEARED! MUST NOT BE DZ11
1969	011630	005011		CLR	(R1)		:GET RID OF MASTER SCAN ENABLE
1970	011632	005061	000004	CLR	4(R1)		:GET RID OF LINE 7 ENABLE
1971							
1972	011636	010122		MOV	R1,(R2)+		:AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZ11 CSR ADDRESS.
1973	011640	005722		TST	(R2)+		:STORE CSR IN CORE TABLE.
1974	011642	+8012722	000005	MOV	#5,(R2)+		:POP OVER VECTOR STORE AREA
1975	011646	012722	000377	MOV	#377,(R2)+		:SET THE DEFAULT BUS LEVEL
1976	011652	012712	017470	MOV	#17470,(R2)		:SET THE DEFAULT LINE SELECTION PARAMETER
1977	011656	012223		MOV	(R2)+,(R3)+		:SET THE DEFAULT PARAMETERS
1978	011660	005022		CLR	(R2)+		:COPY PARAMETERS INTO ETABLE DESCRIPTOR
1979	011662	012712	177777	MOV	#-1,(R2)		:SET THE DEFAULT MODE OF OPERATION
1980	011666	105237	001410	INCB	DZNUM		:TERMINATE LIST
1981	011672	122737	000020	CMPB	#20,DZNUM		:UPDATE DEVICE COUNTER
1982	011700	001405		BEQ	1005		:ARE MAX. NO. OF DEV FOUND?
1983	011702	062701	000010	ADD	#10,R1	35:	:YES DON'T LOOK FOR ANY MORE.
1984	011706	022701	163700	CMP	#163700,R1		:UPDATE CSR POINTER ADDRESS
1985	011712	001307		BNE	25		:BR IF MORE ADDRESS TO CHECK.
1986	011714					1005:	
1987	011714	105737	001410	TSTB	DZNUM		:WERE ANY DZ11'S FOUND AT ALL?
1988	011720	001432		BEQ	55		:ERROR AUTO SIZER FOUND NO DZ11'S IN THIS SYS.
1989	011722	113701	001410	MOVB	DZNUM,R1		
1990	011726	110137	001411	MOVB	R1,SAVNUM		:SAVE NUMBER OF DEVICES
1991	011732	012737	000001	MOV	#1,DZACTV		
1992	011740	005301		DEC	R1	45:	
1993	011742	001404		BEQ	985		
1994	011744	000261		SEC			
1995	011746	+8	006137	ROL	DZACTV		
1996	011752	000772		BR	45		
1997	011754	013737	001500	MOV	DZCRO,\$BASE	985:	:POINT TO THE ADDRESS OF FIRST DEVICE

1998	011762	013737	001512	001314		MOV	MANTO,\$CDW1	:INDICATE TO ETABLE WHAT MODE IS BEING USED
1999	011770	012737	000006	000004	99\$:	MOV	#6,2#4	:RESTORE TRAP VECTOR
2000	011776	013737	001404	001312		MOV	DZACTV,\$DEVN	:SAVE ACTIVE REGISTER
2001	012004	000410				BR	VECMAP	:GO FIND THE VECTOR NOW.
2002	012006	104402	010005		5\$:	TYPE	MERR2	:NOTIFY OPRIB THAT NO DZ11'S FOUND.
2003	012012	005000				CLR	R0	:MAKE DATA DISPLAY ZERO
2004	012014	000000				HALT		:STOP THE SHOW
2005	012016	000776				BR	.-2	:DISABLE CONT. SW.
2006	012020	012716	011702		6\$:	MOV	#3\$,(SP)	:ENTERED BY NON-EXISTENT TIME-OUT
2007	012024	000002				RTI		:RETURN TO MAINSTREAM
2008								
2009	012026	012737	000340	000022	VECMAP:	MOV	#340,2#22	:SET IOT TRAP PRIORITY TO 7
2010	012034	012737	012150	000020		MOV	#4\$,2#20	:SET IOT TRAP VECTOR
2011	012042	012702	001500			MOV	#DZ.MAP,R2	:SET SOFTWARE POINTER
2012	012046	012700	000300			MOV	#300,R0	:FLOATING VECTORS START HERE.
2013	012052	012701	000302			MOV	#302,R1	:PC OF IOT INSTR.
2014	012056	010120			1\$:	MOV	R1,(R0,+	:START FILLING VECTOR AREA
2015	012060	012721	000004			MOV	#4,(R1)+	:WITH .+2; IOT
2016	012064	022021				CMP	(R0)+(R1)+	:ADD 2 TO R0 +R1
2017	012066	020127	001000			CMP	R1,#1000	:HAS THE VECTOR AREA BEEN EXCEEDED?
2018	012072	101771				BLOS	1\$:BR IF MORE TO FILL
2019	012074	013704	001404			MOV	DZACTV,R4	:STORE TEMPORARILY
2020	012100	006004			2\$:	ROR	R4	:BRING OUT A BIT
2021	012102	103036				BCC	5\$:BR IF ALL DONE
2022	012104	106427	000000			MTPS	#0	:ZERO CPU PRIO
2023	012110	012772	040040	000000		MOV	#BIT14+BITS,2(R2)+B	:GET CSR
2024	012116	011201				MOV	(R2),R1	:SET THE TCR BIT!
2025	012120	112761	000200	000004		MOV	#BIT7,4(R1)	:ATTEMPT TO FORCE AN INTERRUPT!
2026								:STALL
2027	012126	005200				INC	R0	
2028	012130	001376				BNE	.-2	:FOR TIME TO INTERRUPT
2029	012132	012762	000300	000002		MOV	#300,2(R2)	:NO INTERRUPT ASSUME 300 AND FIX DZ11 LATER
2030	012140	000005			3\$:	RESET		:INIT
2031	012142	062702	000014			ADD	#14,R2	:POP SOFTWARE POINTER
2032	012146	000754				BR	2\$:KEEP GOING
2033	012150	011662	000002		4\$:	MOV	(SP),2(R2)	:GET VECTOR ADDRESS
2034	012154	162762	000010	000002		SUB	#10,2(R2)	:POINT B+BACK TO THE CORRECT VECTOR
2035	012162	042762	000007	000002		BIC	#7,2(R2)	:CLEAR JUNK
2036	012170	022626				POP2SP		:POP IOT JUNK OFF STACK
2037	012172	012716	012140			MOV	#3\$,(SP)	:SET FOR RETURN
2038	012176	000002				RTI		
2039	012200	013737	001502	001304	5\$:	MOV	DZVCO,\$VECT1	:COPY VECTOR OF FIRST DEVICE INTO ETABLE
2040	012206	012737	004654	000020		MOV	#.SCOPE,IOTVEC	:RESTORE THE SCOPE TRAP
2041	012214	000207				RTS	PC	:ALL DONE WITH "AUTO SIZING"
2042								

```

2043
2044
2045
2046
2047
2048
2049
2050 012216 000004
2051 012220 012737 000001 001122
2052 012226 012737 012406 001360
2053 012234 012737 012374 000004
2054 012242 01182737 000340 000006
2055 012250 012737 012256 001362
2056 012256 013700 002042
2057 012262 011001
2058 012264 000240
2059 012266 005010
2060 012270 000240
2061 012272 012737 012300 001362
2062 012300 013700 002046
2063 012304 011001
2064 012306 000240
2065 011812310 005010
2066 012312 000240
2067 012314 012737 012322 001362
2068 012322 013700 002056
2069 012326 011001
2070 012330 000240
2071 012332 005010
2072 012334 000240
2073 012336 012737 012344 001362
2074 012344 013700 002062
2075 012350 011001
2076 012352 000240
2077 012354 005010
2078 012356 000240
2079 012360 012737 000006 000004
2080 012366 005037 000006
2081 012372 104400
2082 012374 011601
2083 012376 022626
2084 012400 104001
2085 012402 104401
2086 012404 000111
2087
2088
2089 1B
2090
2091
2092
2093 012406 000004
2094 012410 012737 000002 001122
2095 012416 012737 012472 001360
2096 012424 013700 002042
2097 012430 012705 000020
2098 012434 010510

```

```

***** TEST 1 *****
;THIS TEST PROVES THE SLAVE SYNC RESPONSE
;DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
; DZCSR, DZRBUF, DZTCR, DZMSR
::* TEST 1
*****
1ST1: SCOPE
MOV #1, $STNM ;LOAD THE NUMBER OF THIS TEST
MOV #1ST2, NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #55, 4 ;SET TRAP VECTOR
MOV #PR7, 6 ;SET PRIORITY TO LEVEL 7
MOV #15, LOCK ;SET RETURN IF SW09=11
15: MOV DZCSR, R0 ;SET ADDRESS TO TEST
MOV (R0), R1 ;READ THE ADDRESS
NOP ;WASTE TIME
CLR (R0) ;WRITE THE ADDRESS
MOV #25, LOCK ;SET RETURN ADDRESS FOR SW09
25: MOV DZRBUF, R0 ;SET ADDRESS TO TEST
MOV (R0), R1 ;READ THE ADDRESS
NOP ;
CLR (R0) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #35, LOCK ;SET RETURN ADDRESS FOR SW09
35: MOV DZTCR, R0 ;SET ADDRESS TO TEST
MOV (R0), R1 ;READ THE ADDRESS
NOP ;
CLR (R0) ;WRITE THE ADDRESS
MOV #45, LOCK ;SET RETURN ADDRESS
45: MOV DZMSR, R0 ;SET ADDRESS TO TEST
MOV (R0), R1 ;READ FROM ADDRESS
NOP ;
CLR (R0) ;WRITE THE ADDRESS
MOV #6, 4 ;SET TRAP CATCHER BACK TO NORMAL
6
55: ADVANCE ;SCOPE THIS TEST
MOV (SP), R1 ;SAVE PC OF TRAP
CMP (SP)+, (SP)+ ;POP TRAP OFF STACK
ERROR 1 ;NO SLAVE SYNC RESPONSE.
SCOPI ;SW09=1?
JMP (R1) ;RTI
***** TEST 2 *****
;THIS TEST PROVES THAT BIT "DCLR"
;CAN BE SET AND THAT IT WILL CLEAR
;BY ITSELF AFTER A PERIOD OF TIME.
::* TEST 2
*****
1ST2: SCOPE
MOV #2, $STNM ;LOAD THE NUMBER OF THIS TEST
MOV #1ST3, NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR, R0 ;SET POINTER
MOV #DCLR, R5 ;SET DCLR
MOV R5, (R0) ;WRITE DCLR INTO DZCSR

```



```

2155      ::* TEST 4
2156      :*****
2157      012564 000004      000004 001122      †T4: SCOPE
2158      012566 012737      012656 001360      MOV #4,STSTNM      ;LOAD THE NUMBER OF THIS TEST
2159      012574 012737      MOV #TST5,NEXT      ;POINT TO THE START OF THE NEXT TEST
2160      012602 013700      MOV DZCSR,R0      ;GET BASE ADDRESS
2161      012606 012705      MOV #MSENAB,R5      ;SET BIT
2162      012612      MOV R5,(R0)      ;SET SET IN DEVICE
2163      012614 011004      MOV (R0),R4      ;READ THE BIT FROM DEVICE
2164      012616 020504      CMP R5,R4      ;WAS BIT SET?
2165      012620 001401      BEQ 1$      ;BR IF YES
2166      012622 104002      ERROR 2      ;*BIT R/W FAILURE
2167      012624 040510      1$: BIC R5,(R0)      ;CLEAR THE BIT.
2168      012626 011004      MOV (R0),R4      ;READ DEVICE
2169      012630 001404      BEQ 2$      ;BR IF BITS WERE CLEARED.
2170      012632 010546      MOV R5,-(SP)      ;SAVE THE BIT
2171      012634 005005      CLR R5      ;SET EXPECTED RESULTS TO 0
2172      012636 104002      ERROR 2      ;*BIT FAILED TO CLEAR
2173      012640 012605      MOV (SP)+,R5      ;RESTORE THE BIT.
2174      012642 010510      2$: MOV R5,(R0)      ;SET THE BIT AGAIN
2175      012644 104413      DEVICE.CLR      ;ISSUE DEVICE CLEAR
2176      012646 011004      MOV (R0),R4      ;READ THE BIT.
2177      012650 001402      BEQ 3$      ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
2178      012652 005005      CLR R5      ;SET EXPECTED TO ZERO
2179      012654 104002      ERROR 2      ;*BIT NOT CLEARED BY DEVICE CLEAR
2180      012656
2181      :***** TEST 5 *****
2182      ;*TEST TO VERIFY THAT BIT "SILOEN" CAN
2183      ;*BE SET. THEN VERIFY THAT BIT "SILOEN" CAN
2184      ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
2185      ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
2186      ;*CLEARED BY A "DEVICE CLEAR"
2187      ::* TEST 5
2188      :*****
2189      012656 000004      000005 001122      †T5: SCOPE
2190      012660 012737      012750 001360      MOV #5,STSTNM      ;LOAD THE NUMBER OF THIS TEST
2191      012666 012737      MOV #TST6,NEXT      ;POINT TO THE START OF THE NEXT TEST
2192      012674 013700      MOV DZCSR,R0      ;GET BASE ADDRESS
2193      012700 012705      MOV #SILOEN,R5      ;SET BIT
2194      012704 010510      MOV R5,(R0)      ;SET SET IN DEVICE
2195      012706 011004      MOV (R0),R4      ;READ THE BIT FROM DEVICE
2196      012710 020504      CMP R5,R4      ;WAS BIT SET?
2197      012712 001401      BEQ 1$      ;BR IF YES
2198      012714 104002      ERROR 2      ;*BIT R/W FAILURE
2199      012716 040510      1$: BIC R5,(R0)      ;CLEAR THE BIT.
2200      012720 011004      MOV (R0),R4      ;READ DEVICE
2201      012722 001404      BEQ 2$      ;BR IF BITS WERE CLEARED.
2202      012724 010546      MOV R5,-(SP)      ;SAVE THE BIT
2203      012726 005005      CLR R5      ;SET EXPECTED RESULTS TO 0
2204      012730 104002      ERROR 2      ;*BIT FAILED TO CLEAR
2205      012732 012605      MOV (SP)+,R5      ;RESTORE THE BIT.
2206      012734 010510      2$: MOV R5,(R0)      ;SET THE BIT AGAIN
2207      012736 104413      DEVICE.CLR      ;ISSUE DEVICE CLEAR
2208      012740 011004      MOV (R0),R4      ;READ THE BIT.
2209      012742 001402      BEQ 3$      ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
2210      012744 005005      CLR R5      ;SET EXPECTED TO ZERO

```

G06

2211	012746	104002		
2212	012750			
2213				
2214				
2215				
182216				
2217				
2218				
2219				
2220				
2221	012750	000004		
2222	012752	012737	000006	001122
2223	012760	012737	013042	001360
2224	012766	013700	002042	
2225	012772	012705	000100	
2226	012776	010510		
182227		013000	011004	
2228	013002	020504		
2229	013004	001401		
2230	013006	104002		
2231	013010	040510		
2232	013012	011004		
2233	013014	001404		
2234	013016	010546		
2235	013020	005005		
2236	013022	104002		
2237	013024	012605		
2238	013026	010510		
2239	013030	104413		
2240	013032	011004		
2241	013034	001402		
2242	013036	005005		
2243	013040	104002		
2244	013042			
2245				
2246				
2247				
2248				
2249				
2250				
2251				
2252				
2253	013042	000004		
2254	013044	012737	000007	001122
2255	013052	012737	013134	001360
2256	013060	013700	002042	
2257	013064	012705	040000	
2258	013070	010510		
2259	013072	011004		
2260	013074	020504		
2261	013076	001401		
2262	013100	104002		
2263	013102	040510		
2264	013104	011004		
2265	013106	001404		
2266	013110	010546		

```

ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
3$:
;***** TEST 6 *****
;*TEST TO VERIFY THAT BIT "RIE" CAN
;*BE SET. THEN VERIFY THAT BIT "RIE" CAN
;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
;*CLEARED BY A "DEVICE CLEAR"
::* TEST 6
;*****
1$T6: SCOPE
MOV #6,$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #1$T7,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;GET BASE ADDRESS
MOV #RIE,R5 ;SET BIT
MOV R5,(R0) ;SET SET IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 1$ ;BR IF YES
ERROR 2 ;*BIT R/W FAILURE
1$: BIC R5,(R0) ;CLEAR THE BIT.
MOV (R0),R4 ;READ DEVICE
BEQ 2$ ;BR IF BITS WERE CLEARED.
MOV R5,-(SP) ;SAVE THE BIT
CLR R5 ;SET EXPECTED RESULTS TO 0
ERROR 2 ;*BIT FAILED TO CLEAR
MOV (SP)+,R5 ;RESTORE THE BIT.
2$: MOV R5,(R0) ;SET THE BIT AGAIN
DEVICE.CLR ;ISSUE DEVICE CLEAR
MOV (R0),R4 ;READ THE BIT.
BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
CLR R5 ;SET EXPECTED TO ZERO
ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
3$:
;***** TEST 7 *****
;*TEST TO VERIFY THAT BIT "TIE" CAN
;*BE SET. THEN VERIFY THAT BIT "TIE" CAN
;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
;*CLEARED BY A "DEVICE CLEAR"
::* TEST 7
;*****
1$T7: SCOPE
MOV #7,$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #1$T10,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;GET BASE ADDRESS
MOV #TIE,R5 ;SET BIT
MOV R5,(R0) ;SET SET IN DEVICE
MOV (R0),R4 ;READ THE BIT FROM DEVICE
CMP R5,R4 ;WAS BIT SET?
BEQ 1$ ;BR IF YES
ERROR 2 ;*BIT R/W FAILURE
1$: BIC R5,(R0) ;CLEAR THE BIT.
MOV (R0),R4 ;READ DEVICE
BEQ 2$ ;BR IF BITS WERE CLEARED.
MOV R5,-(SP) ;SAVE THE BIT

```

```

2267 013112 005005          CLR      R5          ;SET EXPECTED RESULTS TO 0
2268 013114 104002          ERROR    2          ;*BIT FAILED TO CLEAR
2269 013116 012605          MOV      (SP)+,R5   ;RESTORE THE BIT.
2270 013120 010510          2$:     MOV      R5,(R0) ;SET THE BIT AGAIN
2271 013122 104413          DEVICE.CLR ;ISSUE DEVICE CLEAR
2272 013124 011004          MOV      (R0),R4   ;READ THE BIT.
2273 013126 001402          BEQ     3$         ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR.
2274 013130 005005          CLR      R5          ;SET EXPECTED TO ZERO
2275 013132 104002          ERROR    2          ;*BIT NOT CLEARED BY DEVICE CLEAR
2276 013134
2277
2278 ;***** TEST 10 *****
2279 ;*THIS TESTS THAT ALL OF THE FOLLOWING
2280 ;*BITS CAN BE: SET, CLEARED, CLEARED BY "DEVICE CLEAR "
2281 ;*BITS TESTED ARE:
2282 ;* TCR0, TCR1, TCR2, TCR3, TCR4, TCR5, TCR6, TCR7
2283 ;* TEST 10
2284 ;*****
2285 013134 000004          *B      TST10: SCOPE
2286 013136 012737 000010 001122          MOV      #10,$TSTNM ;LOAD THE NUMBER OF THIS TEST
2287 013144 012737 013272 001360          MOV      #TST11,NEXT ;POINT TO THE START OF THE NEXT TEST
2288 013152 013700 002056          MOV      DZTCR,R0   ;SET DEVICE ADDRESS
2289 013162 012737 000001          MOV      #TCR0,R5   ;SET EXPECTED RESULTS
2290 013170 010510 001362          MOV      #1$ LOCK   ;SET FOR SW09
2291 013172 011004          1$:     MOV      R5,(R0)   ;SET THE BIT
2292 013174 042704 177400          MOV#B    (R0),R4   ;READ THE BIT FROM THE DEVICE
2293 013200 020504          BIC     #1C<377>,R4 ;CLEAR HIGH BYTE
2294 013202 001401          CMP     R5,R4      ;WAS BIT OK?
2295 013204 104002          BEQ     2$         ;BR IF YES
2296 013206 040510          2$:     ERROR    2          ;*BIT FAILED TO SET.
2297 013210 011004          BIC     R5,(R0)   ;CLEAR THE BIT
2298 013212 042704 177400          MOV      (R0),R4   ;READ THE REGISTER
2299 013216 005704          BIC     #1C<377>,R4 ;CLEAR HIGH BYTE
2300 013220 001404          TST     R4        ;BITS CLEAR?
2301 013222 010546          BEQ     3$         ;BR IF YES
2302 013224 005005          MOV      R5,-(SP)  ;SAVE GOOD RESULTS
2303 013226 104002          CLR     R5        ;SET EXPECTED TO 0
2304 013230 012605          ERROR    2          ;*REPORT BIT NOT CLEAR
2305 013232 010510          3$:     MOV      (SP)+,R5 ;RESTORE R5
2306 013234 104413          MOV      R5,(R0)   ;SET THE BIT AGAIN.
2307 013236 011004          DEVICE.CLR ;ISSUE DEVICE CLEAR
2308 013240 042704 177400          MOV      (R0),R4   ;READ THE REGISTER
2309 013244 005704          BIC     #1C<377>,R4 ;CLEAR HIGH BYTE
2310 013246 001404          TST     R4        ;BITS CLEAR?
2311 013250 010546          BEQ     4$         ;BR IF YES
2312 013252 005005          MOV      R5,-(SP)  ;SAVE GOOD RESULTS
2313 013254 104002          CLR     R5        ;SET EXPECTED TO 0
2314 013256 012605          ERROR    2          ;*REPORT BIT NOT CLEAR
2315 013260 104401          *B      4$:     MOV      (SP)+,R5 ;RESTORE R5
2316 013262 106305          SCOPE1 ;LOCK ON BIT? SET SW09=1
2317 013264 001341          ASLB   R5          ;CHANGE TO NEXT BIT
2318 013266 005037 001362          BNE    1$         ;CONTINUE TESTING
2319          CLR     LOCK     ;MAKE SURE TIGHT LOOP IS CLEANED UP
2320 ;***** TEST 11 *****
2321 ;*THIS TESTS THAT ALL OF THE FOLLOWING
2322 ;*BITS CAN BE: SET, CLEARED, CLEARED BY "RESET INSTR *NOT* DEVICE CLEAR "
;*BITS TESTED ARE:

```


JOB

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:07
 DZDZAC.P11 21-OCT-76 13:07

21-OCT-76 13:09 ↑BPAGE 51

DZ11 DEVICE DIAGNOSTICS.

COPYRIGHT 1976 DIGITAL EGLIP. CORP.

