

DZ11

8 LINE ASYNCHRONOUS
MD-11-DZDZA-E
MULTIPLEXER TESTS

EP-DZDZA-E-DL
COPYRIGHT © 76-78
FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

This microfiche card contains a grid of 100 small frames of test data, arranged in 10 rows and 10 columns. Each frame displays a different set of test results, likely related to the MD-11-DZDZA-E multiplexer tests mentioned in the header. The data is presented in a structured, tabular format, with various columns and rows of text and numbers. The frames are separated by thin white lines, and the overall layout is consistent across the entire card.

B01

EOF1CZTAB2SBQ411

00010000

780223

IDENTIFICATION

Q HDR1DZDZAESEQ

00010000

780223
SEQ 0001

PRODUCT CODE: MAINDEC-11-DZDZA-E-D
PRODUCT NAME: DZ11 8 LINE ASYNC MUX TESTS
DATE RELEASED: FEB 1978
MAINTAINER: DIAGNOSTIC ENGINEERING

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 OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976, 1978 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. The AUTOSIZER is designed to detect DZ11 device addresses and vectors and to determine whether the DZ11 that is detected is an EIA or 20mA board. All remaining parameters default to certain values (see SEC.8.5). Console input may be controlled at any start time through the use of SW00, SW03, SW04, and SW06 (see SEC. 4.1.1 for a detailed description of these switches).

Currently there is one standalone diagnostic (DZDZA), one system module for DEC X/11 (DZAA), and an online overlay for DZITA (ITEP) - DZDZB. (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)
 ASA 33 (or equivalent for console)
 DZ11 INTERFACE MODULE (M7819(EIA), M7814(20MA))
 H3271 Staggered turnaround connector for EIA module.
 H3190 Staggered turnaround connector for 20mA module.
 H325 Cable turnaround and dist pnl testing for EIA module.
 H315 This may be substituted for H325.

NOTE: A staggered turnaround connector is needed in order to test the PARITY and BREAK logic.

2.2 STORAGE

Program will use all 8K of memory except where ABL and BOOTSTRAP LOADER reside. Location 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 PROCEDURE

- A. Set switch register to 000200
- B. Depress 'LOAD ADDRESS' key and release
- C. Set SWR to zero for 'AUTO SIZING' or set SW00=1 for user parameter input from console terminal. On first start if SW07=1 and SW00=0 the program will default to console parameter input (SW00=1).
- 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 160100
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 8.4 for help.