PAGE: 0074

```

2379
2380 013446 000004
2381 013450 012737 000012 001122
2382 013456 012737 013564 001360
2383 013464 013700 002042
2384 013470 005005
2385 013472 012710 027607
2386
2387 ↑B013476 011004
2388 C13500 001401
2389 013502 104002
2390 013504 012710 100000
2391 C13510 011004
2392 013512 001401
2393 013514 104002
2394 013516 012705 100000
2395 013522 005077 166324
2396 013526 052777 000001 166322
2397 013534 052710 000040
2398 013540 052705 000040
2399 013544 005002
2400 013546 011004
2401 013550 020504
2402 013552 001404
2403 013554 104414
2404 013556 005202
2405 013560 001372
2406 013562 104002
2407 013564
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417 013564 000004
2418 013566 012737 000013 001122
2419 013574 012737 013650 001360
2420 013602 104413
2421 013604 013700 002042
2422 01361180 012710 177757
2423 013614 012705 050150
2424 013620 011004
2425 013622 020405
2426 013624 001401
2427 013626 104002
2428 013630 012705 000020
2429
2430
2431 013634 052710 000020
2432 013640 011004
2433 013642 020405
2434 013644 001401

:*****
↑ST12: SCOPE
MOV #12,$STNM ;LOAD THE NUMBER OF THIS TEST
MOV #↑ST13,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;SET ADDRESS TO R0
CLR R5 ;SET EXPECTED TO 0
MOV #RDONE+BIT11+BIT10+BIT9+BIT8+BIT2+BIT1+BIT0+SILOAL,(R0) ;WRITE THE BITS
MOV (R0),R4 ;READ BACK THE BITS
BEG 1$ ;BR IF NONE ARE SET.
ERROR 2 ;*BITS WERE SET.
1$: MOV #TRDY,(R0) ;ATTEMPT TO WRITE TRDY
MOV (R0),R4 ;READ TRDY
BEG 2$ ;BR IF NOT SET
ERROR 2 ;*
2$: MOV #TRDY,R5 ;SET EXPECTED BIT
CLR DZLPR ;LOAD LINE 0
BIS #TCRO,DZTCR ;SET TCR BIT
BIS #MSENAB,(R0)
BIS #MSENAB,R5 ;SET SCAN ENABLE
CLR R2 ;SET COUNTER TO ZERO
3$: MOV (R0),R4 ;READ THE REGISTER
CMP R5,R4 ;BIT SET?
BEG 4$ ;BR IF YES
DELAY ;STALL TIME
INC R2 ;UPDATE COUNTER
BNE 3$ ;BR IF COUNTER NOT DONE.
ERROR 2 ;*TRDY NOT SET!
4$:

:***** TEST 13 *****
:THIS TEST PERFORMS RESET TESTING AND
:TESTING OF READ ONLY AND WRITE ONLY BITS
:IN REGISTER DZCSR
:VERIFY THAT "TIE", "SILOEN", "RIE", "MSENAB", "MAINT"
:ARE THE ONLY R/W BITS IN THE DZCSR.
:THEN SET "DCLR" AND VERIFY THEY ARE CLEARED
:
:TEST 13
:*****
↑ST13: SCOPE
MOV #13,$STNM ;LOAD THE NUMBER OF THIS TEST
MOV #↑ST14,NEXT ;POINT TO THE START OF THE NEXT TEST
DEVICE.CLR
MOV DZCSR,R0 ;SET UP FOR ERROR MESSAGE
MOV #↑C<DCLR>,(R0) ;TRY TO WRITE
MOV #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP EXPECTED VS ACTUAL
BEG 1$ ;YES
ERROR 2 ;*NO
1$: MOV #DCLR,R5 ;EXPECTED...NOTE THAT DCLR REMAINS
;SET LONG ENOUGH TO READ IT...HOWEVER
;IF YOU EXAMINE THIS BIT IT SHOULD BE CLEAR.
BIS #DCLR,(R0) ;DEVICE MASTER RESET
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP ACTUAL VS EXPECTED
BEG 2$ ;YES

```

K06

MD-11-DZDZA-C MACY:1 27(1006) 21-OCT-76 13:09 PAGE 52
DZDZAC.P11 21-OCT-76 13:07

DZ11 DEVICE DIAGNOSTICS. COPYRIGHT 1976 DIGITAL EQUIP. CORP.

PAGE: 0075

2435 013646 104002
 2436 013650
 2437
 2438
 2439
 2440
 2441
 2442
 2443 013650 000004
 2444 013652 012737 000014 001122
 2445 013660 012737 013740 001360
 2446 013666 104413
 2447 013670 013700 002046
 2448 013674 012777 177777 166150
 2449 013702 011004
 2450 013704 010405
 2451 013706 042705 104000
 2452 013712 020405
 2453 013714 18001401
 2454 013716 104002
 2455 013720 010403
 2456 013722 005103
 2457 013724 010377 166122
 2458 013730 011004
 2459 013732 020405
 2460 013734 001401
 2461 013736 104002
 2462 013740
 2463
 2464
 2465
 2466
 2467
 2468
 2469 013740 000004
 2470 013742 012737 000015 001122
 2471 013750 012737 014024 001360
 2472 013756 104413
 2473 013760 013700 002062
 2474 013764 012777 177777 166074
 2475 013772 011004
 2476 013774 010405
 2477 013776 020405
 2478 014000 001401
 2479 014002 104002
 2480 014004 010403
 2481 014006 005103
 2482 014010 010377 166052
 2483 014014 011004
 2484 014016 020405
 2485 01418020 001401
 2486 014022 104002
 2487 014024
 2488
 2489
 2490

```

ERROR 2 ;*NO
25:
;***** TEST 14 *****
;THIS TEST PERFORMS RESET TESTING AND
;TESTING OF READ ONLY REGISTER DZRBUF
;AND TESTING OF WRITE ONLY REGISTER DZLPR
;:* TEST 14
;*****
†ST14: SCOPE
MOV #14,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #†ST15,NEXT ;POINT TO THE START OF THE NEXT TEST
DEVICE.CLR ;CLEAR DZ11
MOV DZRBUF,R0 ;SET UP FOR ERROR MESSAGE
MOV #-1,DZLPR ;TRY TO WRITE ALL 1'S
MOV (R0),R4 ;ACTUAL
MOV R4,R5 ;MAKE EXPECTED
BIC #DVALID!BIT11,R5 ;DITTO
CMP R4,R5 ;CMP ACTUAL VS EXPECTED
BEQ 15 ;IF YES,GO CONTINUE PROCESSING
ERROR 2 ;*ERROR- BIT PATTERN NOT CORRECT
15: MOV R4,R3 ;GET A COPY OF THE ACTUAL BIT PATTERN
COM R3 ;GET THE LOGICAL INVERSE OF THE BIT PATTERN
MOV R3,DZLPR ;TRY TO WRITE
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP ACTUAL VS EXPECTED
BEQ 25 ;IF YES, GET OUT OF THIS TEST
ERROR 2 ;*NO
25:
;***** TEST 15 *****
;THIS TEST PERFORMS RESET TESTING AND
;TESTING OF READ ONLY REGISTER DZMSR
;AND TESTING OF WRITE ONLY REGISTER DZTDR
;:* TEST 15
;*****
†ST15: SCOPE
MOV #15,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #†ST16,NEXT ;POINT TO THE START OF THE NEXT TEST
DEVICE.CLR ;CLEAR DZ11
MOV DZMSR,R0 ;SET UP FOR ERROR MESSAGE
MOV #-1,DZTDR ;TRY TO WRITE ALL 1'S
MOV (R0),R4 ;ACTUAL
MOV R4,R5 ;MAKE EXPECTED
CMP R4,R5 ;CMP ACTUAL VS EXPECTED
BEQ 15 ;IF YES,GO CONTINUE PROCESSING
ERROR 2 ;*ERROR- BIT PATTERN NOT CORRECT
15: MOV R4,R3 ;GET A COPY OF THE ACTUAL BIT PATTERN
COM R3 ;GET THE LOGICAL INVERSE OF THE BIT PATTERN
MOV R3,DZTDR ;TRY TO WRITE
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP ACTUAL VS EXPECTED
BEQ 25 ;IF YES, GET OUT OF THIS TEST
ERROR 2 ;*NO
25:
;***** TEST 16 *****
;VERIFY THAT IF WE ARE IN "STAGGERED" MODE

```

```

2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502 014024 000004
2503 014026 012737 000016 001122
2504 014034 012737 014214 001360
2505 014042 012737 014114 001362
2506 014050 105737 001414
↑2507 014054 100001
2508 014056 104400
2509 014060 013700 002062
2510 014064 104413
2511 014066 005003
2512 014070 012702 000001
2513 014074 005737 001370
2514 014100 100405
2515 014102 013737 001360 001126
2516 014110 000177 165012
2517 014114 130237 001364
2518 014120 00101804
2519 014122 005203
2520 014124 106302
2521 014126 103372
2522 014130 104400
2523 014132 010204
2524 014134 032703 000001
2525 014140 001402
2526 014142 006204
2527 014144 000401
2528 014146 006304
2529 014150 005005
2530 014152 150405
2531 014154 000305
2532 014156 150405
2533 014160 150277 165674
2534 014164 011004
2535 014166 020504
2536 014170 001401
2537 014172 104002
2538 014174 140277 165660
2539 014200 011004
2540 014202 001402
2541 014204 005005
2542 014206 104002
2543 014210 104401
2544 014212 000743
2545
2546

```

```

; * THAT SETTING "DTR" FOR A LINE WILL
; * BRING UP "RING" AND "CARRIER" FOR THE
; * ASSOCIATED LINE IN WHICH WE ARE STAGGERED!
; * LINE0 DTR= LINE1 RING AND CARRIER
; * LINE1 DTR= LINE0 RING AND CARRIER
; * LINE2 DTR= LINE3 RING AND CARRIER
; * LINE3 DTR= LINE4 RING AND CARRIER
; *
; * ETC...

; : * TEST 16
; : *****
; : †ST16: SCOPE
; : MOV #16, STSTNM ; LOAD THE NUMBER OF THIS TEST
; : MOV #TST17, NEXT ; POINT TO THE START OF THE NEXT TEST
; : MOV #15, LOCK ; USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
; : TSTB EIAFLG ; EIA OR 20MA?
; : BPL 10$ ; BR IF EIA
; : ADVANCE ; EXIT TEST
; : 10$: MOV DZMSR, R0 ; SET REGISTER
; : DEVICE.CLR ; INIT DZ11
; : CLR R3 ; ZERO LINE NUMBER
; : MOV #1, R2 ; SET POINTER
; : TST MODE ; ARE WE IN STAGGERED MODE?
; : BMI 1$ ; YES WE ARE!
; : MOV NEXT, SLPADR ; LEAVE THIS TEST! NOT STAGGERED
; : JMP @SLPADR ; EXIT
; : 1$: BITB R2, LINE ; TEST THIS LINE?
; : BNE 3$ ; YES
; : 2$: INC R3 ; LINE #
; : ASLB R2 ; GET NEXT LINE
; : BCC 1$ ; KEEP TESTING
; : ADVANCE ; ADVANCE THIS TEST
; : 3$: MOV R2, R4 ; SAVE BINARY BIT FOR LINE #
; : BIT #BIT0, R3 ; GET STAGGERED COMPANION LINE
; : BEQ 4$ ; BR IF LINE EVEN
; : ASR R4 ; ADJUST LINE
; : BR 5$
; : 4$: ASL R4 ; ADJUST LINE
; : 5$: CLR R5 ; SET EXPECTED
; : BISB R4, R5
; : SWAB R5
; : BISB R4, R5
; : BISB R2, @HDZTCR ; SET DTR
; : MOV (R0), R4 ; READ MSR REGISTER
; : CMP R5, R4 ; OK?
; : BEQ 6$ ; YES
; : ERROR 2 ; *ERROR IN RING OR CARRIER
; : 6$: BICB R2, @HDZTCR ; CLEAR DTR
; : MOV (R0), R4 ; READ MSR
; : BEQ 7$ ; BR IF THEY CLEARED
; : CLR R5 ; SET EXPECTED TO 0
; : ERROR 2 ; *BITS NOT CLEARED
; : 7$: SCOP1 ; LOCK ON SIGNAL?
; : BR 2$ ; CONTINUE TEST

; : ***** TEST 17 *****

```

2547
 2548
 2549
 2550
 2551
 2552
 2553
 2554
 2555
 2556
 2557
 2558 014214 000004
 2559 014216 012737 000017 001122
 2560 014224 012737 014352 001360
 2561 014232 012737 014266 001362
 2562 014240 105737 001370
 2563 014244 100401
 2564 014246 104400
 2565 014250 105737 001414
 2566 014254 100774
 2567 014256 013700 002062
 2568 014262 012702 000001
 2569 014266 130237 001364
 2570 014272 001003
 2571 014274 106302
 2572 014276 103373
 2573 014300 104400
 2574 014302 005005
 2575 014304 150205
 2576 014306 000305
 2577 014310 150205
 2578 014312 150277 165542
 2579 014316 104414
 2580 014320 011004
 2581 014322 020504
 2582 014324 001401
 2583 014326 104002
 2584 014330 140277 165524
 2585 014334 104414
 2586 014336 011004
 2587 014340 001402
 2588 014342 005005
 2589 014344 104002
 2590 014346 104401
 2591 014350 000751

```

; *TEST TO VERIFY THAT IF IN "EXTERNAL"
; *MODE; SETTING DTR FOR SELECTED LINES
; *WILL BRING UP "CARRIER" AND "RING"
; *FOR THAT SAME LINE. NOTE: IF YOU HAVE
; *SELECTED MODE AS "EXTERNAL"; THE H325 TEST CONNECTER
; *MUST BE USED ON ALL SPECIFIED LINES.
; *LINES MAY BE SPECIFIED BY SWRO3=1
; *AND SWRO0=1 AT START TIME OR ALTERING
; *STATUS MAP.

; : * TEST 17
; *****
†ST17: SCOPE
MOV #17, $STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST20, NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #3$, LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
TSTB MODE ;EXTERNAL?
BMI 2$ ;BR IF YES
1$: ADVANCE ;EXIT TEST
2$: TSTB EIAFLG ;YOU BETTER BE IN
BMI 1$ ;EIA MODE FOR THIS TEST.
MOV DZMSR, R0 ;SET REGISTER
MOV #1, R2 ;SET LINE POINTER
3$: BITB R2, LINE ;LINE SELECTED?
BNE 5$ ;BR IF YES
4$: ASLB R2 ;NEXT LINE
BCC 3$ ;CONTINUE TEST
5$: ADVANCE ;ADVANCE THIS TEST
CLR R5 ;SET EXPECTED
BISB R2, R5
SWAB R5 ;B;
BISB R2, R5
BISB R2, $HDZTCR ;SET DTR
DELAY ;CABLE DELAY
MOV (R0), R4 ;READ MSR
CMP R5, R4 ;BITS OK?
BEQ 6$ ;BR IF YES
6$: ERROR 2 ;CARRIER OR RING ERROR
BICB R2, $HDZTCR ;CLEAR DTR
DELAY ;CABLE DELAY
MOV (R0), R4 ;READ MSR
BEQ 7$ ;BR IF BITS CLEARED
CLR R5 ;CLEAR EXPECTED LOC.
7$: ERROR 2 ;BITS NOT CLEARED.
SCOPI ;LOCK ON LINE?
BR 4$ ;CONTINUE TEST

```

2592
 2593
 2594
 2595
 2596
 2597
 2598
 2599
 2600 014352 000004
 2601 014354 012737 000020 001122
 2602 014362 012737 014466 001360

```

; ***** TEST 20 *****
; * THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
; * IS READY TO BE LOADED, AND THAT THE LINE SPECI-
; * FIED IN BITS 8-10 OF DZCSR CORRESPOND
; * TO THE LINE SELECTED IN DZTCR

; : * TEST 20
; *****
†ST20: SCOPE
MOV #20, $STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST21, NEXT ;POINT TO THE START OF THE NEXT TEST

```

2603	014370	104413			DEVICE.CLR		;ISSUE A "DEVICE CLEAR" (RESET)
2604	014372	013700	002042		MOV DZCSR,R0		;SET POINTER
2605	014376	012705	100040		MOV #MSENAB!TRDY,R5		;START THE EXPECTED LINE NUMBER AT 0
2606	014402	012702	000001		MOV #1,R2		;USING R2 AS A BIT POINTER, POINT TO LINE 0
2607	014406	130237	001364	15:	BIT R2,LINE		;IS THIS LINE SELECTED?
2608	014418	12	001420		BEQ 55		;IF NO, SKIP THE STARTUP
2609	014414	050277	165436	25:	BIS R2,JDZTCR		;SET THE GO BIT FOR THIS LINE
2610	014420	052710	000040		BIS #MSENAB,(R0)		;START THE SCANNER
2611	014424	005004			CLR R4		;SET FOR DELAY
2612	014426	032710	100000	35:	BIT #TRDY,(R0)		;TX READY?
2613	014432	001004			BNE 45		;BR IF YES
2614	014434	104414			DELAY		;DELAY
2615	014436	005204			INC R4		;COUNTER
2616	014440	001372			BNE 35		;BR IF (>0)
2617	014442	104002			ERROR 2		;*TX NOT READY!
2618	014444	011004		45:	MOV (R0),R4		;GET THE LINE POINTED TO BY THE SCANNER
2619	014446	020405			CMP R4,R5		;IS THE LINE NUMBER WHAT IT SHOULD BE?
2620	014450	001401			BEQ 55		;IF YES,GO WORK ON THE NEXT LINE
2621	014452	104002			ERROR 2		;*LINE NUMBER DID NOT MATCH TCR BIT
2622	014454	062705	000400	55:	ADD #400,R5		;POINT TO THE NEXT EXPECTED LINE
2623	014460	104413			DEVICE.CLR		;ISSUE A "DEVICE CLEAR" (RESET)
2624	014462	106302			ASLB R2		;POINT TO THE NEXT LINE.ARE ALL LINES TESTED?
2625	014464	103350			BCC 15		;IF NOT, GO DO THE NEXT LINE
2626	014466			65:			
2627					;***** TEST 21 *****		
2628					;*TEST TO TRANSMIT ONE CHAR AND		
2629					;*RECEIVE ONE CHAR ON ONE LINE		
2630					;*AT A TIME. THE CHAR IS "252" AND		
2631					;*ALL SELECTED LINES WILL BE TURNED ON		
2632					;*ONE AT A TIME. THIS IS THE FIRST TIME ANY		
2633					;*DATA IS CHECKED IN THE RECEIVER.		
2634					;*USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP		
2635					;*WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.		
2636					;:* TEST 21		
2637					;*****		
2638	014466	000004			TST21: SCOPE		
2639	014470	012737	000021	001122	MOV #21,\$TSTNM		;LOAD THE NUMBER OF THIS TEST
2640	014476	012737	015002	001360	MOV #TST22,NEXT		;POINT TO THE START OF THE NEXT TEST
2641	014504	012737	014760	001362	MOV #16,\$LOCK		;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2642	014512	104417			DCLASM		;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2643	014514	013701	001366		MOV PAR,R1		;PICK UP PARAMETERS
2644	014520	012702	000001		MOV #1,R2		;PICK UP INIT POINTER
2645	014524	030237	001364	15:	BIT R2,LINE		;SHOULD THIS LINE BE SET UP ?
2646	014530	001402			BEQ 25		;NO
2647	014532	010177	165314		MOV R1,JDZLPR		;SET UP LINE PARAMETERS
2648	014536	005201		25:	INC R1		;POSITION POINTER TO THE NEXT LINE
2649	014540	106302			ASLB R2		;GOT 'EM ALL ?
2650	014542	103370			BCC 15		;IF NO, GO SET UP THE NEXT LINE
2651	014544	005037	001372		CLR SAVLIN		;CLEAR LINE # INDICATOR
2652	014550	012702	000001		MOV #1,R2		;LINE POINTER
2653	014554	052777	000040	165260	BIS #MSENAB,JDZCSR		;START SCANNER
2654	014562	030237	001364	35:	BIT R2,LINE		;VALID LINE ?
2655	014566	001462			BEQ 145		;NO SET UP NEXT LINE
2656	014570	010277	165262		MOV R2,JDZTCR		;SET TCR BIT
2657	014574	032777	000200	165240	45:	BIT #RDONE,JDZCSR	;IS REC DONE = 0 ?
2658	014602	001401			BEQ 55		;IF YES, ALLOW TIME FOR TRDY TO SET

B07

```

2659 014604 104020          ERROR 20          ;*REC DONE SHOULD = 0
2660 014606 005005          CLR    R5
2661 014610 032777 100000 165224 65:  BIT    #TRDY, DZCSR
2662 014616 001004          BNE   75
2663 014620 104414          DELAY
2664 014622 105205          INCB  R5
2665 014624 001371          BNE   65
2666 014626 104003          ERROR 3
2667 014630 112777 000252 165230 75:  MOVB  #252, DZTDR ;*TRDY FAILED TO SET!
2668 014636 013705          MOV   SAVLIN, R5 ;LOAD CHARACTER
2669 014642 105737 001371  TSTB  MODE+1      ;MAKE EXPECTED LINE #
2670 014646 001406          BEQ   105         ;IS THIS TEST IN STAGGERED MODE?
2671
2672
2673
2674 014650 006205          ASR   R5          ;GET THE LAST BIT INTO THE CARRY BIT
2675 014652 103402          BCS  85          ;IF IT IS SET, GO CLEAR IT
2676 014654 000261          SEC
2677 014656 000401          BR   95          ;IF IT IS CLEAR SET IT HERE
2678 014660 000241          CLC          ;SKIP THE CLEARING
2679 014662 006105          85:  CLC          ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2680 014664 000305          95:  ROL   R5      ;GET THE NEW BIT BACK INTO R5
2681 014666 152705 00021852 105:  SWAB  R5        ;MOVE THE LINE NUMBER TO THE UPPER BYTE
2682 014672 052705          BISB  #252, R5   ;ADD CHARACTER
2683 014676 005003          CLR   R3        ;ADD DATA VALID
2684 014700 032777 000200 165134 115: BIT    #RDONE, DZCSR
2685 014706 001004          BNE  125
2686 014710 104414          DELAY
2687 014712 105203          INCB  R3
2688 014714 001371          BNE  115
2689 014716 104004          ERROR 4
2690 014720 017704 165122 125:  MOV   DZRBUF, R4 ;*RDONE FAILED TO SET!
2691 014724 020405          CMP   R4, R5    ;LOAD THE VALUE ACTUALLY RECEIVED
2692 014726 001401          BEQ  135       ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
2693 014730 104006          ERROR 6        ;IF YES, GO DO THE NEXT LINE
2694 014732 104401          135: SCOP1      ;*NO DATA/CONTENTS DID NOT COMPARE
2695 014734 040277 165116 145:  BIC   R2, DZTCR ;CHECK TO SEE IF SWITCH NINE IS SET
2696 014740 005237 001372 155:  INC   SAVLIN    ;CLEAR TCR BIT FOR THAT LINE.
2697 014744 013700 001372          MOV   SAVLIN, R0 ;INC EXPECTED LINE
2698 014750 006300          ASL  R0        ;SET UP CHARACTER OFFSET
2699 014752 106302          ASLB R2        ;MAKE THE OFFSET A POWER OF TWO
2700 014754 103302          BCC  35       ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
2701 014756 104400          ADVANCE 35    ;IF NO, GO AROUND AGAIN FOR NEXT LINE
2702
2703
2704
2705 014760 032777 100000 165054 165: BIT    #TRDY, DZCSR ;TIGHT SCOPE LOOP FOR THIS TEST. LIBOOP TRANSMITS CHARACTERS ONLY
2706 014766 001774          BEQ  165
2707 014770 112777 000252 165070  MOVB  #252, DZTDR ;IS TRANSMITTER READY?
2708 014776 104401          SCOP1          ;IF NOT, WAIT FOR IT
2709 015000 000755          BR   145       ;LOAD THE CHARACTER
2710
2711
2712
2713
2714
;***** TEST 2 *****
;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
;* CHARACTERS (FLAG MODE) AND THE RECEIVER RECEIVES (FLAG MODE)
;*(ONE LINE AT A TIME BASED UPON VALID LINES)

```

```

2715                                     : *THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
2716                                     :: * TEST 22
2717                                     : *****
2718 015002 000004                          †T22: SCOPE
2719 015004 012737 000022 001122          MOV    #22,STSTNM      ;LOAD THE NUMBER OF THIS TEST
2720 015012 012737 015330 001360          MOV    #T23,NEXT     ;POINT TO THE START OF THE NEXT TEST
2721 015020 012737 015134 001362          MOV    #4$,LOCK     ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2722 015026 104417                          DCLASM              ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2723 015030 013701 001366          MOV    PAR,R1       ;PICK UP PARAMETERS
2724 015034 012702 000001          MOV    #1,R2        ;PICK UP INIT POINTER
2725 015040 030237 001364          1$: BIT  R2,LINE     ;SHOULD THIS LINE BE SET UP ?
2726 015044 001402                          BEQ    2$           ;NO
2727 015046 010177 165000          MOV    R1,JDZLPR    ;SET UP LINE PARAMETERS
2728 015052 005201 2$: INC  R1           ;POSITION POINTER TO THE NEXT LINE
2729 015054 106302                          ASLB   R2           ;GOT 'EM ALL ?
2730 015056 103370                          BCC   1$           ;IF NO, GO SET UP THE NEXT LINE
2731 015060 005037 001372          CLR   SAVLIN        ;CLEAR LI#BNE # INDICATOR
2732 015064 012700 001422          MOV   #TDD,RO       ;POINT TO THE DATA AREA
2733 015070 005020                          CLR   (RO)+         ;CLEAR A DATA WORD
2734 015072 022700 001462          CMP   #STOP,RO     ;FINISHED ?
2735 015076 001374                          BNE   -6           ;NO
2736 015100 005000                          CLR   RO           ;CLEAR OFFSET
2737 015102 013737 002046 001400          MOV   DZRBUF,REGIST ;SAVE FOR ERROR MSG
2738 015110 012702 000001          MOV   #1,R2        ;LINE POINTER
2739 015114 052777 000040 164720          BIS   #MSENAB,JDZCSR ;START SCANNER
2740 015122 030237 001364          3$: BIT  R2,LINE     ;VALID LINE ?
2741 015126 001465                          BEQ   14$          ;NO SET UP NEXT LINE
2742 015130 010277 164722          MOV   R2,JDZTC+BR  ;SET TCR BIT
2743 015134 032777 000200 164700 4$: BIT  #RDONE,JDZCSR ;IS REC DONE = 0 ?
2744 015142 001401                          BEQ   5$           ;IF YES, ALLOW TIME FOR TRDY TO SET
2745 015144 104020                          ERROR  20          ;*REC DONE SHOULD = 0
2746 015146 005005          5$: CLR  R5
2747 015150 032777 100000 164664 6$: BIT  #TRDY,JDZCSR
2748 015156 001004                          BNE   7$
2749 015160 104414                          DELAY
2750 015162 105205                          INCB  R5
2751 015164 001371                          BNE   6$
2752 015166 104003                          ERROR  3
2753 015170 116077 001422 164670 7$: MOVB  TDD(RO),JDZTDR ;*TRDY FAILED TO SET!
2754 015176 013705 001372          MOV   SAVLIN,R5    ;LOAD CHARACTER
2755 015202 105737 001371          TSTB  MODE+1       ;MAKE EXPECTED LINE #
2756 015206 001406          BEQ   10$          ;IS THIS TEST IN STAGGERED MODE?
2757                                     ;IF NOT, SKIP STAGGERED SETUP
2758                                     ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
2759
2760 015210 006205          ASR   R5           ;GET THE LAST BIT INTO THE CARRY BIT
2761 015212 103402          BCS   8$           ;IF IT IS SET, GO CLEAR IT
2762 015214 000261          SEC   ;IF IT IS CLEAR SET IT HERE
2763 015216 000401          BR   9$           ;SKIP THE CLEARING
2764 015220 000241          8$: CLC           ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2765 015222 006105          9$: ROL  R5       ;GET THE NEW BIT BACK INTO R5
2766 015224 000305          10$: SWAB+R5     ;MOVE THE LINE NUMBER TO THE UPPER BYTE
2767 015226 156005 001422          BISB  TDD(RO),R5  ;ADD CHARACTER
2768 015232 052705 100000          BIS   #DVALID,R5 ;ADD DATA VALID
2769 015236 005003          CLR  R3
2770 015240 032777 000200 164574 11$: BIT  #RDONE,JDZCSR

```

2771	015246	001004			BNE	12\$		
2772	015250	104414			DELAY			
2773	015252	005204			INC	R4		
2774	015254	001371			BNE	11\$		
2775	015256	104004			ERROR	4		; *ROONE FAILED TO SET!
2776	015260	017704	164562	12\$:	MIBOV	DZDRBUF,R4		; LOAD THE VALUE ACTUALLY RECEIVED
2777	015264	020405			CMP	R4,R5		; COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
2778	015266	001401			BEQ	13\$; IF YES, GO DO THE NEXT LINE
2779	015270	104006			ERROR	6		; *NO DATA/CONTENTS DID NOT COMPARE
2780	015272	104401		13\$:	SCOPI			; CHECK TO SEE IF SWITCH NINE IS SET
2781	015274	105260	001422		INCB	TDD(RO)		; INCREMENT BINARY PATTERN FOR THIS LINE
2782	015300	001315			BNE	4\$; GO 'ROUND AGAIN FOR NEXT CHARACTER
2783	015302	040277	164550	14\$:	BIC	R2,DZTCR		; CLEAR TCR BIT FOR THAT LINE.
2784	015306	005237	001372	15\$:	INC	SAVLIN		; INC EXPECTED LINE
2785	015312	013700	001372		MOV	SAVLIN,RO		; SET UP CHARACTER OFFSET
2786	015316	006300			ASL	RO		; MAKE THE OFFSET A POWER OF TWO
2787	015320	106302			ASLB	R2		; SHIFT THE LINE POINTER. ARE WE ALL DONE?
2788	015322	103277			BCC	3\$; IF NO, GO AROUND AGAIN FOR NEXT LINE
2789	015324	005037	001362		CLR	LOCK		; MAKE SURE LOCK IS CLEAR FOR NEXT TEST

2790
2791
2792
2793
2794
2795
2796
2797
2798
2799

```

***** TEST 23 *****
; THIS TEST WILL PROVE THAT:
; 1) THE TRANSMITTER "BREAK BIT" WORKS
; 2) THE RECEIVER CAN FLAG "FRAMING ERRORS"
; 3) THE RECEIVER CAN FLAG "PARITY ERRORS"
; ONLY ONE LINE AT A TIME WILL BE EXERCISED.
; THIS TEST WILL NOT BE EXERCISED UNLESS
; CONNECTED BY EXTERNAL PLUG.

```

2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826

```

; * TEST 23
*****
TST23: SCOPE
MOV #23,$TSTNM ; LOAD THE NUMBER OF THIS TEST
MOV #TST24,NEXT ; POINT TO THE START OF THE NEXT TEST
MOV #3$ LOCK ; SET FOR LOOP
TST MODE ; ARE WE RUNNING IN INTERNAL MODE?
BEQ 12$ ; IF SO, SKIP THIS TEST
DCLASM ; CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV PAR,R1 ; PICK UP PARAMETERS
BIS #ODDPAR,PARITY,R1 ; FORCE ODD PARITY
MOV #1,R0 ; PICK UP INIT POINTER
1$: BIT R0,LINE ; SHOULD THIS LINE BE SET UP ?
BEQ 2$ ; IF NOT, DON'T SET IT UP
MOV R1,DZLPR ; OTHERWISE, SET UP LINE PARAMETERS
2$: INC R1
ASLB R0 ; GOT 'EM ALL ?
BCC 1$ ; NO
CLR SAVLIN ; CLEAR LINE #
MOV #1,R2 ; LINE POINTER
BIS #MSENAB,DZCSR ; SET MASTER SCAN ENABLE
MOV DZRBUF,REGIST ; SAVE FOR ERRR MESSAGE
3$: BIT R2,LINE
BEQ 10$
MOV R2,DZTCR ; SET TCR BIT
MOVB R2,DZTDR ; SET BREAK BIT
4$: MOVB #377,DZTDR ; LOAD CHARACTER

```

E07

MO-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 59
DZDZAC.P11 21-OCT-76 13:07

DZ11 DEVICE DIAGNOSTICS. COPYRIGHT 1976 DIGITAL EQUIP. CORP.

PAGE: 0082

```

2827 015470 013705 001372      MOV     SAVLIN,R5      ;MAKE EXPECTED DATA
2828 015474 105737 001371      TSTB   MODE+1        ;IS THIS TEST IN STAGGERED MODE?
2829 015500 001406                BEQ     7$            ;IF NOT, SKIP STAGGERED SETUP
2830
2831                                ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
2832
2833 015502 006205      ASR     R5            ;GET THE LAST BIT INTO THE CARRY BIT
2834 015504 103402      BCS     5$            ;IF IT IS SET, GO CLEAR IT
2835 015506 000261      SEC                ;IF IT IS CLEAR SET IT HERE
2836 015510 000401      BR      6$            ;SKIP THE CLEARING
2837 015512 000241      5$:    CLC                ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2838 015514 006105      6$:    ROL     R5            ;GET THE NEW BIT BACK INTO R5
2839 015516 000305      7$:    SWAB   R5            ;PUT LINE NUMBER IN UPPER BYTE
2840 015520 052705 130000      BIS     #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
2841 015524 005004      CLR     R4
2842 015526 032777 000200 164306 8$:    BIT     #RDONE,JDZCSR
2843 015534 001004      BNE     9$
2844 015536 104414      DELAY
2845 015540 005204      INC     R4
2846 015542 001371      BNE     8$
2847 015544 104004      ERROR  4             ;*RDONE FAILED TO SET!
2848 015546 017704 164274      9$:    MOV     JDZRBUF,R4 ;ACTUAL
2849 015552 020405      CMP     R4,R5        ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
2850 015554 001401      BEQ     10$          ;IF YES, GO CLEAN UP
2851 015556 104006      ERROR  6             ;*DATA/CONTENTS FAILED TO COMPARE
2852 015560 105077 164304      10$:   CLRB   JDZTDR    ;CLEAR BREAK BITS
2853 015564 104401      SCOPI  ;LOOP?
2854 015566 005237 001372      11$:   INC     SAVLIN    ;INC LINE #
2855 015572 040277 164260      BIC     R2,JDZTCR    ;CLEAR TCR BIT
2856 015576 106302      ASLB   R2
2857 015600 103321      BCC     3$
2858 015602 005037 001362      12$:   CLR     LOCK      ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST
2859
2860      ;***** TEST 24 *****
2861      ;* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
2862      ;*WHILE THE PROCESSOR STATUS IS SET EXACTLY
2863      ;*TO WHAT THE DZ11 PRIORITY IS SET TO.
2864      ;*DEFAULT PRIORITY IS AT 5 (240).
2865      ;:* TEST 24
2866      ;*****
2867 015606 000004      TEST24: SCOPE
2868 015610 012737 000024 001122      MOV     #24,STSTNM   ;LOAD THE NUMBER OF THIS TEST
2869 015616 012737 016114 001360      MOV     #TST25,NEXT ;POINT TO THE START OF THE NEXT TEST
2870 015624 104417      DCLASM
2871 015626 013701 001366 000001      MOV     PAR,R1       ;CLEAR DEVICE AND SET MAINT BIT IF MODE
2872 015636 030237 001364      MOV     #1,R2        ;PICK UP PARAMETERS
2873 015642 001402      1$:    BIT     R2,LINE   ;PICK UP INIT POINTER
2874 015644 010177 164202      BEQ     2$            ;SHOULD THIS LINE BE SET UP?
2875 015650 005201      2$:    MOV     R1,JDZLPR ;NO
2876 015652 106302      INC     R1            ;SET UP LINE PARAMETERS
2877 015654 103370      ASLB   R2            ;POSITION POINTER TO THE NEXT LINE
2878 015656 005037 001372      BCC     1$           ;GOT 'EM ALL?
2879 015662 106437 026216      CLR     SAVLIN       ;IF NO, GO SET UP THE NEXT LINE
2880 015666 113777 001364 164162      MTPS   #DZPRT       ;CLEAR LINE # INDICATOR
2881 015674      MOVVB  LINE,JDZTCR   ;SET CPU STATUS TO DZ11 PRIO,
2882 015762 164174      3$:    MOV     #6$,JDZTIV ;ENABLE THE VALID LINES
2883
2884      ;SET UP THE TRANSMITTER INTERRUPT VECTOR

```

```

2883 015702 012777 015770 164162      MOV      #7$,DZDRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
2884 015710 013777 026216 164156      MOV      DZPRT,DZDRIS   ;SET THE INTERRUPT VECTOR STATUS
2885 015716 013777 026216 164154      MOV      DZPRT,DZDRIS   ;SET TRANSMITTER INTERRUPT PRIORITY
2886 015724 052777 040040 164110      BIS      #TIE!#SENAB,DZDCSR ;ENABLE THE DEVICE
2887 015732 00185005          CLR      R5
2888 015734 032777 100000 164100 4$:    BIT      #TRDY,DZDCSR
2889 015742 001403          BEQ      5$
2890 015744 000240          NOP
2891 015746 000240          NOP
2892 015750 000411          BR       8$
2893 015752 104414          5$:    DELAY
2894 015754 005205          INC      R5
2895 015756 001366          BNE     4$
2896 015760 104003          ERROR   3              ;*TRDY NOT SET!
2897 015762 104010          6$:    ERROR   10         ;*TRANSMITTER SHOULD NOT INTERRUPT
2898 015764 022626          CMP     (SP)+,(SP)+    ;POP FOR FAKE RTI
2899 015766 000402          BR      8$            ;CONTINUE TEST
2900 015770 104012          7$:    ERROR   12         ;*RECEIVER SHOULD NOT INTERRUPT
2901 015772 022626          CMP     (SP)+,(SP)+    ;POP FOR ↑BFAKE RTI
2902 015774 042777 040000 164040 8$:    BIC     #TIE,DZDCSR    ;RESET TRANSMITTER INTERRUPT ENABLE
2903 016002 113777 001422 164056      MOV     #TDO,DZTDR     ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
2904 016010 012777 016100 164060      MOV     #11$,DZDRIV    ;SET UP THE TRANSMITTER INTERRUPT VECTOR
2905 016016 012777 016106 164046      MOV     #12$,DZDRIV    ;SET UP THE RECEIVER INTERRUPT VECTOR
2906 016024 013777 026216 164042      MOV     DZPRT,DZDRIS   ;SET THE INTERRUPT VECTOR STATUS
2907 016032 013777 026216 164040      MOV     DZPRT,DZDRIS   ;SET TRANSMITTER INTERRUPT PRIORITY
2908 016040 052777 000140 163774      BIS     #RIE!#SENAB,DZDCSR ;ENABLE THE DEVICE
2909 016046 005005          CLR     R5
2910 016050 032777 000200 163764 9$:    BIT     #RDONE,DZDCSR
2911 016056 001403          BEQ     10$
2912 016060 000240          NOP
2913 016062 000240          NOP
2914 016064 000412          BR     13$
2915 016066 104414          10$:   DELAY
2916 016070 005205          INC     R5
2917 016072 001366          BNE     9$
2918 016074 104004          ERROR   4              ;*NO RX DONE! (NOT SET)
2919 016076 000405          BR     13$            ;CONTINUE TEST
2920 016100 104010          11$:   ERROR   10         ;*TRANSMITTER SHOULD NOT INTERRUPT
2921 016102 022626          CMP     (SP)+,(SP)+    ;POP FOR FAKE RTI
2922 016104 000402          BR     13$            ;CONT TEST
2923 016106 104012          12$:   ERROR   12         ;*R↑BCEIVER SHOULD NOT INTERRUPT
2924 016110 022626          CMP     (SP)+,(SP)+    ;POP FOR FAKE RTI
2925 016112
2926 016112 104413          13$:   DEVICE.CLR          ;ISSUE DEVICE CLEAR (RESET)
2927
2928
2929
2930
2931
2932
2933
2934 016114 000004          ;***** TEST 25 *****
2935 016116 012737 000025 001122  ↑TST25: SCOP↑BE
2936 016124 012737 016450 001360      MOV     #25,$STSTM     ;LOAD THE NUMBER OF THIS TEST
2937 016132 104417          MOV     #TST26,NEXT    ;POINT TO THE START OF THE NEXT TEST
2938 016134 013701 001366      DCLASH          ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999

```

G07

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 61
 DZDZAC.P11 21-OCT-76 13:07

DZ11 DEVICE DIAGNOSTICS. COPYRIGHT 1976 DIGITAL EQUIP. CORP.

PAGE: 0084

2939	016140	012702	000001			MOV	#1,R2	;PICK UP INIT POINTER
2940	016144	030237	001364	15:		BIT	R2,LINE	;SHOULD THIS LINE BE SET UP ?
2941	016150	001402				BEQ	25	;NO
2942	016152	010177	163674			MOV	R1,@DZLPR	;SET UP LINE PARAMETERS
2943	016156	005201		25:		INC	R1	;POSITION POINTER TO THE NEXT LINE
2944	016160	106302				ASLB	R2	;GET 'EM ALL ?
2945	016162	103370				BCC	15	;IF NO, GO SET UP THE NEXT LINE
2946	016164	005037	001372			CLR	SAVLIN	;CLEAR LINE # INDICATOR
2947	016170	106437	026216			MTPS	@DZPRT	;SET CPU STATUS TO DZ11 PRIO
2948	016174	106437	026220			MTPS	@LESS1	;MAKE CPU ONE LEVEL LOWER THAN DZ11
2949	016200	113777	001364	163650		MOVB	LINE,@DZTCR	;ENABLE THE VALID LINES
2950	016206				35:			
2951	016206	012777	016276	163662		MOV	#65,@DZTIV	;SET UP THE TRANSMITTER INTERRUPT VECTOR
2952	016214	012777	016320	163650		MOV	#75,@DZRIV	;SET UP THE RECEIVER INTERRUPT VECTOR
2953	016222	013777	026216	163644		MOV	DZPRT,@DZRTS	;SET THE INTERRUPT VECTOR STATUS
2954	016230	013777	026216	163642		MOV	DZPRT,@DZTIS	;SET TRANSMITTER INTERRUPT PRIORITY
2955	016236	052777	040040	163576		BIS	#TIE!MSENAB,@DZCSR	;ENABLE THE DEVICE
2956	016244	005005				CLR	R5	
2957	016246	032777	100000	163566	45:	BIT	#TRDY,@DZCSR	
2958	016254	001404				BEQ	55	
2959	016256	000240				NOP		
2960	016260	000240				NOP		
2961	016262	104007				ERROR	7	;*TRANSMITTER FAILED TO INTERRUPT
2962	↑B016264		000417			BR	85	
2963	016266	104414			55:	DELAY		
2964	016270	005205				INC	R5	
2965	016272	001365				BNE	45	
2966	016274	104003				ERROR	3	;*TRDY NOT SET!
2967	016276	022626			65:	POP2SP		;REMOVE THE INTERRUPT FROM THE STACK
2968	016300	042777	040000	163534		BIC	#TIE,@DZCSR	;DON'T LET ANY MORE INTERRUPTS OCCUR
2969	016306	106437	026216			MTPS	@DZPRT	;SET CPU STATUS TO DZ11 PRIORITY
2970	016312	106437	026220			MTPS	@LESS1	;MAKE CPU ONE LEVEL LOWER THAN DZ11
2971	016316	000402				BR	85	;RETURN TO THE NORMAL FLOW
2972	016320	104012			75:	ERROR	12	;*RECEIVER SHOULD NOT INTERRUPT
2973	016322	022626				CMP	(SP)+,(SP)+	;POP FOR FAKE RTI
2974	016324	042777	040000	163510	85:	BIC	#TIE,@DZCSR	;RESET TRANSMITTER INTERRUPT ENABLE
2975	016332	113777	001422	163526		MOVB	TDO,@DZTCR	;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
2976	016340	012777	016432	163530		MOV	#115,@DZTIV	;SET UP THE TRANSMITTER INTERRUPT VECTOR
2977	016346	012777	016440	163516		MOV	#125,@DZRIV	;SET UP THE RECEIVER INTERRUPT VECTOR
2978	016354	013777	026216	163512		MOV	DZPRT,@DZRTS	;SET THE INTERRUPT VECTOR STATUS
2979	016362	013777	026216	163510		MOV	DZPRT,@DZTIS	;SET TRANSMITTER INTERRUPT PRIORITY
2980	016370	052777	000140	163444		BIS	#R1BIE!MSENAB,@DZCSR	;ENABLE THE DEVICE
2981	016376	005005				CLR	R5	
2982	016400	032777	000200	163434	95:	BIT	#RDONE,@DZCSR	
2983	016406	001404				BEQ	105	
2984	016410	000240				NOP		
2985	016412	000240				NOP		
2986	016414	104011				ERROR	11	;*RECEIVER FAILED TO INTERRUPT
2987	016416	000413				BR	135	
2988	016420	104414			105:	DELAY		
2989	016422	005205				INC	R5	
2990	016424	001365				BNE	95	
2991	016426	104004				ERROR	4	;*NO RX DONE! (NOT SET)
2992	016430	000406				BR	135	;CONTINUE TEST
2993	016432	104010			115:	ERROR	10	;*TRANSMITTER SHOULD NOT INTERRUPT
2994	016434	022626				CMP	(SP)+,(SP)+	;POP ↑B FOR FAKE RTI

```

2995 016436 000403
2996 016440 022626
2997 016442 005077 163374
2998 016446
2999 016446 104413
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011 016450 000004
3012 016452 012737 000026 001122
3013 016460 012737 017102 001360
3014 016466 104417
3015 01647180 013701 001366
3016 016474 012702 000001
3017 016500 030237 001364
3018 016504 001402
3019 016506 010177 163340
3020 016512 005201
3021 016514 106302
3022 016516 103370
3023 016520 005037 001372
3024 016524 012777 016754 163340
3025 016532 01813777 026216 163334
3026 016540 012777 017044 163330
3027 016546 013777 026216 163324
3028 016554 052777 000040 163260
3029 016562 012702 000001
3030 016566 030237 001364
3031 016572 001004
3032 016574 005237 001372
3033 016600 106302
3034 016602 000771
3035 016604 106427 000340
3036 016610 000240
3037 016612 000240
3038 016614 110277 163236
3039 016620 005777 163222
3040 016624 100001
3041 016626 104017
3042 016630 105777 163206
3043 016634 100001
3044 016636 104020
3045 016640 005005
3046 016642 005004
3047 016644 005777 163172
3048 016650 100404
3049 016652 104414
3050 016654 005204