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

4.1 CONTROL SWITCH SETTINGS

NOTE: 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. If 1st start of program after loading the
operator must input address and vector from console.
SW 06 Set: Reselect DZ11's desired active
SW 05 Set: Reserved
SW 04 Set: Select delay parameter (see SEC. 4.1.1)
SW 03 Set: Extra parameter input (see SEC. 4.1.1)
SW 02 Set: Lock on selected test
**SW 01 Set: Restart program at selected test
*SW 00 Set: Get users parameters from console
```

* For Echo or Cable tests (program started at loc. 210) this switch set to 1 allows the user to type in the Vector and the CSR for the DZ11 under test.

** For Echo or Cable test this switch set to 1 allows the selection of either the Echo or Cable test, baud rate, and the line number under test.

4.1.1 SWITCH REGISTER CONTROL OF PARAMETER INPUT FROM CONSOLE

- SW 00 GET USERS PARAMETERS FROM CONSOLE. Setting this switch at start up time allows the user to input at the Console terminal the following parameters: base device address, base vector address, bus request level, declare EIA or 20mA module, mode of operation (External, Internal, or Staggered), and the number of DZ11's that are running. Using this switch alone defaults the following parameters: all 8 lines are set to be tested on each DZ11, the default baud rate is set at 19.2 kbaud, and the character length for the majority of testing is set at eight bits per character with two stop bits.
- SW 03 EXTRA PARAMETER INPUT. Setting this switch at start up time provides the user with the ability to set the lines active for testing and to set the default baud rate used for the majority of the diagnostic tests. The Delay Parameter is automatically adjusted to the baud rate given by the user.
- SW 04 SELECT DELAY PARAMETER. The DELAY parameter this switch controls determines the length of time the program stalls waiting for a character to be completely transmitted or received. This delay count is automatically set to provide enough delay time for the default baud rate specified when running the program on an 11/45 with bipolar memory. When running this program on a faster processor the delay parameter should be adjusted proportionally higher than the following defaulted values:
- | | | | |
|------|------------|------|-------|
| 2450 | ; time for | 50 | baud |
| 1560 | ; time for | 75 | baud |
| 1120 | ; time for | 110 | baud |
| 0750 | ; time for | 134 | baud |
| 0660 | ; time for | 150 | baud |
| 0330 | ; time for | 300 | baud |
| 0150 | ; time for | 600 | baud |
| 0060 | ; time for | 1200 | baud |
| 0040 | ; time for | 1800 | baud |
| 0030 | ; time for | 2000 | baud |
| 0020 | ; time for | 2400 | baud |
| 0010 | ; time for | 3600 | baud |
| 0001 | ; time for | 4800 | baud |
| 0001 | ; time for | 7200 | baud |
| 0001 | ; time for | 9600 | baud |
| 0001 | ; time for | 19.2 | kbaud |

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. This switch is designed to provide an aid for a trained troubleshooter to sample various signals on the module and is not meant to be used as a general user control switch.
- 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 USED 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 177776. (see SEC. 4.1.1)

4.1.3 SWITCH REGISTER PRIORITIES

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'). If an '*' is printed in front of the test no. on an error report (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 the program user is technically trained to electronically isolate signal problems on the DZ11 module. If SW09 is not enabled; and there is a *HARD* error (constant); SW08 is best.
2. For intermittent errors either start the program with SW01 and SW02 set which will allow the user to lock on a selected test, or else set SW14 as an error is being typed out on the terminal. SW14 will continue to loop on that test regardless of whether an error occurs.
3. SW 14 Loop on current test.

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 terminal supported by the DZ11.

NOTE: If address 000042 is non-zero the program assumes it is under 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.

"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.

"MAINTENANCE MODE

[EXTERNAL <H325>-EIA ONLY (E)]
[INTERNAL <DZCSR03=1> (I)]
[STAGGERED <H3271>-EIA ONLY (S)]
[STAGGERED <H3190>-20mA ONLY (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 an R325 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. Baud rate choices are:
"00"(50 baud), "01"(75 baud), "02"(110 baud), "03"(134 baud),
"04"(150 baud), "05"(300 baud), "06"(600 baud), "07"(1200 baud),
"10"(1800 baud), "11"(2000 baud), "12"(2400 baud), "13"(3600 baud),
"14"(4800 baud), "15"(7200 baud), "16"(9600 baud), "17"(19.2 kbaud)
Low default baud rates are not suggested since they lengthen the
time to complete a program pass dramatically.

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.

Normal starting for the first time would be: LOAD ADDRESS 210; START WITH THE SWR EQUAL TO 003.

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 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 in; just load address 210 and start with SWR equal to 002. 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. Baud rate choices are: 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600. The program will then 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 THE LINE NO. IS REQUESTED ABOVE. 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 <f> ON THE DZ11 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: type any printable character on the CONSOLE TERMINAL (not the DZ11 terminal). The program will again type "LINE: " and wait for a response.

5.4 PROGRAM AND/OR OPERATOR ACTION