```

```

12$: BR 13$ :CONT TEST
POP2SP :REMOVE THE INTERRUPT FROM THE STACK
CLR 2DZCSR :DON'T ALLOW ANY MORE INTERRUPTS
13$: DEVICE.CLR ;ISSUE DEVICE CLEAR (RESET)

:***** TEST 26 *****
:THIS TEST VERIFIES THAT THE RECEIVER WILL
:INTERRUPT BEFORE THE TRANSMITTER EVEN
:THOUGH THE TRANSMITTER WAS ENABLED
:FIRST. SET PS TO LEVEL 7;
:GET RDONE AND TRDY TO SET;
:SET TX IE AND RX IE;
:CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
::* TEST 26
:*****
↑ST26: SCOPE
MOV #26,$STNM ;LOAD THE NUMBER OF THIS TEST
MOV #↑ST27,NEXT ;POINT TO THE START OF THE NEXT TEST
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV PAR,R1 ;PICK UP PARAMETERS
MOV #1,R2 ;PICK UP INIT POINTER
1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
BEQ 2$ ;NO
MOV R1,2DZLPR ;SET UP LINE PARAMETERS
2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
ASLB R2 ;GOT 'EM ALL ?
BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
CLR SAVLIN ;CLEAR LINE # INDICATOR
MOV #8,$DZRIV ;SETUP INTERRUPT STUFF
MOV DZPRT,2DZRIS ;
MOV #12,$DZTIV ;
MOV DZPRT,2DZTIS ;
BIS #MSENAB,2DZCSR ;
MOV #1,R2 ;LINE POINTER
3$: BIT R2,LINE ;VALID LINE ?
BNE 4$
INC SAVLIN
ASLB R2
BR 3$
4$: MTPS #PR7
NOP
NOP
MOV R2,2DZTCR ;SET TCR BIT
TST 2DZRBUF ;VALID DATA?
BPL .+4 ;IT BETTER NOT BE SET
ERROR 17 ;DATA VALID SHOULD NOT BE SET
5$: TSTB 2DZCSR ;RECEIVER DONE ?
BPL .+4
ERROR 20 ;RECEIVER DONE BIT SHOULD NOT BE SET
CLR R5
CLR R4
99$: TST 2DZCSR ;WAIT FOR TRDY
BMI 100$ ;BR IF READY
DELAY ;STALL TIME
INC R4

```

```

3051 016656 001372          BNE      99$
3052 016660 104003          ERROR    3          ;TRDY FAILED TO SET
3053 016662 105077 163200 100$: CLR      2DZTDR
3054 016666 005004          CLR      R4
3055 016670 032777 000200 163144 6$: BIT      #RDONE,2DZCSR
3056 016676 001004          BNE      7$
3057 016700 104414          DELAY
3058 016702 005204          INC      R4
3059 016704 001371          BNE      6$
3060 016706 104004          ERROR    4          ;*RDONE FAILED TO SET!
3061 016710 005777 163126 7$: TST      2DZCSR          ;TRANS DONE BIT = 1 ?
3062 016714 100401          BMI      +4          ;YES
3063 016716 104003 1B      ERROR    3          ;*NO TRANS DONE FAILED TO SET
3064          ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
3065          ;SET INTERRUPT ENABLES AND WATCH THE FUR FLY
3066 016720 052777 040000 163114 BIS      #TIE,2DZCSR
3067 016726 052777 000100 163106 BIS      #RIE,2DZCSR
3068 016734 106427 000000 MTPS    #0
3069 016740 000240          NOP
3070 016742 000240          NOP
3071 016744 104007          ERROR    7          ;*TRANSMITTER FAILED TO INTERRUPT
3072 016746 104011          ERROR    11         ;*RECEIVER FAILED TO INTERRUPT
3073          ;CHECK BR LEVEL
3074 016750 000137 017050 JMP      13$          ;GET OUT
3075
3076          ;RECEIVER INTERRUPT ROUTINE
3077 016754 017704 163066 8$: MOV      2DZRBUF,R4          ;ACTUAL
3078 016760 010403          MOV      R4,R3
3079 016762 000303          SWAB    R3
3080 016764 042703 177770 BIC      #1C<7>,R3          ;STRIP JUNK
3081 016770 105737 001371 TSTB    MODE+1          ;IS THIS TEST IN STAGGERED MODE?
3082 016774 001406          BEQ      11$          ;IF NOT, SKIP STAGGERED SETUP
3083
3084          ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3085
3086 016776 006203          ASR      R3          ;GET THE LAST BIT INTO THE CARRY BIT
3087 017000 103402          BCS     9$          ;IF IT IS SET, GO CLEAR IT
3088 017002 000261          SEC
3089 017004 000401          BR      10$          ;IF IT IS CLEAR SET IT HERE
1B 3090          ;SKIP THE CLEARING
3091 017010 006103 000241 10$: CLC          ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY);
3092 017012 020337 001372 11$: ROL      R3          ;GET THE NEW BIT BACK INTO R3
3093 017016 001401          CMP      R3,SAVLIN          ;IS THIS A VALID LINE
3094 017020 104015          BEQ      +4          ;YES
3095 017022 042704 177400          ERROR    15         ;*INVALID LINE
3096 017026 120504          BIC      #1C<377>,R4          ;STRIP JUNK
3097 017030 001401          CMPB    R5,R4          ;DATA COMPARE ?
3098 017032 104005          BEQ      +4          ;YES
3099 017034 040277 163016          ERROR    5          ;*DATA DOES NOT COMPARE
3100 017040 022626          BIC      R2,2DZTCR          ;CLEAR TCR BIT
3101 017042 000402          POP2SP          ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
3102          BR      13$          ;GO GET OUT OF INTERRUPT MODE
3103 017044 104011 12$: ;TRANSMITTER INTERRUPT SVC ROUTINE
3104          ERROR    11         ;THE RECEIVER INTERRUPT FAILED
3105          ;TO OVERRIDE THE TRANSMITTER
3106 017046 022626          POP2SP          ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
3106 017050 042777 040100 162764 13$: BIC      #TIE!RIE,2DZCSR ;CLEAR INTERRUPT ENABLES
    
```

```

3107 017056 013777 002074 163006      MOV     DZRIS, DZRIV ;RESTORE TRAPCATCHER
3108 017064 005077 163004      CLR     DZRIS
3109 017070 013777 002100 163000      MOV     DZTIS, DZTIV
3110 017076 005077 16277E      CLR     DZTIS
3111                                     ;***** TEST 27 *****
3112                                     ;*THIS TEST VERIFIES OVERRUN AND SILO ALARM
3113                                     ;*ONE LINE AT A TIME - BASED UPON VALID LINES
3114                                     ;*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
3115                                     ;*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
3116                                     ;*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
3117                                     ;*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
3118                                     ;*CHAR PULLED OUT OUT THE SILO.
3119                                     ;*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
3120                                     ;*ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
3121                                     ;*USED TO SCOPE SILO ALARM PULSES, ETC.
3122                                     ;:* TEST 27
3123                                     ;*****
3124 017102 000004      TST27: SCOPE
3125 017104 012737 000027 001122      MOV     #27, STSTNM ;LOAD THE NUMBER OF THIS TEST
3126 017112 012737 017630 001360      MOV     #TST30, NEXT ;POINT TO THE START OF THE NEXT TEST
3127 017120 012737 017534 001362      MOV     #18$, LOCK ;SET FOR LOOP
3128 017126 104417      DCLASM ;CLEAR DEVICE AND SET MAINT BIT I+BF I MODE
3129 017130 013701 001366      MOV     PAR, R1 ;PICK UP PARAMETERS
3130 017134 012702 000001      MOV     #1, R2 ;PICK UP INIT POINTER
3131 017140 030237 001364      1$: BIT  R2, LINE ;SHOULD THIS LINE BE SET UP ?
3132 017144 001402      BEQ     2$ ;NO
3133 017146 010177 162700      MOV     R1, DZLPR ;SET UP LINE PARAMETERS
3134 017152 005201      2$: INC  R1 ;POSITION POINTER TO THE NEXT LINE
3135 017154 106302      ASLB   R2 ;GOT 'EM ALL ?
3136 017156 103370      BCC    1$ ;IF NO, GO SET UP THE NEXT LINE
3137 017160 005037 001372      CLR     SAVLIN ;CLEAR LINE # INDICATOR
3138 017164 012700 001422      MOV     #TDO, RO ;POINT TO THE DATA AREA
3139 017170 005020      CLR    (RO)+ ;CLEAR A DATA WORD
3140 017172 022700 001462      CMP     #STOP, RO ;FINISHED ?
3141 017176 001374      BNE    .-6 ;NO
3142 017200 005000      CLR     RO ;CLEAR OFFSET
3143 017202 012702 000001      MOV     #1, R2 ;LINE POINTER
3144 017206 052777 010040 162626      BIS    #MSENAB!SILOEN, DZCSR ;START SCANNER & SET SILO ENABLE
3145 017214 030237 001364      3$: BIT  R2, LINE ;VALID LINE?
3146 017220 001002      BNE    .+6 ;YES
3147 017222 000137 017510      JMP     17$ ;TRY NEXT LINE
3148 017226 013700 001372      MOV     SAVLIN, RO ;MAKE OFFSET
3149 017232 006300      ASL    RO ;MAKE POWER OF TWO
3150 017234 010277 162616 1B      MOV     R2, DZTCR ;SET TCR BIT
3151 017240 105777 162576      4$: TSTB DZCSR ;REC DONE = 1 ?
3152 017244 100001      BPL    .+4
3153 017246 104020      ERROR  20 ;REC DONE SHOULD NOT = 1
3154 017250 005003      CLR    R3 ;SET CHARACTER COUNT
3155 017252 005004      5$: CLR  R4
3156 017254 032777 100000 162560 6$: BIT  #TRDY, DZCSR
3157 017262 001004      BNE    7$
3158 017264 104414      DELAY
3159 017266 105204      INCB   R4
3160 017270 001371      BNE    6$
3161 017272 104003      ERROR  3 ;*TRDY FAILED TO SET
3162 017274 116077 001422 162564 7$: MOVB  TDO(RO), DZTDR ;LOAD A CHARACTER

```

3163	017302	005260	001422		INC	T00(R0)		;SET UP NEXT CHARACTER
3164	017306	020327	000017		CMP	R3,#15.		;16 CHARACTERS ?
3165	017312	103006			BHIS	8\$		
3166	017314	032777	020000	162520	BIT	#SILOAL, @DZCSR		;SILO ALARM = 0 ?
3167	017322	001401			BEQ	.+4		;YES
3168	017324	104013			ERROR	13		;*SILO ALARM SHOULD NOT = 1
3169								;UNTIL 16. DATA CHARACTERS
3170	017326	000411			BR	10\$		
3171	017330	005004			CLR	R4		
3172	017332	032777	020000	162502	8\$: 9\$:	BIT	#SILOAL, @DZCSR	
3173	017340	001004			BNE	10\$		
3174	017342	104414			DELAY			
3175	017344	005204			INC	R4		
3176	017346	001371			BNE	9\$		
3177	017350	104014			ERROR	14		;*SILO ALARM FAILED TO SET!
3178								;SILO ALARM SHOULD =1 AFTER 16.
3179								;DATA CHARACTERS
3180	017352	005203			10\$:	INC	R3	;INC CHAR COUNT
3181	017354	022703	000102		CMP	#66.,R3		;FINISHED SENDING CHARACTERS ?
3182	017360	001334			BNE	5\$;NO
3183	017362	005004			CLR	R4		
3184	017364	104414			DELAY			
3185	017366	105204			INCB	R4		
3186	017370	001375			BNE	.-4		
3187								;NOW LETS READ THE SILO
3188	017372	013705	001272		MOV	SAVLIN,R5		;MAKE EXPECTED L+IBINE #
3189	017376	105737	021371		TSTB	MODE+1		;IS THIS TEST IN STAGGERED MODE?
3190	017402	001406			BEQ	13\$;IF NOT, SKIP STAGGERED SETUP
3191								
3192								;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3193								
3194	017404	006205			ASR	R5		;GET THE LAST BIT INTO THE CARRY BIT
3195	017406	103402			BCS	11\$;IF IT IS SET, GO CLEAR IT
3196	017410	000261			SEC			;IF IT IS CLEAR SET IT HERE
3197	017412	000401			BR	12\$;SKIP THE CLEARING
3198	017414	000241			11\$: 12\$:	CLC		;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3199	017416	006105			ROL	R5		;GET THE NEW BIT BACK INTO R5
3200	017420	000301B5			13\$:	SWAB	R5	;PUT IN UPPER BYTE
3201	017422	052705	100000		BIS	#DVALID,R5		;ADD DATA VALID
3202	017426	017704	162414		14\$:	MOV	@DZRBUF,R4	;ACTUAL
3203	017432	020405			CMP	R4,R5		;ACTUAL VS. EXPECTED
3204	017434	001401			BEQ	15\$;YES
3205	017436	104006			ERROR	6		;*DATA/CONTENTS DID NOT COMPARE
3206	017440	032777	020000	162374	15\$:	BIT	#SILOAL, @DZCSR	;SILO ALARM= 0 ?
3207	017446	001401			BEQ	16\$;YES
3208	017450	104016			ERROR	16		;READING DZRBUF DID NOT CLEAR SILO ALARM
3209	017452	005205			16\$:	INC	R5	;UP CHARACTER
3210	017454	120527	000077		CMPB	R5,#63.		;LAST SILO CHAR ?....64TH CHAR
3211	017460	101762			BLOS	14\$		
3212	017462	005205			INC	R5		;ADD 1 MORE FOR THE CLOBBERED CHAR
3213	017464	052705	040000		BIS	#OVRUN,R5		;ADD OVERRUN TO EXPECTED
3214	017470	120527	000101		CMPB	R5,#65.		;LAST CHARACTER ?
3215	017474	001754			BEQ	14\$		
3216	017476	017704	162344		MOV	@DZRBUF,R4		;FOR GOOD MEASURE
3217	017502	005704			TST	R4		;DATA VALID SHOULD = 0
3218	017504	100001			BPL	17\$;YES

```

3219 017506 104017
3220 017510 040277 162342
3221 017514 104401
3222 017516 005237 001372
3223 017522 106302
3224 017524 103402
3225 017526 000137 017214
3226 017532 104400
3227
3228
3229
3230
3231
3232 017534 052777 010040 162300
3233 017542 012777 017620 162326
3234 017550 012737 000024 001216
3235 017556 050277 162274
3236 017562 052777 040000 162252
3237 017570 106427 000000
3238 017574 000001
3239 017576 005337 001216
3240 017602 001374
3241 017604 042777 050040 162230
3242 017612 104401
3243 017614 000137 017510
3244 017620 112777 000252 162240
3245 017626 000002
3246
3247
3248
3249
3250
3251
3252
3253
3254 017630 000004
3255 017632 012737 000030 001122
3256 017640 012737 020212 001360
3257 017646 012737 017734 001362
3258 017654 104417
3259 017656 013701 001366
3260 017662 012702 000001
3261 017666 030237 001364
3262 017672 001402
3263 017674 010177 162152
3264 017700 005201
3265 017702 106302
3266 017704 103370
3267 017706 005037 001372
3268 017712 012700 001422
3269 017716 005020
3270 017720 022700 001462
3271 017724 001374
3272 017726 005000
3273 017730 012702 000001
3274 017734 012777 020154 162130

```

```

17$: ERROR 17 ;DATA VALID SHOULD = 180
BIC R2,ADZTCR ;CLR TCR BIT
SCOPI ;LOOP?
INC SAVLIN ;INC EXPECTED LINE
ASLB R2 ;NEXT LINE
BCS +6 ;NO
JMP 3$ ;YES
ADVANCE ;GO TO NEXT TEST

;TIGHT SCOPE LOOP FOR THIS TEST. SENDS 20 CHARACTERS
;ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
;USED TO SCOPE SILO ALARM PULSES, ETC.

18$: BIS #MSENAB!SILOEN,ADZCSR ;SETUP DEVICE
MOV #20$,ADZTIV ;SETUP TRANSMITTER VECTOR
MOV #20$,STMPD ;TEMPORARY COUNT OF CHARACTER BURST
BIS R2,ADZTCR ;ENABLE LINE
BIS #TIE,ADZCSR ;ENABLE INTERRUPTS
MTPS #0 ;LOWER PRIORITY
19$: WAIT ;ALLOW INTERRUPTS
DEC STMPD ;REDUCE COUNT. ALL CHARACTERS SENT?
BNE 19$ ;IF NO, WAIT FOR MORE
BIC #SILOEN!MSENAB!TIE,ADZCSR ;RESET SILO COUNTER, CLEAR STROBE
SCOPI ;LOOP AGAIN?
JMP 17$ ;IF NOT, RETURN TO WHERE YOU LEFT OFF
20$: MOV #252,ADZTDR ;SEND A CHARACTER
RTI ;ALLOW MORE CHARACTERS TO COME

;***** TEST 30 *****
;THIS TEST THAT "SILO ENABLE" WILL INHIBIT
;RECEIVER INTERRUPTS AND THAT ON THE
;16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
;INTERRUPT WITH "RIE" SET.
;THIS WILL DO ALL SELECTED LINES ONE AT A TIME.

;:* TEST 30
;:*18*****
18$: SCOPE
MOV #30,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST31,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV #3$,LOCK ;SET FOR LOOP
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV PAR,R1 ;PICK UP PARAMETERS
MOV #1,R2 ;PICK UP INIT POINTER
1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
18$ BEQ 2$ ;NO
MOV R1,ADZLPR ;SET UP LINE PARAMETERS
2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
ASLB R2 ;GOT 'EM ALL ?
BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
CLR SAVLIN ;CLEAR LINE # INDICATOR
MOV #TDO,RO ;POINT TO THE DATA AREA
CLR (RO)+ ;CLEAR A DATA WORD
CMP #STOP,RO ;FINISHED ?
BNE -6 ;NO
CLR RO ;CLEAR OFFSET
MOV #1,R2 ;LINE POINTER
3$: MOV #1$,ADZRIV ;SET FOR UNEXPECTED INTER.

```

3275	017742	012777	000340	162124		MOV	#PR7, @DZRIS	:SET PRIO.
3276	017750	052777	010140	162064		BIS	#MSENAB!SILOEN!RIE, @DZCSR	:START SCANNER & SET SILO ENABLE
3277								:VALID LINE?
3278	017756	030237	001364			BIT	R2, LINE	:YES
3279	017762	001002				BNE	.+6	:TRY NEXT LINE
3280	017764	000137	020164			JMP	17\$:EMPTY THE 1B SILO
3281	017770	005777	162052			TST	@DZRBUF	:BR IF DATA VALID IS SET!
3282	017774	100775				BMI	.-4	:SET PROCESSOR PRIORITY TO 0
3283	017776	106427	000000			MTPS	#0	:MAKE OFFSET
3284	020002	013700	001372			MOV	SAVLIN, R0	:MAKE POWER OF TWO
3285	020006	006300				ASL	R0	:SET TCR BIT
3286	020010	010277	162042			MOV	R2, @DZTCR	
3287	020014	005004			5\$:	CLR	R4	
3288	020016	032777	100000	162016	6\$:	BIT	#TRDY, @DZCSR	
3289	020024	001004				BNE	7\$	
3290	020026	104414				DELAY		
3291	020030	005204				INC	R4	
3292	020032	001371				BNE	6\$	
3293	020034	104003				ERROR	3	:*TRDY FAILED TO SET
3294	020036	116077	001422	162022	7\$:	MOVB	T00(R0), @DZTDR	:LO!BAD A CHARACTER
3295	020044	005260	001422			INC	T00(R0)	:SET UP NEXT CHARACTER
3296	020050	022760	000017	001422		CMP	#15., T00(R0)	:15 CHARS YET?
3297	020056	001406				BEQ	8\$	
3298	020060	032777	020000	161754		BIT	#SILOAL, @DZCSR	:SILO ALARM = 0 ?
3299	020066	001401				BEQ	.+4	:YES
3300	020070	104013				ERROR	13	:*SILO ALARM SHOULD NOT = 1
3301								:UNTIL 16. DATA CHARACTERS
3302	020072	000751				BR	6\$	
3303	020074	012777	020162	161770	8\$:	MOV	#12\$, @DZRIV	:SET NEW VECTOR
3304	020102	032777	100000	161732		BIT	#TRDY, @DZCSR	:READY FOR 16TH CHAR
3305	020110	001774				BEQ	.-6	
3306	020112	016077	001422	161746		MOV	T00(R0), @DZTDR	:LOAD THE 16TH CHAR.
3307	020120	005004				CLR	R4	
3308	020122	032777	020000	161712	9\$:	BIT	#SILOAL, @DZCSR	
3309	020130	001005				BNE	10\$	
3310	020132	104414				DELAY		
3311	020134	005204				INC	R4	
3312	020136	001371				BNE	9\$	
3313	020140	104014				ERROR	14	:*SILO ALARM FAILED TO SET!
3314	020142	000410				BR	17\$:SILO ALARM SHOULD =1 AFTER 16.
3315								:DATA CHARACTERS
3316	020144	000240			10\$:	NOP		:STALL
3317	020146	000240				NOP		
3318	020150	104000				ERROR		:SILO ALARM NOT INTERRUPTING.
3319	020152	000404				BR	17\$:CONTINUE TEST.
3320	020154	022626			11+8\$:	CMP	(SP)+, (SP)+	:FAKE RTI
3321	020156	104012				ERROR	12	:RX SHOULD NOT INTERRUPT
3322	020160	000401				BR	17\$:CONTINUE
3323	020162	022626			12\$:	CMP	(SP)+, (SP)+	:GOOD INTERRUPT TO HERE.
3324	020164	040277	161666		17\$:	BIC	R2, @DZTCR	:CLR TCR BIT
3325	020170	104401				SCOP1		:LOOP?
3326	020172	005237	001372			INC	SAVLIN	:INC EXPECTED LINE
3327	020176	106302				ASLB	R2	:NEXT LINE
3328	020200	103402				BCS	.+6	:NO
3329	020202	000137	017734			JMP	3\$:YES
3330	020206	005037	001362			CLR	LOCK	:CLEAR TIGHT LOOP FOR NEXT TEST

```

3331 ;***** TEST 31 *****
3332 ;*THIS TEST RUNS ALL LINES FULL BORE
3333 ;*BASED UPON QUALIFIED LINES
3334 ;*.. THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
3335 ;*TRANSMITTER
3336 ;:* TEST 31
3337 ;*****
3338 020212 000004          †ST31: SCOPE
3339 020214 012737 000031 001122      MOV    #31,STSTNM      ;LOAD THE NUMBER OF THIS TEST
3340 020222 012737 021020 001360      MOV    #TST32,NEXT    ;POINT TO THE START OF THE NEXT TEST
3341 020230 104417          DCLASM                ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3342 020232 013737 001364 021016      MOV    LINE,RXTCR     ;SET IMAGE OF TCR BITS
3343 020240 013701 001366      RSTART: MOV   PAR,R1   ;PICK UP PARAMETER
3344 020244 012700 000001          MOV    #1,R0          ;PICK UP INIT POINTER
3345 020250 030037 001364      INIT:  BIT    R0,LINE  ;SHOULD THIS LINE BE SET UP
3346 020254 001402          BEQ    1$             ;NO
3347 020256 010177 161570          MOV    R1,@DZLPR     ;SET UP LINE PARAM REGISTER
3348 020262 005201          1$:   INC    R1
3349 020264 106300          ASLB   R0             ;GOT 'EM ALL ?
3350 020266 103370          BCC    INIT          ;NO
3351 020270 012700 001422 18          MOV    #TDO,R0       ;CLEAR TRANS DATA POINTER & REC POINTERS
3352 020274 005020          INIT1: CLR   (R0)+
3353 020276 022700 001462          CMP    #STOP,R0     ;FINISHED ?
3354 020302 001374          BNE   INIT1         ;NO, CONTINUE CLEARING
3355 020304 012777 020542 161560      MOV    #RXSVC,@DZRIV ;SET UP REC INTR VECTOR
3356 020312 012777 000340 161554      MOV    #PR7,@DZRI5   ;STATUS
3357 020320 012777 020442 161550      MOV    #TXSVC,@DZTIV ;SET UP TRANS INTR VECTOR
3358 020326 012777 000340 161544      MOV    #PR7,@DZTI5   ;STATUS
3359 020334 052777 000040 161500      BIS    #MSENAB,@DZCSR ;SET MASTER SCAN ENABLE
3360 020342 052777 000100 161472      BIS    #RIE,@DZCSR   ;SET REC I18NTR ENABLE
3361 020350 052777 040000 161464      BIS    #TIE,@DZCSR   ;SET TRANS INTR ENABLE
3362 020356 113777 001364 161472      MOV    LINE,@DZTCR   ;SET TCR BITS...UP UP AND AWAY !
3363 020364 106437 026220          MTPS   @#LESS1      ;ALLOW INTERRUPTS
3364
3365
3366 020370 005037 020440          SNAP: CLR   66$
3367 020374 013727 006604          67$:  MOV    DLYCNT,(PC)+ ;SET FOR DELAY
3368 020400 000000          68$:  0
3369 020402 005337 020400          DEC   68$
3370 020406 001375          BNE   -4
3371 020410 105737 021016          TSTB  RXTCR          ;WAIT FOR ALL RECIEVERS TO FINISH
3372 020414 001002          BNE   3$
3373 020416 000137 020716          JMP   OUT
3374 020422 005237 020440          3$:   INC   66$
3375 020426 001362          BNE   67$
3376 020430 104007          ERROR 7              ;*TRANSMITTER FAILED TO INTERRUPT
3377 020432 104011          ERROR 11             ;*RECEIVER FAILED TO INTERRUPT
3378 020434 000137 020770          JMP   FINI
3379 020440 000000          66$:  0
3380
3381 ;TRANS INTR SVC ROUTINE
3382 020442 005777 161374          †XSVC: TST   @DZCSR    ;TRANS INTR ?
3383 020446 100401          BMI   +4
3384 020450 104003          ERROR 3              ;*TRANSMITTER FAILED
3385 020452 117703 161366          MOV    #DZDZCSR,R3  ;SAVE IT
3386 ;NOW TEST FOR LINE # ETC

```

3387	020456	042703	177770		BIC	#1C(7),R3	:STRIP JUNK
3388	020462	010304			MOV	R3,R4	:SAVE
3389	020464	012702	000001		MOV	#1,R2	:SET UP POSITION POINTER
3390	020470	105303		35:	DECB	R3	:IS IT THIS LINE ?
3391	020472	100402			SMI	45	:YES
3392	020474	006302			ASL	R2	:UP THE LINE #
3393	020476	000774			BR	35	:GO 'ROUND AGAIN
3394	020500	030237	001364	45:	BIT	R2,LINE	:VALID LINE?
3395	020504	001001			BNE	.+4	:YES
3396	020506	104011			ERROR	11	:NO,INVALID LINE!!!!
3397	020510	042704	177770		BIC	#1C(7),R4	:STRIP JUNK
3+8398		020514	006304		ASL	R4	:MAKE POWER OF 2
3399	020516	116477	001422	161342	MOVB	TDO(R4),DZTDR	:LOAD CHARACTER
3400	020524	105264	001422		INCB	TDO(R4)	:SET UP NEXT CHARACTER
3401	020530	001002			BNE	55	:LAST CHARACTER ?
3402	020532	040277	161320		BIC	R2,DZTCR	:YES,CLEAR TCR BIT
3403	020536	005200		55:	INC	RO	:INCR RECEIVER TIMER
3404	020540	000002			RTI		
3405							
3406							
3407							
3408	020542	105777	161274				
3409	020546	100401					
3410	020550	104004					
3411	020552	032777	020000	161262			
3412	020560	001401					
3413	020562	104000					
3414	020564	017704	161256				
3415	020570	100401					
3416	020572	104000					
3417	020574	032704	070000				
3418	020600	001401					
3419	020602	104000					
3420	020604	010403					
3421	020606	000303					
3422	020610	042703	177770				
3423	020614	010337	001372				
3424	020620	†8012702		000001			
3425	020624	105303					
3426	020626	100402					
3427	020630	006302					
3428	020632	000774					
3429	020634	030237	001364	65:			
3430	020640	001001					
3431	020642	104011					
3432	020644	013703	001372				
3433	020650	006303					
3434	020652	126304	001442				
3435	020656	001405					
18 3436		020660	016305	001442			
3437	020664	042704	177400				
3438							
3439							
3440							
3441	020670	104005					
3442	020672	005263	001442	25:			

.REC INTR SVC ROUTINE

RXSVC:	TSTB	DZCSR	:REC DONE ?
	BMI	.+4	:YES
	ERROR	4	:FALSE INTERRUPT
	BIT	#SILOAL,DZCSR	:SILO ALARM?
	BEQ	.+4	:NO
	ERROR		:SILO ALARM SHOULD NOT =1
	MOV	DZRBUF,R4	:SAVE IT
	BMI	.+4	:YES
	ERROR		:YOU LOSE ...DATA VALID WAS'NT SET
	BIT	#OVRRUN!FRMERR!PARER,R4	
	BEQ	.+4	
	ERROR		:RECEIVER ERROR FLAG/S WERE SET
	MOV	R4,R3	
	SWAB	R3	
	BIC	#1C(7),R3	:STRIP JUNK
	MOV	R3,SAVLIN	:SAVE LINE NUMBER
	MOV	#1,R2	:SET UP POSITION POINTER
55:	DECB	R3	
	BMI	65	
	ASL	R2	:RE POSITION POINTER
	BR	55	:GO 'ROUND AGAIN
65:	BIT	R2,LINE	:LINE VALID ?
	BNE	.+4	:YES
	ERROR	11	:INVALID LINE #
	MOV	SAVLIN,R3	:GET THE LINE NUMBER AGAIN
	ASL	R3	:USE R3 AS A POINTER IN THE DATA TABLE
	CMPB	TRO(R3),R4	:DOES THE DATA CHARACTER COMPARE ?
	BEQ	25	:YES
	MOV	TRO(R3),R5	:SAVE EXPECTED
	BIC	#1C(37),R4	:CLEAR JUNK
			:R2 = LINE # BY BIT POSITION
			:R4 = ACTUAL DATA
			:R5 = EXPECTED DATA
			:*NO, DATA DOES NOT COMPARE
25:	INC	TRO(R3)	:SET UP FOR NEXT CHARACTER

```

3443 020676 105763 001442      TSTB   TR0(R3) ;ALL CHARS DONE?
3444 020702 001002              BNE    .+6
3445 020704 040237 021016      BIC    R2,RXTCR ;ZERO LINE DONE INDICATOR.
3446 020710 012716 020370      MOV    #SNAP,(SP) ;RESET THE BACKGROUND TIMING LOOP
3447 020714 000002              RTI
3448
3449
3450
3451 020716 106427 000340      OUT:   ;FINISH UP ROUTINE
3452 020722 104413              MTPS   #PR7 ;STOP ALL INTERRUPTS
3453 020724 005003              DEVICE.CLR ;CLEAR ALL INTERRUPTS AWAY
3454 020726 005037 001372      CLR    R3
3455 020732 012702 000001      CLR    SAVLIN
3456 020736 030237 001364      MOV    #1,R2
3457 020742 001405              IS:   BIT    R2,LINE ;VALID LINE ?
3458 020744 022763 000400 001442      BEQ    Z$ ;NO
3459 020752 001401              CMP    #400,TR0(R3) ;RECEIVED A BINARY COUNT PATTERN ?
3460 020754 104000              BEQ    .+4 ;YES
3461
3462 020756 005237 001372      ERROR  0 ;THE LINE FAILED TO RECEIVE A FULL
3463
3464
3465
3466
3467 020770 013777 002074 161074      INC    SAVLIN ;BINARY COUNT PATTERN
3468 020776 005077 161072              TST    (R3)+ ;SET UP FOR NEXT LINE
3469 021002 013777 002100 161066      ASLB   R2 ;ADD 2
3470 021010 005077 161064      BCC    IS ;SET UP NEXT LINE POINTER
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493 021020 000004              FINI:  MOV    DZRI$ ,DZRIV ;RESTORE TRAPCATCHER
3494 021022 012737 000032 001122      CLR    DZRI$
3495 021030 012737 000002 18001226      MOV    DZTI$ ,DZTIV
3496 021036 012737 021514 001360      CLR    DZTI$
3497 021044 012737 021170 001362      ADVANCE ;G+80 TO THE NEXT TEST
3498 021052 005037 023140      RXTCR: 0 ;RX IMAGE OF TCR BITS

;***** TEST 32 *****
;DZ11 RELATIVE TIMING TEST.
;EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
;AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
;WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
;DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
;THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
;AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
;PARAMETERS ARE:
; EIGHT BITS/PER/CHAR - TWO STOP BITS AT
; 50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
; 2400, 3600, 4800, 7200, 9600 BAUD.
; 19.2 K BAUD - TWO STOP BITS AT
; SEVEN, SIX, FIVE BITS/PER/CHAR.
;AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
;THE NEXT SELECTED LINE IS THE TESTED.

;: * TEST 32
;*****
†ST32: SCOPE
MOV    #32,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV    #2,STTIMES
MOV    #TST33,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV    #3$,LOCK ;SET FOR LOOP
CLR    OFFSET ;RESET THIS VARIABLE

```



```

3555 021420 105237 001224          INCB  $TMP3          ;DOUBLE COUNT
3556 021424 001366          BNE   4$            ;CONTINUE TEST
3557 021426 104011          ERROR 11           ;INTERRUPTS NOT FINISHED
3558 021430 004737 007242          JSR   PC,SERV.G    ;<IG>?
3559 021434 104401          SCOPI             ;LOOP?
3560 021436 062737 000002 023140 18   ADD   #2,OFFSET
3561 021444 013700 021512          MOV   7$,R0
3562 021450 042700 170377          BIC   #1<17*400>,R0
3563 021454 022700 007400          CMP   #<17*400>,PC
3564 021460 001010          BNE   6$
3565 021462 032737 000030 021512          BIT   #BIT4+BIT3,7$
3566 021470 001610          BEQ   2$
3567 021472 162737 000010 021512          SUB   #BIT3,7$
3568 021500 000633          BR   3$
3569 021502 062737 000400 021512 6$:   ADD   #400,7$
3570 021510 000627          BR   3$
3571 021512 000000          7$:   0
3572                                     ;***** TEST 33 *****
3573                                     ;* THIS TEST VERIFIES THAT EVEN PARITY WORKS
3574                                     ;* FOR ALL ODD LINES SELECTED AND THAT ODD PARITY WORKS FOR ALL
3575                                     ;* EVEN LINES SELECTED.
3576                                     ;* THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
3577                                     ;* THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
3578                                     ;* THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
3579                                     ;* YOU ARE IN "STAGGERED" MODE.
3580                                     ;* 40(B) CHARS ARE USED FOR THIS TEST.
3581                                     ;* ALL SELECTED LINES WILL BE ENABLED
3582                                     ;* AT THE SAME TIME!
3583                                     ;:* TEST 33
3584                                     ;*****
3585 021514 000004          TEST33: SCOPE
3586 021516 012737 000033 001122          MOV   #33,$STNM    ;LOAD THE NUMBER#B OF THIS TEST
3587 021524 012737 022146 001360          MOV   #TS134,NEXT ;POINT TO THE START OF THE NEXT TEST
3588 021532 005737 001370          TST   MODE         ;IS THIS STAGGERED MODE?
3589 021536 100111          BPL   6$           ;IF NOT, DON'T DO THIS TEST
3590 021540 104417          DCLASM             ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3591 021542 013701 001366          MOV   PAR,R1       ;USE R1 TO BUILD PARAMETERS TO BE LOADED
3592 021546 042701 000200          BIC   #00DPAR,R1  ;MAKE SURE ODD PARITY ISN'T SET
3593 021552 052701 000100          BIS   #PARITY,R1  ;MAKE SURE PARITY IS TURNED ON
3594 021556 012702 000001          MOV   #1,R2        ;USE R2 AS A LINE POINTER
3595 021562 030237 00131864          1$:   BIT   R2,LINE   ;IS THIS A VALID LINE?
3596 021566 001411          BEQ   3$           ;IF NOT, SKIP TO THE NEXT LINE
3597 021570 032701 000001          BIT   #BIT0,R1    ;IS THIS LINE AN ODD LINE?
3598 021574 001002          BNE   2$           ;IF IT'S ODD, USE EVEN PARITY
3599 021576 052701 000200          BIS   #00DPAR,R1 ;IF IT'S EVEN, USE ODD PARITY
3600 021602 010177 160244          2$:   MCV   R1,$DZLPR ;LOAD THE LINE PARAMETER REGISTER
3601 021606 042701 000200          BIC   #00DPAR,R1 ;SET UP THE NEXT PARITY TO EVEN
3602 021612 005201          3$:   INC   R1       ;POINT TO THE NEXT LINE
3603 021614 106302          ASLB  R2           ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
3604 021616 103361          BCC   1$           ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
3605 021620 005037 001372          CLR   SAVLIN      ;CLEAR THE LINE NUMBER INDICATOR
3606 021624 005002          CLR   R2          ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
3607 021626 005003          CLR   R3          ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
3608 021630 012737 000040 001376          MOV   #40,XMTCNT  ;TRANSMIT A BINARY COUNT PATTERN(00-40)
3609 021636 012700 001422          MOV   #TDO,R0     ;POINT TO THE DATA AREA
3610 021642 005020          CLR   (R0)+       ;CLEAR A DATA WORD

```

```

3611 021644 022700 001462          CMP      #STOP,RO          ;FINISHED ?
3612 021650 001374          BNE      .-6              ;NO
3613 021652 005000          CLR      RO              ;CLEAR OFFSET
3614 021654 012777 022600 160214    MOV      #XMTSRV,JDZTIV   ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3615 021662 012777 021770 160202    MOV      #9$,JDZRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
3616 021670 013777 026216 160176    MOV      DZPRT,JDZRSIS   ;SET THE INTERRUPT VECTOR STATUS
3617 021676 013777 026216 160174    MOV      DZPRT,JDZTIS    ;SET TRANSMITTER INTERRUPT PRIORITY
3618 021704 052777 040140 160130    BIS      #RIE!TIE!MSENAB,JDZCSR ;ENABLE THE DEVICE
3619 021712 113777 001364 1601836   MOV      LINE,JDZTCR     ;ENABLE ALL SELECTED LINES
3620 021720 106427 000000          MTPS     #0              ;ALLOW INTERRUPTS
3621 021724 005037 021764          4$:     CLR      7$
3622 021730 005037 021766          CLR      8$
3623 021734 032777 000100 160100  5$:     BIT      #RIE,JDZCSR    ;WHEN RX DONE; RIE WILL =0
3624 021742 001407          BEQ      6$              ;BR IF ALL DONE
3625 021744 005237 021764          INC      7$
3626 021750 001371          BNE      5$
3627 021752 105237 021766          INCB     8$
3628 021756 100366          BPL      5$
3629 021760 104011          ERROR    11             ;*RX FAILED TO FINISH (INTERRUPT)
3630 021762 104400          6$:     ADVANCE          ;ADVANCE LOOP
3631 021764 000000          7$:     0
3632 021766 000000          8$:     0
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
    ;RECEIVER SERVICE ROUTINE
    9$:     MOV      JDZRBUF,R4          ;GET THE CHARACTER
    BMI     10$                    ;IF IT WAS VALID, CONTINUE TESTING
    ERROR   ;ERROR- ILLEGAL CHAR... DATA VALID NOT SET
    10$:    MOV      R4,R1              ;COPY THE RECEIVED INFORMATION
    SWAB    R1                      ;GET THE LINE NUMBER IN THE LOWER BYTE
    BIC     #1C<7>,R1                ;ISOLATE THE LINE NUMBER
    ASL     R1                        ;ALIGN IT ON A WORD BOUNDARY
    BIT     #PARER,R4                ;PARITY ERROR SHOULD BE SET. IS IT?
    BNE     11$                      ;IF SO, GO CHECK CHARACTER
    MOV     DZRBUF,REGIST            ;SET UP FOR THE ERROR MESSAGE
    MOV     R4,R5
    BIC     #377,R5
    BISB    TRO(R1),R5              ;GET THE CORRECT CHARACTER
    BIS     #DVALID!PARER,R5        ;BUILD WHAT WAS EXPECTED
    ERROR   6                        ;*ERROR- DID NOT GET CORRECT INFORMATION
    11$:    CMPB    TRO(R1),R4         ;CHECK THE CHARACTER. IS IT CORRECT?
    BEQ     12$                      ;IF SO, GO SET UP NEXT CHARACTER
    MOV     TRO(R1)+B,R5            ;LOAD THE CHARACTER FOR ERROR REPORTING
    BIC     #1C<377>,R5              ;CLEAR SIGN EXTEND
    MOV     R1,SAVLIN                ;GET THE LINE NUMBER FOR REPORTING
    ASR     SAVLIN                    ;ALIGN IT CORRECTLY
    BIC     #1C<377>,R4              ;REMOVE THE JUNK FROM R4, THE ACTUAL CHARACTER
    ERROR   5                        ;DATA ERROR
    12$:    INC     TRO(R1)            ;SET UP THE NEXT CHARACTER
    INC     R3                        ;ADD TO THE TOTAL RECEIVED COUNT
    BIT     #TIE,JDZCSR              ;ARE TRANSMISSIONS DONE?
    BNE     13$                      ;IF NO, GO RECEIVE SOME MORE
    CMP     R2,R3                    ;ARE ALL CHARACTERS RECEIVED?
    BNE     13$                      ;IF NO, GO RECEIVE SOME MORE
    BIC     #RIE,JDZCSR              ;DISABLE RECEIVER INTERRUPTS
    
```

G08

MD-11-DZDZA-C MACY:1 27(1006) 21-OCT-76 13:09 PAGE 74
DZDZAC.P11 21-OCT-76 13:07

DZ11 DEVICE DIAGNOSTICS.

COPYRIGHT 1976 DIGITAL EQUIP. CORP.

PAGE: 0097

```

3667 022132 012716 021762      MOV      #6$, (SP)      ;CRUNCH THE STACK
3668 022136 000002      RTI              ;RETURN AND FINISH
3669 022140 012716 021724      13$: MOV      #4$, (SP)      ;CRUNCH THE STACK
3670 022144 000002      RTI              ;GO BACK TO RECEIVER WAIT L+BCOP
3671                                     ;***** TEST 34 *****
3672                                     ;*THIS TEST VERIFIES THAT ODD PARITY WORKS FOR ALL ODD LINES
3673                                     ;* SELECTED AND THAT EVEN PARITY WORKS FOR ALL EVEN LINES SELECTED
3674                                     ;*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
3675                                     ;*THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
3676                                     ;*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
3677                                     ;*YOU ARE IN "STAGGERED" MODE.
3678                                     ;*40(8) CHARS ARE USED FOR THIS TEST.
3679                                     ;*ALL SELECTED LINES WILL BE ENABLED
3680                                     ;*AT THE SAME TIME!
3681 ::* TEST 34
3682 :*****B*****
3683 022146 000004      †S34: SCOPE
3684 022150 012737 000034 001122      MOV      #34, $STSTNM    ;LOAD THE NUMBER OF THIS TEST
3685 022156 012737 004444 001360      MOV      #SEOP, NEXT     ;POINT TO THE END-OF-PASS HANDLER
3686 022164 005737 001370      TST      MODE            ;IS THIS STAGGERED MODE?
3687 022170 100111      BPL      6$              ;IF NOT, DON'T DO THIS TEST
3688 022172 104417      DCLASM                    ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3689 022174 013701 001366      MOV      PAR, R1         ;USE R1 TO BUILD PARAMETERS TO BE LOADED
3690 022200 042701 000200      BIC      #ODDPAR, R1     ;MAKE SURE ODD PARITY ISN'T SET
3691 022204 052701 000100      BIS      #PARITY, R1     ;MAKE SURE PARITY IS TURNED ON
3692 022210 012702 000001      MOV      #1, R2          ;USE R2 AS A LINE POINTER
3693 022214 030237 001364      1$: BIT      R2, LINE     ;IS THIS A VALID LINE?
3694 022220 001411      BEQ      3$              ;IF NOT, SKIP TO THE NEXT LINE
3695 022222 032701 000001      BIT      #BIT0, R1       ;IS THIS LINE AN ODD LINE?
3696 022226 001402      BEQ      2$              ;IF IT'S EVEN, USE EVEN PARITY
3697 022230 052701 000200      BIS      #ODDPAR, R1     ;IF IT'S ODD, USE ODD PARITY
3698 022234 010177 157612      2$: MOV      R1, @DZLPR    ;LOAD THE LINE PARAMETER REGISTER
3699 022240 042701 000200      BIC      #ODDPAR, R1     ;SET UP THE NEXT PARITY TO EVEN
3700 †B 022244 005201      3$: INC      R1           ;POINT TO THE NEXT LINE
3701 022246 106302      ASLB     R2              ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
3702 022250 103361      BCC     1$              ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
3703 022252 005037 001372      CLR     SAVLIN          ;CLEAR THE LINE NUMBER INDICATOR
3704 022256 005002      CLR     R2              ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
3705 022260 005003      CLR     R3              ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
3706 022262 012737 000040 001376      MOV     #40, XMTCNT     ;TRANSMIT A BINARY COUNT PATTERN(00-40)
3707 022270 012700 001422      MOV     #TDO, RO        ;POINT TO THE DATA AREA
3708 022274 005020      CLR     (RO)+           ;CL+BEAR A DATA WORD
3709 022276 022700 001462      CMP     #STOP, RO       ;FINISHED "
3710 022302 001374      BNE     -6              ;NO
3711 022304 005000      CLR     RO              ;CLEAR OFFSET
3712 022306 012777 022600 157562      MOV     #XMTSRV, @DZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3713 022314 012777 022422 157550      MOV     #9$, @DZRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
3714 022322 013777 026216 157544      MOV     DZPRT, @DZRIIS ;SET THE INTERRUPT VECTOR STATUS
3715 022330 013777 026216 157542      MOV     DZPRT, @DZTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
3716 022336 052777 040140 157476      BIS     #RIE!+IE!MSENAB, @DZCSR ;ENABLE THE DEVICE
3717 022344 113777 001364 157504      MOV     LINE, @DZTCR    ;ENABLE ALL SELECTED LINES
3718 022352 106427 000000      MTPS    #0              ;ALLOW INTERRUPTS
3719 022356 005037 022416      4$: CLR     7$
3720 022362 005037 022420      CLR     8$
3721 022366 032777 000100 157446      5$: BIT     #RIE, @DZCSR  ;WHEN RX DONE: PIE WILL =0
3722 022374 001407      BEQ     6$              ;BR IF ALL DONE

```



```

3769
3770
3771
3772
3773 022600 117701 157240 XMTSRV: MOVB @DZCSR,R1 ;GET THE LINE NUMBER. IS THE TRANSMITTER
3774 022604 100401 BMI 1$ ;REALLY READY? IF SO, GO LOAD THE CHARACTER
3775 022606 104003 ERROR 3 ;*TRANSMITTER NOT READY- FALSE INTERRUPT
3776 022610 042701 177770 1$: BIC #1C<7>,R1 ;ISOLATE THE LINE NUMBER
3777 022614 006301 ASL R1 ;MAKE SURE IT REFERENCES A WORD BOUNDARY
3778 022616 116177 001422 157242 MOVB TDO(R1),@DZTDR ;LOAD THE CURRENT CHARACTER FOR THIS LINE
3779 022624 005261 001422 INC TDO(R1) ;SET UP NEXT CHARACTER FOR THIS LINE
3780 022630 005202 INC R2 ;UP THE NUMBER OF TRANSMISSIONS
3781 022632 023761 001376 001422 CMP XMTCNT,TDO(R1) ;HAVE WE DONE ALL PATTERNS ON THIS LINE?
3782 022640 001015 BNE 4$ ;IF NOT, KEEP ON TRANSMITTING
3783 022642 012700 000001 MOV #1,R0 ;SET UP A DESELECTION POINTER
3784 022646 006201 ASR R1 ;GET THE LINE NUMBER AGAIN
3785 022650 005301 2$: DEC R1 ;REDUCE THE COUNT. WAS THIS THE LINE?
3786 022652 100402 BMI 3$ ;IF SO, GO DISABLE THE ENABLE BIT FOR IT
3787 022654 006300 ASL R0 ;MOVE THE POINTER TO THE NEXT LINE
3788 022656 000774 BR 2$ ;GO CHECK THE NEXT LINE
3789 022660 140077 157172 3$: BICB R0,@DZTCR ;DISABLE THE LINE POINTED TO BY R0
3790 022664 001003 BNE 4$ ;IF MORE LINES ARE ACTIVE, GO CONTINUE TRANSMIT
3791 022666 042777 040000 157146 BIC #TIE,@DZCSR ;IF NOT, DISABLE TRANSMITTER INTERRUPTS
3792 022674 000002 4$: RTI ;RETURN TO THE TIMING LOOP
    
```

```

3793
3794
3795
3796
3797 022676 012737 00001804 001222 BUILD: MOV #4,$TMP2 ;ROTATE 4 BITS BACK INTO $TMP1
3798 022704 006037 001224 1$: ROR $TMP3 ;GET THE BITS FROM $TMP3, THE HIGH BYTE
3799 022710 006037 001220 ROR $TMP1 ;OF THE RELATIVE TIME COUNTER. PUT THEM BACK
3800 022714 005337 001222 DEC $TMP2 ;INTO $TMP1 USING THE CARRY BIT WITH
3801 ;ROTATE INSTRUCTIONS
3802 022720 001371 BNE 1$ ;REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
3803 022722 000207 RTS PC ;RETURN TO CALLING TEST
3804
    
```

```

3805                                     ;RECEIVER SERVICE ROUTINE
3806
3807 022724 105777 157112      RXISR1: TSTB   QDZCSR      ; IS THE RECEIVER REALLY READY?
3808 022730 100401              BMI     1$          ; IF SO, GO SERVICE IT
3809 022732 104004              ERROR   4            ; *ERROR- RECEIVER DONE FLAG ISN'T SET
3810 022734 017704 157106      1$:   MOV    QDZRBUF,R4    ; SAVE THE RECEIVER INFORMATION
3811 022740 100401              BMI     2$          ; IF IT WAS VALID, GO PROCESS IT
3812 022742 104000              ERROR   2$          ; ERROR- DATA VALID WASN'T SET
3813 022744 032704 070000      2$:   BIT    #OVRRUN!FRMERR!PARER,R4 ; ARE ANY ERROR FLAGS SET?
3814 022750 001404              BEQ    3$          ; IF NOT, GO CONTINUE PROCESSING
3815 02182752 013737 072046 001400 MOV    DZRBUF,REGIST ; SET UP FOR ERROR REPORTING
3816 022760 104002              ERROR   2            ; ERROR- RECEIVER ERROR FLAG SET
3817 022762 010401              3$:   MOV    R4,R1      ; COPY THE RECEIVER INFORMATION
3818 022764 000301              SWAB   R1          ; GET THE LINE NUMBER IN THE LOWER BYTE
3819 022766 042701 177770      BIC    #1C<7>,R1   ; ISOLATE THE LINE NUMBER
3820 022772 006301              ASL    R1          ; ALIGN IT ON A WORD BOUNDARY
3821 022774 120461 001442      CMPB   R4,TR0(R1)  ; IS THE CHARACTER WHAT IT SHOULD BE?
3822 023000 001413              BEQ    4$          ; IF SO, GO CONTINUE PROCESSING
3823 023002 116105 001442      MOVB   TR0(R1),R5  ; GET WHAT WAS EXPECTED FOR ERROR+R REPORTING
3824 023006 042705 177400      BIC    #1C<377>,R5 ; ELIMINATE PROPAGATED SIGN
3825 023012 042704 177400      BIC    #1C<377>,R4 ; ISOLATE THE ACTUAL CHARACTER
3826 023016 010137 001372      MOV    R1,SAVLIN  ; GET THE LINE NUMBER OF THE RECEIVER ERROR
3827 023022 006237 001372      ASR    SAVLIN      ; ALIGN IT CORRECTLY FOR REPORTING
3828 023026 104005              ERROR   5            ; *DATA ERROR
3829 023030 005261 001442      4$:   INC    TR0(R1)   ; SET UP THE NEXT EXPECTED CHARACTER
3830 023034 005203              INC    R3          ; INCREMENT THE COUNT OF RECEIVED CHARACTERS
3831 023036 032761 000020 001442 BIT    #20,TR0(R1) ; HAVE ALL CHARACTERS BEEN RECEIVED?
3832 023044 001402              BEQ    5$          ; IF NOT, GO RECEIVE SOME MORE
3833 023046 020203              CMP    R2,R3      ; HAVE WE RECEIVED ALL CHARACTERS?
3834 023050 001401              BEQ    6$          ; IF SO, GO DETERMINE THE TIMING
3835 023052 000002      5$:   RTI                    ; GO CONTINUE TIMING AND ALLOW INTERRUPTS
3836 023054 004737 022676      6$:   JSR    PC,BUILD   ; GET THE RELATIVE TIME (SIGNIFICANT BITS)
3837
3838 023060 013700 023140      MOV    OFFSET,R0   ; GET POINTER
3839 023064 013760 001220 002102 MOV    $TMP1,TMTBL(R0) ; SAVE THIS TEST'S TIME
3840 023072 005737 023140      TST    OFFSET      ; FIRST TEST?
3841 023076 001414              BEQ    7$          ; IF NOT, GO CHECK THE TIME
3842 023100 005740              TST    -(R0+8)    ; POINT TO THE PREVIOUS TIME TAKEN
3843 023102 026037 002102 001220 CMP    TMTBL(R0),$TMP1 ; IS THIS TIME WHAT IT SHOULD BE?
3844 023110 101007              BHI    7$          ; IF SO, GO TO THE NEXT TEST
3845 023112 016005 002102      MOV    TMTBL(R0),R5 ; PLACE WHAT WAS EXPECTED IN R5
3846 023116 010137 001372      MOV    R1,SAVLIN  ; GET THE LINE NUMBER OF THE RECEIVER
3847 023122 006237 001372      ASR    SAVLIN      ; MAKE SURE IT'S THE LINE NUMBER
3848 023126 104021              ERROR   21         ; TIMING ERROR
3849 023130 042777 000140 156704 7$:   BIC    #RIE!MSENAB,QDZCSR ; DISABLE THE DEVICE
3850 023136 000002              RTI                    ; RETURN TO THE PROGRAM
3851 023140 000000      OFFSE1BT: 0
    
```

3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907

;DZ11 ECHO/CABLE TEST
;COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

;*STARTING PROCEDURE
;*LOAD PROGRAM
;*LOAD ADDRESS 000210
;*PRESS START
;*PROGRAM WILL TYPE DZ11 ECHO/CABLE TEST
;*PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
;*TYPE IN E OR C RESPECTIVELY
;*PROGRAM WILL TYPE "VECTOR ADDRESS-"
;*TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
;*PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"
;*TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
;*FOR THE DZ11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
;*PROGRAM WILL TYPE "LINE NUMBER-"
;*TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
;* FOLLOWED BY <CARRIAGE RETURN>
;*PROGRAM WILL TYPE "BAUD RATE-"
;*TYPE IN THE BAUD RATE OF THE DZ11 TERMINAL
;* FOLLOWED BY <CARRIAGE RETURN>
;*THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL

* 50
* 75
* 110
* 135 (ROUNDED OFF 134.5)
* 150
* 300
* 600
* 1200
* 1800
* 2000
* 2400
* 3600
* 4800
* 7200
* 9600

;*ALL OTHERS ARE REJECTED

;*PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTED

;PROGRAM INITIALIZATION
;LOCK OUT INTERRUPTS
;SET UP PROCESSOR STACK
;SET UP POWER FAIL VECTOR
;CLEAR PROGRAM FLAGS AND COUNTS

XSTART: MOV #STACK, SP ;SET UP PROCESSOR STACK
MTPS #PR7 ;LOCK OUT INTERRUPTS
MOV #XSTART, \$LPADR ;SET UP IN CASE OF POWER FAIL
CLR STFLG ;CLEAR TEST START FLAG
CLR \$PASS ;CLEAR PASS COUNT
CLR \$ERTTL ;CLEAR ERROR COUNT
CLRB \$ERFLG ;CLEAR ERROR FLAG

023142 012706 001120
023146 106427 000340
023152 012737 023142 001126
023160 005037 025322
023164 005037 001242
023170 005037 001132
023174 105037 001123

```

3908 023200 005037 025326          CLR      LAST          ;CLEAR LAST ERROR PC
3909 023204 032777 000001 155746 VEC1:  BIT      #SW00,DSWR  ;IF SW00=1, GET NEW VECTOR
3910 023212 001465          BEQ      OTHER        ;AND CSR
3911 023214 012701 000300          VEC2:  MOV      #300,R1
3912 023220 012702 000302          MOV      #302,R2
3913 023224 010221          1$:    MOV      R2,(R1)+  ;RESTORE TRAPCATCHER
3914 023226 005022          CLR      (R2)+        ;IN FLOATING VECTOR AREA
3915 023230 022122          CMP      (R1)+,(R2)+ ;UPDATE THE POINTERS
3916 023232 020127 001000          CMP      R1,#1000
3917 023236 001372          BNE      ↑B1$
3918 023240 104403          INSTR          ;INPUT ADDRESS OF DEVICE VECTOR
3919 023242 025354          MVECTOR        ;MESSAGE "VECTOR ADDRESS-"
3920 023244 104405          PARAM         ;CONVERT STRING TO OCTAL
3921 023246 000300          300           ;LOW LIMIT
3922 023250 000770          770           ;HIGH LIMIT
3923 023252 002072          DZRIV         ;LOCATIONS TO BE FILLED
3924 023254 003          .BYTE 3       ;LSB MASK
3925 023255 004          .BYTE 4       ;NUMBER OF LOCATIONS
3926 023256 104403          INSTR          ;INPUT ADDRESS OF DEVICE CSR
3927 023260 025376          MREGAD        ;MESSAGE "CONTROL REGISTER ADDRESS-"
3928 023262 104405          PARAM         ;CONVERT STRING TO OCTAL
3929 023264 1600↑800          160000       ;LOW LIMIT
3930 023266 163700          163700       ;HIGH LIMIT
3931 023270 002042          DZCSR         ;LOCATIONS TO BE FILLED
3932 023272 007          .BYTE 7       ;LSB MASK
3933 023273 001          .BYTE 1       ;NUMBER OF LOCATIONS
3934 023274 013737 002042 002046          MOV      DZCSR,DZRBUF ;BEGIN BUILDING DEVICE ADDRESSES
3935 023302 062737 000002 002046          ADD      #2,DZRBUF    ;FORM THE READ BUFFER ADDRESS
3936 023310 013737 002046 002052          MOV      DZRBUF,DZLPR ;REMEMBER THAT THIS IS ALSO LINE PARAMETER REG.
3937 023316 013737 002046 002056          MOV      DZRBUF,DZTCR ;BEGIN BUILDING TRANSMITTER CONTROL REGISTER
3938 023324 062737 000002 002056          ADD      #2,DZTCR    ;FORM THE TRANSMITTER CONTROL REGISTER POINTER
3939 023332 013737 002056 002060          MOV      DZTCR,HDZTCR
3940 023340 005237 002060          INC      HDZTCR
3941 023344 013737 002056 002066          MOV      DZTCR,DZTDR ;BEGIN FORMING TRANSMITTER DATA REGISTER
3942 023352 062737 000002 002066          ADD      #2,DZTDR    ;FORM THE TRANSMITTER DATA REGISTER
3943 023360 013737 002066 002062          MOV      DZTDR,DZMSR
3944 023366 032777 000002 155564 OTHER: BIT      #SW01,DSWR  ;RESELECT OF TEST?
3945 023374 001427          BEQ      XBEGIN     ;IF NOT, SKIP ASKING WHICH ONE
3946 023376 104403          INSTR          ;INPUT WHICH TEST YOU ARE RUNNING
3947 023400 025562          MWHICH        ;ECHO OR C@BBLE
3948 023402 104416          PAWCH        ;SET FLAG
3949 023404 025320          WCHFLG       ;THIS FLAG
3950 023406 104403          BAUD: INSTR    ;INPUT BAUD RATE
3951 023410 025504          MSPEED       ;MESSAGE "BAUD RATE-"
3952 023412 104415          PARMD        ;CONVERT DECIMAL STRING TO OCTAL
3953 023414 000062          50           ;LOW LIMIT
3954 023416 022600          9600         ;HIGH LIMIT
3955 023420 025336          LINESP       ;LOCATION TO BE FILLED
3956 023422 000          .BYTE 0       ;LSB MASK
3957 023423 001          .BYTE 1       ;NUMBER OF LOCATIONS
3958 023424 104413          LINEX:        ;CLEAR DEVICE
3959 023426 005037 025322          CLR      STFLG     ;CLEAR PROGRAM START FLAG
↑83960 023432 104403          INSTR          ;INPUT LINE NUMBER
3961 023434 025474          MLINE        ;MESSAGE "LINE NUMBER-"
3962 023436 104405          PARAM         ;CONVERT STRING TO OCTAL
3963 023440 000000          0           ;LOW LIMIT

```

M08

MD-11-DZDZA-C MACY11 27(1006) 21-OCT-76 13:09 PAGE 80
 DZDZAC.P11 21-OCT-76 13:07

DZ11 DEVICE DIAGNOSTICS. COPYRIGHT 1976 DIGITAL EQUIP. CORP.

PAGE: 0103

```

3964 023442 000007          7          ;HIGH LIMIT
3965 023444 001372          SAVLIN      ;LOCATION TO BE FILLED
3966 023446          000      .BYTE 0      ;LSB MASK
3967 023447          001      .BYTE 1      ;NUMBER OF LOCATIONS
3968 023450 004537 025124    JSR        R5,SET
3969
3970 023454 106427 000340    XBEGIN: MTPS  #PR7      ;LOCK OUT INTERRUPTS
3971 023460 012706 001120    MOV        #STACK,SP  ;SET UP PROCESSOR STACK
3972 023464 005037 025324    CLR        LOCKUP     ;CLEAR TIMEOUT
3973 023470 005737 025320    TST        WCHFLG     ;ECHO OR CABLE TEST ?
3974 023474 001413          BEQ        2$         ;ECHO
3975 023476 012737 024176 001126  MOV        #TEST2,$LPADR ;CABLE TEST
3976 023504 005737 025322    TST        STFLG      ;ARE YOU LOOPING ?
3977 023510 001017          BNE        1$         ;YES
3978 023512 005137 025322    COM        STFLG      ;NO
3979 023516 104402 025655    TYPE      ,MCABLE    ;TYPE CABLE TEST
3980 023522 000412          BR         1$
3981 023524 012737 023554 001126 2$: MOV        #TEST1,$LPADR ;SET U+BP ECHO TEST
3982 023532 005737 025322    TST        STFLG      ;ARE YOU LOOPING ?
3983 023536 001004          BNE        1$         ;YES
3984 023540 005137 025322    COM        STFLG      ;NO
3985 023544 104402 025630    TYPE      ,MTERM     ;TYPE ECHO TEST
3986 023550 000177 155352    1$: JMP        2$LPADR   ;START TESTING
3987
3988          ;THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
3989          ;((IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
3990          ;ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
3991 023554 104413          TEST1: DEVICE.CLR    ;CLEAR DZ11
3992 023556 012737 000001 001122  MOV        #1,$STNM
3993 023564 013777 025344 156264  MOV        NUMTCR,$DZTCR ;SET T+BCR BIT
3994 023572 013737 025342 001366  MOV        NUMLIN,PAR   ;SET PARAMETERS
3995 023600 053737 025340 001366  BIS        SPEED,PAR    ;SET BAUD RATE
3996 023606 013777 001366 156236  MOV        PAR,$DZLPR   ;LOAD PARAM.
3997 023614 012777 000040 156220  MOV        #MSENAB,$DZCSR ;SET SCANN ENABLE
3998 023622 005004          CLR        R4
3999 023624 012705 025672          MOV        #MQUICK,R5  ;SET MESSAGE BUFFER
4000 023630 005777 156206 3$: TST        $DZCSR     ;TRDY?
4001 023634 100404          BMI        2$         ;BR IF YES
4002 023636 104414          DELAY
4003 023640 005304          DEC        R4
4004 023642 001372          BNE        3$
4005 023644 104003          ERROR     3          ;NO TRDY SET! WHY?
4006 023646 005004          CLR        R4        ;RESET COUNTER TO 0
4007 023650 112577 156212 2$: MOV        (R5)+,$DZTDR ;LOAD CHAR
4008 023654 001365          BNE        3$
4009 023656 004737 007242          JSR        PC,SERV.G   ;<↑G>?
4010 023662 122777 000377 155270  CMP        #377,$SWR   ;RE-DO QUICK BROWN?
4011 023670 001731          BEQ        TEST1     ;BR IF REPEAT PATTERN
4012 023672 104413          DEVICE.CLR
4013 023674 106427 000340          MTPS      #PR7      ;LOCK OUT INTERRUPTS
4014 023700 012737 024634 001360  MOV        #XEOP,NEXT
4015 023706 104413          DEVICE.CLR
4016 023710 013737 025342 001366  MOV        NUMLIN,PAR  ;SELECT LINE # & SET INTERRUPT ENABLE
4017 023716 053737 025340 00131866  BIS        SPEED,PAR  ;SET LINE SPEED AND
4018          ;CHARACTER LENGTH (TRANS. & REC.)
4019 023724 052737 010000 001366  BIS        #RCVON,PAR ;MAKE SURE RECEIVER IS TURNED ON
  
```

```

4020 023732 013777 001366 156112      MOV    PAR,ADZLPR      ;LOAD THE LINE PARAMETER REGISTER
4021 023740 012777 024014 156124      MOV    #INTSVC,ADZRIV ;SET UP INTERRUPT SERVICE
4022 023746 013777 025346 156120      MOV    PRIO,ADZRIS    ;AND LEVEL
4023 023754 106437 026220                MTPS   #LESSL        ;ALLOW INTERRUPTS
4024 023760†B 012777 000140 156054      MOV    #RIE!MSENAB,ADZCSR ;SET RECEIVER INTERRUPT ENABLE
4025 023766 104402 025522                TYPE   MCHAR          ;TYPE "ANY CHARACTER"
4026 023772 105777 155166      1$:   TSTB   #STKS      ;IF SOMEBODY HITS A KEY- GET NEW LINE #
4027 023776 100375                BPL    1$            ;LOOP HERE
4028 024000 005777 155162                TST   #STKB         ;CLEAR CHAR
4029 024004 004737 007242                JSR   PC,SERV.G     ;MAKE SURE IT WASN'T <†G>
4030 024010 000137 023424                JMP   LINEX         ;
4031
4032
4033

```

```

4034 024014 105777 156022      INTSVC: ;THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
4035 024020 100401                †TSTB  ADZCSR        ;TEST REC. FLAG
4036 024022 104004                BMI   .+4
4037 024024 017737 156016 025350      ERROR  4            ;ERROR - INTERRUPT NOT CAUSED BY FLAG
4038 024032 100401                MOV   ADZRBUF,RECDAT
4039 024034 104023                BMI   .+4
4040 024036 032737 020000 025350      ERROR  23          ;NON- VALID CHARACTER
4041 024044 001401                BIT   #BIT13,RECDAT ;CHECK FOR FRAMING ERROR
4042 024046 104025                BEQ   .+4           ;BR IF NO ERROR
4043                                ERROR  25          ;EITHER SOMEBODY HIT THE
4044                                ;"BREAK KEY" OR YOU HAVE AN ERROR!
4044 024050 113737 025350 025352      MOVB  RECDAT,TBUF    ;MOVE CHARACTER TO OUTPUT AREA
4045 024056 113737 025350 010502      MOVB  RECDAT,INBUF   ;MOVE CHARACTER TO CHECK FOR †C
4046 024064 042737 17760†B0 010502  BIC   #†C<177>,INBUF ;STRIP JUNK PLUS PARITY
4047 024072 042737 174377 025350      BIC   #174377,RECDAT ;SAVE ONLY LINE NUMBER
4048 024100 000337 025350                SWAB  RECDAT
4049 024104 023737 001372 025350      CMP   SAVLIN,RECDAT ;DOES THE LINE # COMPARE?
4050 024112 001401                BEQ   .+4
4051 024114 104015                ERROR  15          ;*WRONG LINE NUMBER
4052 024116 013777 025344 155732      MOV   NUMTCR,ADZTCR ;ENABLE THE LINE TO TRANSMIT
4053 024124 012777 000040 155710      MOV   #MSENAB,ADZCSR ;START THE TRANSMITTERS SCANNER
4054 024132 123727 010502 000003      CMPB  INBUF,#3      ;IS IT A †C ?
4055 024140 001004                BNE   1$           ;NO
4056 024142 10441†B3                DEVICE.CLR
4057 024144 012716 024634                MOV   #XEOP,(SP)   ;CRUNCH STACK
4058 024150 000002                RTI
4059 024152 013777 025344 155676      1$:   MOV   NUMTCR,ADZTCR ;ENABLE THE LINE
4060 024160 113777 025352 155700      MOVB  TBUF,ADZTOR   ;TRANSMIT THE CHARACTER
4061 024166 012777 000140 155646      MOV   #RIE!MSENAB,ADZCSR ;RESTART THE RECEIVER
4062 024174 000002                RTI
4063
4064
4065

```

```

;THIS TEST TRANSMITS A BINARY COUNT PATTERN
;VIA INTERRUPT MODE TO THE RECEIVER
;THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
†TEST2: MTPS   #PR7            ;DISABLE INTERRUPTS
        MOV   #2,$STNM
        MOV   #XEOP,NEXT
        DEVICE.CLR
        ;*TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
        ;*WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
        ;*THE DIST PNL MUST HAVE JUMPER FROM DTR TO ROST TO SEND
        ;*IN ORDER FOR THIS TEST TO WORK!

```

```

4066
4067
4068 024176 106427 000340
4069 024202 012737 000002 001122
4070 024210 012737 024634 001360
4071 024216 104413
4072
4073
4074
4075

```

```

4076 024220 012737 024226 001362      MOV      #15,LOCK      ;LOOP
4077 024226 113777 025344 155624 15:  MOV#B NUMTCR, @DZTCR ;SET DTR
4078 024234 005005          CLR      R5           ;
4079 024236 153705 025344          BISB    NUMTCR,R5     ;BUILD EXPECTED
4080 024242 000305          SWAB    R5           ;PUT IN HIGH BYTE
4081 024244 153705 025344          BISB    NUMTCR,R5     ;
4082 024250 104414          DELAY           ;WAIT FOR CABLE DELAY
4083 024252 017704 155604          MOV     @DZMSR,R4    ;READY MODEM BITS
4084 024256 020504          CMP     R5,R4       ;ARE THEY OK?
4085 024260 001401          BEQ    25          ;BR IF YES
4086 024262 104022          ERROR   22         ;IS THE TEST CONNECTOR ON?
4087          ;HAS RIGHT LINE BEEN SELECTED?
4088          ;IF SO- YOU HAVE A PROBLEM!
4089          ;MODEM BITS NOT RIGHT
4090 024264 #B104401          25:    SCOP1      ;LOOP
4091 024266 104413          35:    DEVICE.CLR ;INIT DZ11
4092 024270 013737 025340 001366      MOV     SPEED,PAR    ;SET LINE SPEED
4093 024276 053737 025342 001366      BIS     NUMLIN,PAR   ;SELECT LINE # & REC. INTERRUPT ENABLE
4094 024304 052737 010000 001366      BIS     #RCVON,PAR   ;ENABLE THE RECEIVER FOR THIS LINE
4095 024312 052777 040140 155522      BIS     #TIE!RI!MSENAB,@DZCSR ;SET TRANSMITTER INTERRUPT ENABLE
4096 024320 012777 024434 155544      MOV     #INTREC,@DZRV ;SET UP INTR SERVICE
4097 024326 013777 025346 155540      MOV     PRIO,@DZRV  ;SET UP LEVEL
4098 024334 012777 024614 155534      MOV     #INTRAN,@DZTIV ;SET UP INTR SERVICE
4099 024342 013777 025346 155530      MOV     PRIO,@DZTIS  ;SET UP LEVEL
4100 024350 005001          CLR     R1          ;RX DATA POINTER- SET TO 0
4101 024352 005002          CLR     R2          ;TX DATA POINTER- SET TO 0
4102 024354 013777 001366 155470      MOV     PAR,@DZLPR   ;SET THE PARAMETERS AND TURN ON RECEIVER
4103 024362 106437 026220          MTPS   @LSS1       ;ALLOW INTERRUPTS
4104 024366 013777 025344 155462      MOV     NUMTCR,@DZTCR ;SET UP TCR BIT
4105
4106          ;YOU RETURN HERE AFTER EVERY RECEIVER INTERRUPT
4107 024374 105777 154564      SPIN:  TSTB   @STKS    ;IF SOMEBODY HITS A KEY- GET A NEW LINE #
4108 024400 100006          BPL     15         ;BR IF NO KEY HIT
4109 02440#B2 005777 154560          TST    @STKB      ;CLEAR CHAR
4110 024406 004737 007242          JSR    PC,SERV.G   ;MAKE SURE IT WASN'T (↑G)
4111 024412 000137 023424          JMP    LINEX       ;SW02=1
4112 024416 005237 025324      15:    INC     LOCKUP   ;INC TIMEOUT FLAG
4113 024422 001364          BNE    SPIN        ;IF NOT 0 RETURN SPINNING
4114 024424 104011          ERROR  11         ;*RECEIVER FAILED TO INTERRUPT CHECK CABLE, TERMINATOR
4115 024426 104413          QUIT:  DEVICE.CLR ;
4116 024430 000137 024634          JMP    XEOP        ;CALL FOR END OF PASS
4117 024434 005037 025324      INTREC: CLR    LOCKUP ;CLEAR TIMEOUT FLAG
4118 024440 105777 155376          TSTB   @DZCSR      ;TEST REC DONE
4119 024444 100401          BMI#B .+4         ;YES
4120 024446 104004          ERROR  4          ;*FALSE INTERRUPT
4121 024450 017737 155372 025350      MOV     @DZRBUF,RECDAT ;SAVE WORD
4122 024456 100401          BMI   .+4         ;
4123 024460 104023          ERROR  23         ;*NON VALID CHARACTER
4124 024462 032737 040000 025350      BIT     #BIT14,RECDAT ;DATA OVERRUN ?
4125 024470 001401          BEQ    .+4         ;NO
4126 024472 104024          ERROR  24         ;*YES
4127 024474 032737 020000 025350      BIT     #BIT13,RECDAT ;FRAMING ERROR ?
4128 024502 001401          BEQ    .+4         ;NO
4129 024504 104025          ERROR  25         ;*YES
4130 024506 032737 010000 025350      BIT     #BIT12,RECDAT ;PARITY ERROR ^
4131 024514 001401          BEQ    .+4         ;NO

```

4132	024516	104026			ERROR	26		: *YES
4133	024520	110105			MOVB	R1, R5		: SET EXPECTED
4134	024522	042705	177400		BIC	#1C(377), R5		: CLEAR HIGH BYTE
4135	024526	113704	025350		MOVB	RECDAT, R4		: GET FOUND
4136	024532	042704	177400		BIC	#1C(377), R4		: CLEAR HIGH BYTE
4137	024536	020504			CMP	R5, R4 ; OK?		
4138	024540	001401			BEQ	.+4		
4139	024542	104005			ERROR	5		: DATA ERROR
4140	024544	042737	174377	025350	BIC	#174377, RECDAT		: SAVE ONLY LINE NUMBER
4141	024552	00180337		025350		SWAB RECDAT		
4142	024556	023737	001372	025350	CMP	SAVLIN, RECDAT		: DOES THE LINE # COMPARE ?
4143	024564	001401			BEQ	.+4		: YES
4144	024566	104015			ERROR	15		: *WRONG LINE #
4145	024570	120127	000377		CMPB	R1, #377		: LAST CHARACTER ?
4146	024574	001003			BNE	15		: NO
4147	024576	012716	024426		MOV	#QUITS, (SP)		: CRUNCH STACK
4148	024602	000403			BP	25		
4149	024604	105201			15:	INCB R1		: UPDATE EXPECTED DATA
4150	024606	012716	024374		MOV	#SPIN, (SP)		: CRUNCH STACK
4151	024612	000002			25:	RTI		
4152								
4153	024614	005777	155222		INTRAN: TST	#DZCSR ; TEST TRANSMIT FLAG		
4154	024620	100401		18	BMI	.+4		
4155	024622	104003			ERROR	3		: *FALSE INTERRUPT
4156	024624	110277	155236		MOVB	R2, #DZTDR		: TRANSMIT A CHARACTER
4157	024630	105202			INCB	R2		: UPDATE TX DATA
4158	024632	000002			RTI	: RETURN		

4159					
4160					:END OF PASS
4161					:RESTART TEST
4162					
4163	024634	104402			XEOP: TYPE ;TYPE NAME OF TEST
4164	024636	025432			MPASS
4165	024640	005037	025326		CLR LAST ;CLEAR LAST ERROR PC
4166	024644	105037	001123		CLRB SERFLG ;CLEAR ERROR FLAG
4167	024650	000137	023454		RSTRT: JMP XBEGIN
4168					
4169					:CONVERT DECIMAL ASCII STRING TO OCTAL
4170	024654	011605			.PARMD: MOV (SP),R5
4171	024656	012537	025040		MOV (R5)+,6\$
4172	024662	012537	025042		MOV (R5)+,7\$
4173	024666	012537	025044		MOV (R5)+,8\$
4174	024672	112537	025046		MOV (R5)+,9\$
4175	024676	112537	025047		MOV (R5)+,10\$
4176	024702	010516			MOV R5,(SP)
4177	024704	005005			2\$: CLR R5
4178	024706	012704	010502		MOV #INBUF,R4
4179	024712	122714	000C15		CMPB #15,(R4)
4180	024716	001424			BEQ 3\$
4181	024720	121427	000060		1\$: CMPB (R4),#0
4182	↑B024724		002421		BLT 3\$
4183	024726	121427	000071		CMPB (R4),#9
4184	024732	003016			BGT 3\$
4185	024734	142714	000060		BICB #0,(R4)
4186	024740	005002			CLR R2
4187	024742	152402			BISB (R4)+,R2
4188	024744	060205			ADD R2,R5
4189	024746	122714	000015		CMPB #15,(R4)
4190	024752	001410			BEQ 4\$
4191	024754	006305			ASL R5 ;X2
4192	024756	010502			MOV R5,R2 ;SAVE X2
4193	024760	006305			ASL R5 ;X4
4194	024762	006305			ASL R5 ;X8
4195	024764	060205			ADD R2,R5 ;TIMES 10
4196	024766	000754			BR 1\$
4197	024770	104404			3\$: INSTER
4198	024772	000744			BR 2\$
4199					
4200		↑B			:TEST TO SEE IF NUMBER IS WITHIN LIMITS
4201					
4202	024774	020537	025042		4\$: CMP R5,7\$
4203	025000	101373			BHI 3\$
4204	025002	020537	025040		CMP R5,6\$
4205	025006	103770			BLO 3\$
4206	025010	133705	025046		BITB 9\$,R5
4207	025014	001365			BNE 3\$
4208					
4209					:STORE NUMBER AT SPECIFIED ADDRESS
4210					
4211	025016	013704	025044		5\$: MOV 8\$,R4
4212	025022	010524			MOV R5,(R4)+
4213	025024	062705	000002		ADD #2,R5
4214	025030	105337	025047		DECB 10\$

4215 025034 001372
4216 025036 000002
4217 025040 000000
4218 025042 000000
4219 025044 000000
4220 025046 000
4221 025047 000

BNE 55
RTI
65:
75:
85:
95:
105:

;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
;BUFFER TO THE CHARACTERS "E" AND "C"
;IF THE CHARACTER IS "E" CLEAR THE FLAG
;IF THE CHARACTER IS "C" SET THE FLAG

4229 025050 017605 000000
4230 025054 142737 000040 010502
4231 025062 122737 00180105
4232 025070 001002
4233 025072 105015
4234 025074 000406
4235 025076 122737 000103 010502
4236 025104 001005
4237 025106 112715 177777
4238 025112 062716 000002
4239 025116 000002
4240 025120 104404
4241 025122 000752

.PAWCH:MOV 2(SP),R5
BICB #40,INBUF ;SET FOR LOWER CASE INPUT
CMPB #'E,INBUF ;IS IT "E" ?
BNE 15
CLRB (R5) ;000
BR 25
15: CMPB #'C,INBUF ;IS IT "C" ?
BNE 35
MOVB #-1,(R5) ;3177
25: ADD #2,(SP)
35: RTI
INSTR ;RETRY
BR .PAWCH

;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
;LINE NUMBER (SAVLIN) FOR DZLPP, DZTCR AND DZCSR
;REGISTER USAGE.

4249 01825124 013737 001372 025342
4250 025132 013700 001372
4251 025138 005037 025344
4252 025142 012702 000001
4253 025146 005300
4254 025150 100402
4255 025152 006302
4256 025154 000774
4257 025156 012701 025220
4258 025162 010237 025344
4259 025166 022137 025336
4260 025172 001407
4261 025174 005721
4262 025176 001373
4263 025200 104402 025446
4264 025204 012705 023406
4265 025210 000402
4266 025212 011137 025340
4267 025216 000205

SET: MOV SAVLIN,NUMLIN ;SAVE SAVLIN
XTCR0: MOV SAVLIN,R0 ;COPY THE LINE NUMBER FOR LOOP CONTROL
CLR NUMTCR ;SET A DEFAULT OF LINE 0 OR NO LINES
MOV #1,R2 ;SET A BIT POINTER TO THE FIRST LINE
XTCR1: DEC R0 ;REDUCE THE INDICATOR. IS IT MINUS YET?
BMI SET1 ;IF SO, R2 POINTS TO THE RIGHT LINE
ASL R2 ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
BR XTCR1 ;GO SEE IF THIS LINE IS THE ONE
SET1: MOV #TABLE2,R1
MOV R2,NUMTCR ;COPY THE CORRECT BIT POINTER
15: CMP (R1)+,LINESP
BEQ 25
TST (R1)+ ;IS IT THE END OF TABLE?
BNE 15 ;NO
TYPE ,MINVAL ;INVALID BAUD RATE,BEGIN AGAIN
MOV #BAUD,R5 ;JUMP TO BAUD THRU R5
BR 35
25: MOV (R1),SPEED ;SET UP BAUD RATE
35: RTS R5

4268
4269
4270

```

4271                                     ;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR.)
4272 025220 000062 TABLE2: .WORD 50. ;50 BAUD
4273 025222 010070 .WORD 10070
4274 025224 000113 .WORD 75. ;75 BAUD
4275 025226 010470 .WORD 10470
4276 025230 000156 .WORD 110. ;110 BAUD
4277 025232 011070 .WORD 11070 ;TWO STOP BITS
4278 025234 000207 .WORD 135. ;134.5 BAUD
4279 025236 011470 .WORD 11470 ;TWO STOP BITS
4280 025240 000226 .WORD 150. ;150 BAUD
4281 025242 012070 .WORD 12070 ;TWO STOP BITS
1B4282 025244 000454 .WORD 300. ;300 BAUD
4283 025246 012470 .WORD 12470 ;ONE STOP BIT
4284 025250 001130 .WORD 600. ;600 BAUD
4285 025252 013070 .WORD 13070 ;ONE STOP BIT
4286 025254 002260 .WORD 1200. ;1200 BAUD
4287 025256 013470 .WORD 13470 ;ONE STOP BIT
4288 025260 003410 .WORD 1800. ;1800 BAUD
4289 025262 014070 .WORD 14070 ;ONE STOP BIT
4290 025264 003720 .WORD 2000. ;2000 BAUD
4291 025266 014470 .WORD 14470 ;ONE STOP BIT
4292 025270 004540 .WORD 2400. ;2400 BAUD
4293 025272 015070 .WORD 15070 ;ONE STOP BIT
4294 025274 007020 .WORD 3600. ;3600 BAUD
4295 025276 015470 .WORD 15470 ;ONE STOP BIT
4296 025300 011300 .WORD 4800. ;4800 BAUD
4297 025302 016070 .WORD 16070 ;ONE STOP BIT
4298 025304 016040 .WORD 7200. ;7200 BAUD
4299 025306 016470 .WORD 16470 ;ONE STOP BIT
4300 025310 022600 .WORD 9600. ;9600 BAUD
4301 025312 017070 .WORD 17070
4302 025314 177777 000000 .WORD -1,0 ;TABLE TERMINATOR
4303
4304
4305 025320 000000 WCHFLG: 0 ;ECHO OR CABLE FLAG
4306 025322 000000 STFLG: 0 ;PROGRAM START FLAG
4307 025324 000000 LOCKUP: 0 ;TIMEOUT FLAG
4308 025326 000000 LAST: 0 ;LAST ERROR PC
1B4309 025330 000000 TDATA: 0
4310 025332 000000 RDATA: 0
4311 025334 000000 BYTCNT: 0
4312 025336 000156 LINESP: 110. ;DEFAULT BAUD RATE
4313 025340 006307 SPEED: 6307 ;DEFAULT 110 BAUD, 8 BITS/CHAR,
4314 ;FDX, 2 STOP BITS
4315 025342 000100 NUMLIN: 100 ;DEFAULT VALUE, REC. INTERRUPT ENABLED
4316
4317 025344 000001 NUMTCR: 1 ;DEFAULT VALUE, TCR BIT 0
4318 025346 000240 PRIO: 240 ;DEFAULT DEVICE PRIORITY 5
4319 025350 000000 RECDAT: 0
4320 025352 000000 TBUF: 0
4321 025354 053200 041505 047524 MVECTO: .ASCIZ <200>/VECTOR ADDRESS- /
025376 041600 047117 051121B4 MREGAD: .ASCIZ <200>/CONTROL REGISTER ADDRESS- /
025432 050200 051501 020123 MPASS: .ASCIZ <200>/PASS DONE. /
025446 044600 053116 046101 MINVAL: .ASCIZ <200>/INVALID BAUD RATE - /
025474 046200 047111 035105 MLINE: .ASCIZ <200>/LINE: /
025504 041200 052501 020104 MSPEED: .ASCIZ <200>/BAUD RATE - /

```

```

025522 052200 050131 020105 MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZ11 TERMINAL /
025562 053600 044510 044103 MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /
025630 052200 051105 044515 MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /
025655 200 040503 046102 MCABLE: .ASCIZ <200>/CABLE TEST /
025672 006777 177777 177412 MQUICK: .ASCII <377><15><377><377><12><377><377>
025701 124 042510 050440 .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/
025776 006777 177777 177412 .ASCII <377><15><377><377><12><377><377><377><0>
02601C

```

4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367

```

.EVEN
:*****
;UTILITIES
:*****

```

```

;THIS UTILITY CALCULATES PRIORITY LEVEL, SETS UP CSR'S, SETS UP VECTORS.
DZLEV: ASL DZPRT ;BUILD PRIORITY IN THIS LOCATION
ASL DZPRT ;USING ARITHMETIC SHIFTS, ROTATE
ASL DZPRT ; THE PRIORITY LEVEL PAST
ASL DZPRT ; THE BIT POSITIONS CORRE-
ASL DZPRT ;SPONDING TO THE CONDITION CODES
MOV DZPRT, LESS1 ;MOVE THIS TO LESS1
SUB #1, LESS1 ;CREATE THE NEXT LOWEST PRIORITY
BIC #37, LESS1 ;INSURE THAT THE TNZVC BITS ARE CLEAR
MOV DZRIV, RO ;PLACE THE BASE VECTOR ADDRESS IN RO
ADD #2, RO ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
MOV RO, DZRIS ;STORE IT HERE
ADD #2, RO ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
MOV RO, DZTIV ;STORE IT HERE
ADD #2, RO ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
MOV RO, DZTIS ;STORE IT HERE

```

```

;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZ11. $BASE IS THE BASE ADDRESS
;OF THE DEVICE

```

```

MOV $BASE, RO ;COPY THE ADDRESS BEING LOADED
MOV RO, DZCSR ;XXX0
INC RO
MOV RO, HDZCSR ;XXX1
INC RO
MOV RO, DZRBUF ;XXX2
MOV RO, DZLPR ;XXX2
INC RO
MOV RO, HDZRBUF ;XXX3
MOV RO, HDZLPR ;XXX3
INC RO
MOV RO, DZTCR ;XXX4
INC RO
MOV RO, †BHDZTCR ;XXX5
INC RO
MOV RO, DZMSR ;XXX6
MOV RO, DZTDR ;XXX6
INC RO
MOV RO, HDZMSR ;XXX7
MOV RO, HDZTDR ;XXX7
RTS
PC

```

```

DZPRT: PR5
LESS!: PR4 ;LEVEL TO ALLOW INTERRUPTS

```

			:ERROR ERROR TABLE	
			.ERRTAB:	
4368				
4369	026222	000000	01B	:ERROR 0
4370	026224	000000	0	
4371	026226	000000	0	
4372				
4373	026230	026434	EM1	:ERROR
4374	026232	027634	DH1	
4375	026234	030032	DT1	
4376				
4377	026236	026507	EM2	:ERROR 2
4378	026240	027657	DH2	
4379	026242	030044	DT2	
4380				
4381	026244	026535	EM3	:ERROR 3
4382	026246	027712	DH3	
4383	026250	030062	DT3	
4384				
4385	026252	026574	EM4	:ERROR 4
4386	026254	027712	DH3	
4387	026256	030062	DT3	
4388				
4389	026260	026623	EM5	:ERROR 5
4390	026262	027724	DH4	
4391	026264	030070	DT4	
4392				
4393	026266	026652	EM6	:ERROR 6
4394	026270	027724	DH4	
4395	026272	030070	DT4	
4396				
4397	026274	026710	EM7	:ERROR 7
4398	026276	027712	DH3	
4399	026300	030062	DT3	
4400				
4401	026302	026751	EM8	:ERROR 10
4402	026304	027712	DH3	
4403	026306	030062	DT3	
4404				
4405	026310	027013	EM9	:ERROR 11
4406	026312	027712	DH3	
4407	026314	030062	DT3	
4408				
4409	026316	027051	EM10	:ERROR 12
4410	026320	027712	DH3	
4411	026322	030062	DT3	
4412				
4413	026324	027110	EM13	:ERROR 13
4414	026326	027712	DH3	
4415	026330	030062	DT3	
4416				
4417	026332	027141	EM14	:ERROR 14
4418	026334	027712	DH3	
4419	026336	030062	DT3	
4420				
4421	026340	027173	EM15	:ERROR 15
4422	026342	000000	0	
4423	026344	000000	0	

1B

4424				
4425	026346	027235	EM16	
4426	026350	027712	DH3	
4427	026352	030062	DT3	
4428				
4429	026354	027306	EM17	;ERROR 17
4430	026356	027712	DH3	
4431	026360	030062	DT3	
4432				
4433	026362	027344	EM20	
4434	026364	027712	DH3	
4435	026366	030062	DT3	
4436				
4437	026370	027405	EM21	;ERROR 21
4438	026372	027753	DH5	
4439	026374	030106	DT5	
4440				
4441	026376	027435	EM22	;ERROR 22
4442	026400	027724	DH4	
4443	026402	030070	DT4	
4444				
4445	026404	027477	EM23	;ERROR 23
4446	026406	027712	DH3	
4447	026410	030062	DT3	
4448				
4449	026412	027527	EM24	
4450	026414	027712	DH3	
4451	026416	030062	DT3	
4452				
4453	026420	027555	EM25	
4454	026422	027712	DH3	
4455	026424	030062	DT3	
4456				
4457	026426	027605	EM26	
4458	026430	027712	DH3	
4459	026432	030062	DT3	

↑B

```

4460
4461 026434 047200 020117 046123 EM1: .ASCIZ <200>/NO SLAVE SYNC RESPONSE FROM DZ11 REGISTER/
      026507      200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE/
      026535      200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TROY) NOT SET/
      026574 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
      026623      200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
      026652 042200 030532 020061 EM6: .ASCIZ <200>/DZ11 *RECEIVER BUFFER* ERROR/
      026710 052200 040522 051516 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
      026751      200 047125 054105 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
      027013      200 042522 042503 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
      027051      200 047125 054105 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
      027110 051600 046111 020117 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
      027141      200 044523 047514 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
      027173      200 041501 044524 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
      027235      200 042522 042101 EM16: .ASCIZ <200>/READING DZRBUF DID NOT CLEAR SILO †BALARM/
      027306 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
      027344 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
      027405      200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
      027435      200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
      027477      200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
      027527      200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
      027555      200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
      027605      200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/

      027634 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZ11 REG/
      027657      200 054105 042520 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
      027712 046200 047111 020105 DH3: .ASCIZ <200>/LINE NO./
      027724 042600 050130 041505 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
      027753      200 054124 046040 DHS: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/
    
```

.EVEN

:DATA TABLES FOR ERROR MESSAGES

```

030032 000002 DT1: 2
030034      006      003      .BYTE      6,3
030036 001204 $REG1
030040      006      001      .BYTE      6,1
030042 001202 $REG0

030044 000003 DT2: 3
030046      006      004      .BYTE      6,4
030050 001214 $REG5
030052      00†B6      001      .BYTE      6,1
030054 001212 $REG4
030056      006      001      .BYTE      6,1
030060 001202 $REG0

030062 000001 DT3: 1
030064      003      001      .BYTE      3,1
030066 001372 SAVLIN

030070 000003 DT4: 3
030072      006      004      .BYTE      6,4
030074 001214 $REG5
030076      006      001      .BYTE      6,1
030100 001212 $REG4
030102      003      001      .BYTE      3,1
    
```

030104	001372		SAVLIN
030106	000004		DTS: 4
030110	003	005	.BYTE 3,5
030112	001372		↑B SAVLIN
030114	006	011	.BYTE 6,9.
030116	001214		\$REGS
030120	006	007	.BYTE 6,7
030122	001220		\$TMP1
030124	006	001	.BYTE 6,1
030126	001400		REGIST

:TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

030130	002450	DLYTBL: 2450	:TIME FOR 50 BAUD
030132	001560	1560	:TIME FOR 75 BAUD
030134	001120	1120	:TIME FOR 110 BAUD
030136	000750	750	:TIME FOR 134 BAUD
030140	000660	660	:TIME FOR 150 BAUD
030142	000330	330	:TIME FOR 300 BAUD
030144	000150	150	:TIME FOR 600 BAUD
030146	000060	60	:TIME FOR 1200 BAUD
030150	000040	40	:TIME FOR 1800 BAUD
030152	000030	30	:TIME FOR 2000 BAUD
030154	000020	20	:TIME FOR 2400 BAUD
030156	000010	10	:TIME FOR 3600 BAUD
030160	000001	1	:TIME FOR 4800 BAUD
030162	000001	1	:TIME FOR 7200 BAUD
030164	000001	1	:TIME FOR 9600 BAUD
030166	000001	1	:TIME OF DELAY FOR 19200 BAUD

:DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
:FOR ALL TESTS TO FUNCTION CORRECTLY ON A PDP11/45 WITH BIPOLAR
:MEMORY. THE TIMES WERE ALSO TESTED ON AN 11/40 AND 11/10.

030170 000001 C↑BORMAX:
.END

ABASE = 160010	AUTO.S = 011462	C05 = 020000	DZCR4 = 001560	EIGHTS= 000070
ACDW1 = 000000	AVECT = 030300	C06 = 040000	DZCR5 = 001574	EMTVEC= 000030
ACDW2 = 000000	AVECT1= 000000	C07 = 100000	DZCR6 = 001610	EM1 = 026434
ACPUOP= 000000	AVECT2= 000000	CR = 000015	DZCR7 = 001624	EM10 = 027051 †B
ACTIVE = 001412	BAUD = 023406	CRLF = 000200	DZCSR = 002042	EM13 = 027110
ADDW0 = 000000	BINWRD = 006510	CSRMAP = 011470	DZLEV = 026010	EM14 = 027141
ADDW1 = 000000	BIT0 = 000001	CYCLE = 010752	DZLPR = 002052	EM15 = 027173
ADDW10= 000000	BIT00 = 000001	DATABP = 007102	DZLVO = 001504	EM16 = 027235
ADDW11= 000000	BIT01 = 000002	DATAHD = 007070	DZLV1 = 001520	EM17 = 027306
ADDW12= 000000	BIT02 = 000004	DCLASM= 104417	DZLV10 = 001644	EM2 = 026507
ADDW13= 000000	BIT03 = 000010	DCLR = 000020	DZLV11 = 001660	EM20 = 027344
ADDW14= 000000	BIT04 = 000020	DDISP = 177570	DZLV12 = 001674	EM21 = 027405
ADDW15= 000000	BIT05 = 000040	DELAY = 104414	DZLV13 = 001710	EM22 = 027435
ADDW2 = 000000	BIT06 = 000100	DEVADR = 006232	DZLV14 = 001724	EM23 = 027477 †B
ADDW3 = 000000	BIT07 = 000200	DEVICE= 104413	DZLV15 = 001740	EM24 = 027527
ADDW4 = 000000	BIT08 = 000400	DH1 = 027634	DZLV16 = 001754	EM25 = 027555
ADDW5 = 000000	BIT09 = 001000	DH2 = 027657	DZLV17 = 001770	EM26 = 027605
ADDW6 = 000000	BIT1 = 000002	DH3 = 027712	DZLV2 = 001534	EM3 = 026535
ADDW7 = 000000	BIT10 = 002000	DH4 = 027724	DZLV3 = 0018550	EM4 = 026574
ADDW8 = 000000	BIT11 = 004000	DH5 = 027753	DZLV4 = 001564	EM5 = 026623
ADDW9 = 000000	BIT12 = 010000	DISPLA = 001162	DZLV5 = 001600	EM6 = 026652
ADEVCT= 000000	BIT13 = 020000	DISPRE = 000174	DZLV6 = 001614	EM7 = 026710
ADEVN = 000000	BIT14 = 040000	DLYCNT = 006604	DZLV7 = 001630	EM8 = 026751
ADRcnt = 006235	BIT15 = 100000	DLYTBL = 030130	DZMSR = 002062	EM9 = 027013
ADVANC= 104400	BIT2 = 000004	DONFLG = 001420	DZNUM = 001410	ERRMSG = 007056
RENV = 000000	BIT3 = 000010	DSWR = 177570	DZPRT = 026216	ERRVEC= 000004
RENVN = 000000	BIT4 = 000020	DTR0 = 000400	DZRBUF = 002046	ERTAB0 = 007226
AFATAL = 000000	BIT5 = 000040	DTR1 = 001000	DZRIS = 002074	EVEPAR= 000000
AMADR1 = 000000	BIT6 = 000100	DTR2 = 002000 †B	DZRIV = 002072	EXITER = 007162
AMADR2 = 000000	BIT7 = 000200	DTR3 = 004000	DZTCR = 002056	FINI = 020770
AMADR3 = 000000	BIT8 = 000400	DTR4 = 010000	DZTDR = 002066	FIVE = 000000
AMADR4 = 000000	BIT9 = 001000	DTR5 = 020000	DZTIS = 002100	FIVES = 000040
AMAMS1 = 000000	BPTVEC= 000014	DTR6 = 040000	DZTIV = 002076	FRMERR= 020000
AMAMS2 = 000000	BRK0 = 000400	DTR7 = †B 100000	DZVCO = 001502	HALTS = 007106
AMAMS3 = 000000	BRK1 = 001000	DT1 = 030032	DZVC1 = 001516	HDRFLG = 001416
AMAMS4 = 000000	BRK2 = 002000	DT2 = 030044	DZVC10 = 001642	HDZCSR = 002044
AMSGAD = 000000	BRK3 = 004000	DT3 = 030062	DZVC11 = 001656	HDZLPR = 002054
AMSGLG = 000000	BRK4 = 010000	DT4 = 030070	DZVC12 = 001672	HDZMSR = 002064
AMSGTY = 000000	BRK5 = 020000	DT5 = 030106	DZVC13 = 001706	HDZRBU = 002050
AMTYP1 = 000000	BRK6 = 040000	DVALID= 100000	DZVC14 = 001722	HDZTCR = 002060
AMTYP2 = 000000	BRK7 = 100000	DZACTV = 001404	DZVC15 = 001736	HDZTDR = 002070
AMTYP3 = 000000	BRW = 005114	DZCRO = 001500	DZVC16 = 001752	HILIM = 006230
AMTYP4 = 000000	BUILD = 022676	DZCR1 = 001514	DZVC17 = 001766	HT = 000011
APASS = 000000	BYTCNT = 02533†B4	DZCR10 = 001640	DZVC2 = 001532	INBUF = 010502
APRIOR = 000000	CHRCNT = 006506	DZCR11 = 001654	DZVC3 = 001546	INIFLG = 001415
APTCSU = 000040	CNVRT = 104412	DZCR12 = 001670	DZVC4 = 001562	INIT = 020250
APTEMV = 000001	CONVRT= 104411	DZCR13 = 001704	DZVC5 = 001576	INIT1 = 020274
APTSIZ = 000200	CORMAX = 030170	DZCR14 = 001720	DZVC6 = 001612	INSTER= 104404
APTSP0 = 000100	C00†B = 000400	DZCR15 = 001734	DZVC7 = 001626	INSTR = 104403
ASWREG = 000000	C01 = 001000	DZCR16 = 001750	DZ.END = 002000	INSTR2 = 006030
ATESTN = 000000	C02 = 002000	DZCR17 = 001764	DZ.MAP = 001500	INTRAN = 024614
AUNIT = 000000	C03 = 004000	DZCR2 = 001530	EIAFLG = 001414	INTREC = 024434
AUSWR = 000000	C04 = 010000	DZCR3 = 001544	EIGHT = 000030	INTSVC = 024014

IOTVEC= 000020	MASTEK 010177	PAR14 001730	RL3 = 001400	SW10 = 002000
LAST 025326	MBADLN 010306	PAR15 001744	RL4 = 002000	SW11 = 004000
LESS1 026220	MCABLE 025655	PAR16 001760	RL5 = 002400	SW12 = 010000
LF = 000012	MCHAR 025522	PAR17 001774	RL6 = 003000	SW13 = 020000
LIMITS 006156	MCSRX †B 010127	PAR2 001540	RL7 = 003400	SW14 = 040000
LINE 001364	MDATA 010606	PAR3 001554	RSTART 020240	SW15 = 100000
LINESP 025336	MEPASS 007745	PAR4 001570	RSTRT 024650	SW2 = 000004
LINEX 023424	MERRPC 010254	PAR5 001604	RUN 001406	SW3 = 000010
LINE0 001506	MERRX 010154	PAR6 001620	RXISR1 022724	SW4 = 000020
LINE1 001522	†B MERR2 010005	PAR7 001634	RXSVC 020542	SW5 = 000040
LINE10 001646	MERR3 010054	PAWCH = 104416	RXTCR 021016	SW6 = 000100
LINE11 001662	MINVAL 025446	PIRQ = 177772	R6 = %000006	SW7 = 000200
LINE12 001676	MLINE 025474	PIRQVE= 000240	R7 = %000007	SW8 = 000400
LINE13 001712	MLOCK 010100	POPPO = 012600	SAVLIN 001372	SW9 = 001000
LINE14 001726	MNEW 010202	POP1SP= 005726	SAVNUM 001411	S110 = 001000
LINE15 001742	MNTFLG 001417	POP2SP= 022626	SAVPC 001402	S1200 = 003400
LINE16 001756	MODE 001370	PRI0 025346	SAVOS = 104407	S134 = 001400
LINE17 001772	MPASS 025432	PRO = 000000	SCOPI = 104401	S150 = 002000
LINE2 001536	MPASSX 010143	PR1 = 000040	SERV.G 007242	S1800 = 004000
LIN1BE3 001552	MPFAIL 007702	PR2 = 000100	SET 025124	S19200= 007400
LINE4 001566	MQUICK 025672	PR3 = 000140	SETAPT 011310	S2000 = 004400
LINE5 001602	MR 007771	PR4 = 000200	SETFLG= 104406	S2400 = 005000
LINE6 001616	MREGAD 025376	PR5 = 000240	SET.PS 010650	S300 = 002400
LINE7 001632	MSENAB= 000040	PR6 = 000300	SET1 025156	S3600 = 005400
LOBITS 006234	MSPEED 025504	PR7 = 000340	SEVEN = 000020	S4800 = 006000
LOCK 001362	MTERM 025630	PS = 177776	SEVENS= 000060	S50 = 000000
LOCKUP 025324	MTITLE 001000	PSW = 177776	SILUAL= 020000	S600 = 003000
LOLIM 006226	MTSTN 010165	PUSHRO= 010046	SILUEN= 010000	S7200 = 006400
LPO = 000000	MVECTO 025354	PUSH1S= 005746	SIX = 000010	S75 = 000400
LP1 = 000001	MVECX 010135	PUSH2S= 024646	SIXS = 000050	S9600 = 007000
LP2 = 000002	MWHICH 025562	PWRVEC= 000024	SNAP 020370	TABLE2 025220
LP3 = 000003	NEXT 001360	QUITS 024426	SPACNT 006507	TBITVE= 000014
LP4 = 000004	NUMLIN 025342	RCVON = 010000	SPEED 025340	TBUF 025352
LP5 = 000005	NUMTCR 025344	RDATA 025332	SPIN 024374	†B TCRO = 000001
LP6 = 000006	ODDPAR= 000200	RDONE = 000200	STACK = 001120	TCR1 = 000002
LP7 = 000007	OFFSET 023140	RECDAT 025350	STFLG 025322	TCR2 = 000004
MAINT = 000010	ONESTO= 000000	REGIST 001400	STKLMT= 177774	TCR3 = 000010
MANT0 001512	OTHER 023366	RESREG 007104	STOP 001462	TCR4 = 000020
MANT1 001526	OUT 020716	RESTAR 011304	SVOS 006244 †B	TCR5 = 000040
MANT10 001652	OVRUN= 040000	RESVEC= 000010	SWR 001160	TCR6 = 000100
MANT11 001666	PAR 001366	RESOS = 104410	SWREG 000176	TCR7 = 000200
MANT12 001702	PARAM = 104405	RIE = 000100	SWO = 000001	TDATA 025330
MANT13 001716	PARAM1 006076	RING0 = 000001	SW00 = 000001	TDO 001422
MANT14 001732	PARER = 010000	RING1 = 000002	SW01 = 000002	TD1 001424
MANT15 001746	PARERR 006152	RING2 = 000004	SW02 = 000004	TD2 001426
MANT16 001762	PARITY= 000100	RING3 = 000010	SW03 = 000010	TD3 001430
MANT17 001776	PARMD = 104415	RING4 = 000020	SW04 = 000020	TD4 001432
MANT2 001542	PARO 001510	RING5 = 000040	SW05 = 000040	TD5 001434
MANT3 001556	PAR1 001524	RING6 = 000100	†B SW06 = 000100	TD6 001436
MANT4 001572	PAR10 001650	RING7 = 000200	SW07 = 000200	TD7 001440
MANT5 001606	PAR11 001664	RLO = 000000	SW08 = 000400	TEIGHT 002140
MANT6 001622	PAR12 001700	RL1 = 000400	SW09 = 001000	TEMP 010544
MANT7 001636	PAR13 001714	RL2 = 001000	SW1 = 000002	TEST1 023554

TEST2	024176	TST5	012656	\$ATY4	005462	\$FILLS	001175	\$SCOPE	004654
TFIVE	002146	TST6	012750	\$AUTO8	001154	\$GDADR	001140	\$SETUP=	000000
TIE	= 040000	TST7	013042	\$BASE	001310	\$GDDAT	001144	\$SVLAD	005042
TKVEC	= 000060	TTST	004672	\$BDADR	001142	\$GET42	004600	\$SVPC =	000040
TLAST	= 022146	TWOSTO=	000040	\$BDDAT	001146	\$HD =	000001	\$SWR =	164000
TLO	= 000000	TXSVC	020442	\$CDW1	001314	\$HIBTS	001462	\$SWREG	001256
TL1	= 000400	TYPDAT	007072	\$CDW2	001316	\$ICNT	001124	\$SWRMK=	000000
TL2	= 001000	TYPE =	104402	\$CHARC	005440	\$ILLUP	007674	\$TESTN	001240
TL3	= 001400	TYPMSG	006762	\$CMTAG	001120	\$INTAG	001155	\$TIMES	001226
TL4	= 002000	T110	002106	\$CM1 =	000006 †B	\$ITEMB	001134	\$TKB	001166
TL5	= 002400	T1200	002120	\$CM2 =	000014	\$LF	001232	\$TKS	001164
TL6	= 003000	T134	002110	\$CM3 =	000006	\$LFLG	005707	\$TMP0	001216
TL7	= 003400	T150	002112	\$CM4 =	000004	\$LPADR	001126	\$TMP1	001220
TMTBL	002102	T1800	002122	\$CPUOP	001262	\$LPERR	001130	\$TMP2	001222
TPVEC	= 0000F4	T2000	002124	\$CRAP†B	= 177777	\$MADR1	001266	\$TMP3	001224
TRAPVE=	000034	T2400	002126	\$CRLF	001231	\$MADR2	001272	\$TN =	000035
TRDY	= 100000	T300	002114	\$DDW0	001320	\$MADR3	001276	\$TPB	001172
TRTVEC=	000014	T3600	002130	\$DDW1	001322	\$MADR4	001302	\$TPFLG	001177
TRO	001442	T4800	002132	\$DDW10	001344	\$MAIL	001234	\$TPS	001170
TR1	001444	T50	002102	\$DDW11	001346	\$MAMS1	001264	\$STM	001466
TR2	001446	T600	002116	\$DDW12	001350	\$MAMS2	001270	\$STNM	001122
TR3	001450	T7200	002134	\$DDW13	001352	\$MAMS3	001274	\$TYPE	005162
TR4	001452	T75	002104	\$DDW14	001354	\$MAMS4	001300	\$TYPEC	005374
TR5	001454	T9600	002136	\$DDW15	001356	\$MBADR	001464	\$TYPEX	005442
TR6	001456	VECMAP	01218026	\$DDW2	001324	\$MFLG	005706	\$UNIT	001246
TR7	001460	VEC1	023204	\$DDW3	001326	\$MSGAD	001250	\$UNITM	001472
TSEVEN	002142	VEC2	023214	\$DDW4	001330	\$MSGLG	001252	\$USWR	001260
TSIX	002144	WCHFLG	025320	\$DDW5	001332	\$MSGTY	001234	\$VECT1	001304
TST1	012216	WRDCNT	006504	\$DDW6	001334	\$MTYP1	001265	\$VECT2	001306
TST10	013134	W†BTBS.F	007060	\$DDW7	001336	\$MTYP2	001271	\$XTSTR	004720
TST11	013272	XBEGIN	023454	\$DDW8	001340	\$MTYP3	001275	\$Y =	000020
TST12	013446	XBX	006650	\$DDW9	001342	\$MTYP4	001301	\$SET4=	000000
TST13	013564	XCSR	004624	\$DEVCT	001244	\$MXCNT	005116	=	030170
TST14	013650	XEOP	024634	\$DEVM	001312	\$N =	000034	.ADVAN	006606
TST15	013740	XERR	004646	\$DOAGN	004620	\$NULL	001174	.BEGIN	004356
TST16	014024	XHEAD	010261	\$E =	000036	\$NWTST=	000000	.CNVRT	006334
TST17	014214	XMTCNT	001376	\$ENDAD	004610	\$OVER	005060	.CONVR	006330
TST2	012406	XMTLIN	001374	\$ENDCT	004574	\$PASS	001242	.DCLAS	006554
TST20	014352	XMTSRV	022600	\$ENV	001254	\$PASTM	001470	.DELAY	006566
TST21	†B 014466	XPASS	004640	\$ENVM	001255	\$PWRAD	00767	.DEVIC	006534
TST22	015002	XSTART	023142	\$EOP	004444	\$PWDRN	007530	.ERRTA	026222
TST23	015330	XSTATQ	010350	\$EOPCT	004566	\$PWARMG	007664	.INSTE	006016
TST24	015606	XTCRO	025132	\$ERFLG	001123	\$PWRUP	007602	.INSTR	005712
TST25	016114	XTCRI	025146	\$ERMAX	001135	\$QUES	001230	.INST1	005732
TST26	016450	XTSTN	007234	\$ERROR	006620	\$REGAD	001200	.MSG	005734
TST27	017102	XVEC	004632	\$ERRPC	001136	\$REGO	001202	.PARAM	006036
TST3	012472	XX =	160210	\$ERRTB	001360	\$REG1	001204	.PARAMD	024654
TST30	017630	YY =	000500	\$ERTTL	001132	\$REG2	001206	.PAWCH	025050
TST31	020212	ZZ =	000020	\$ETABL	001254	\$REG3	001210	.RES05	006276
TST32	021020	\$APTHD	001462	\$ETEND	001360	\$REG4	001212	.SAV05	006236
TST33	021514	\$ATYC	005470	\$FATAL	001236	\$REG5	001214	.SCOPE	004654
TST34	022146	\$ATY1	005444	\$FFLG	005710	\$RTNAD	004622	.SCOPI	005120
TST4	012564	\$ATY3	005452	\$FILLC	001176	\$SAVR6	007700	.SETFL	010362

B10

MO-11-DZDZA-C MACY: 27:1006: 21-OCT-76 13:09 PAGE 96
DZDZAC.P11 21-OCT-76 13:07 SYMBOL TABLE

PAGE: 0118

.IBSTART 00215C .TRPSR 006512 .TRPTA 002002 .TYPE 005144 .SX = 001462

.ABS. 030170 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DZDZAC, DZDZAC/SOL+SYSMAC.SML(400,1066), DZDZAC.P11(400,2670)
RUN-TIME: 57.74 2 SECONDS
RUN-TIME RATIO: 292/135=2.1
CORE USED: 50K (100 PAGES)