The variety of program Control Switches provided in this Diagnostic Package is designed to provide the user with a wide range of troubleshooting techniques. Before the user attempts to run this diagnostic he should become familiar with the use of these Control Switches and their restrictions. (See Sec. 4.1, 4.1.1, 4.1.2, 4.1.3)

When the program detects an error the TEST NUMBER and PC will be typed out and possibly an error message (depending on the particular error). If it is necessary to know more information concerning the error report then look in the program listing for that TEST NUMBER and then note the PC of the error report. The reason for the error report will become clearer when reading the comments in the program listing.

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 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 4.1.2
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=1 (INHIBIT ITERATIONS) is set to give the fastest possible execution. The actual execution time depends greatly on the PDP11 CPU configuration. An 11/40 with Core memory will take around 100 seconds to execute a pass with no iterations and about 400 seconds to execute a fully iterated pass. Any other PDP11 CPU type will execute a pass in time proportional to the execution speed of the CPU's memory in relation to that of an 11/40.

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-D 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.

B.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.

SBASE (1310) Contains the receiver CSR of the current DZ11 under test.

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

'MAP OF DZ11 STATUS'	
1500	160100
1502	000300
1504	000005
1506	000377
1510	017470
1512	000000

The above information will be repeated for each of up to 16 DZ11's in the system (these will follow under this table). EXPLANATION:

1500	160100	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 an M7819, if bit15=1 module should be an M7814.
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 indicates 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. This location is used to load the DZ11 Line Parameter Register for each line. The meaning of the bits set in this location is the same as the function of the related bits in the device Line Parameter Register.
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.

B.5 *** METHOD OF AUTO SIZING ***

B.5.1 FINDING THE CONTROL STATUS REGISTER.

The program will start at address 160000 and start 'REFERENCING' 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.

After identifying a DZ11 the program then attempts to set all DTR bits in Device Register 4. If any DTR bits did set the module is assumed to be an EIA module (M7B19) otherwise the status map entry is set for 20mA (M7B14).

B.5.2 FINDING THE VECTOR

The vector area (address 300-776) is filled with the instruction IOT and '+2' (next address). Bit14 and Bit5 (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.

B.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 if 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".

For all parameter 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
- \$DDW0 - 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.

DOCUMENT

DZDZAE LST

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

4 COPYRIGHT (C) 1977
 DIGITAL EQUIPMENT CORP.
 MAYNARD, MASS. 01754

 THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.

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

29 MISCELLANEOUS DEFINITIONS

41 GENERAL PURPOSE REGISTER DEFINITIONS

53 PRIORITY LEVEL DEFINITIONS

63 "SWITCH REGISTER" SWITCH DEFINITIONS

91 DATA BIT DEFINITIONS (BIT00 TO BIT15)

119 BASIC "CPU" TRAP VECTOR ADDRESSES

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

425 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

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

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

476 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:

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

1090 INCREMENT THE PASS NUMBER (\$PASS)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO CYCLE

1151 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

1227 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:
1) USING A TRAP INSTRUCTION
TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
OR
TYPE
MESADR

1932 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)

2054 *****TEST 1 *****
THIS TEST PROVES THE SLAVE SYNC RESPONSE
DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
DZCSR, DZRBUF, DZTCR, DZMSR

2097 *****TEST 2 *****
THIS TEST PROVES THAT BIT "DCLR"
CAN BE SET AND THAT IT WILL CLEAR
BY ITSELF AFTER A PERIOD OF TIME.

2127 *****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"

2159 ***** 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"

2191 ***** 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"

2223 ***** 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"

2255 ***** 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"

2287 ***** 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

2329 ***** TEST 11 *****
THIS TESTS THAT ALL OF THE FOLLOWING
BITS CAN BE: SET, CLEARED, CLEARED BY "RESET INSTR *NOT* DEVICE CLEAR "
BITS TESTED ARE:

2333 DTR0, DTR1, DTR2, DTR3, DTR4, DTR5, DTR6, DTR7
THIS TEST IS NOT DONE IF MODULE IS 20MA VERSION

2382 ***** 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.

- 2419 ***** 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 VERIFY THAT A RESET WILL CLEAR THESE BITS
THIS TEST ALSO CHECKS BYTE OPERATIONS ON THE CSR
- 2463 ***** TEST 14 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZRBUF
AND TESTING OF WRITE ONLY REGISTER DZLPR
- 2489 ***** TEST 15 *****
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZMSR
AND TESTING OF WRITE ONLY REGISTER DZTDR
- 2516 ***** TEST 16 *****
VERIFY THAT IF WE ARE IN "STAGGERED" MODE
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 RING AND CARRIER
ETC...
- 2575 ***** TEST 17 *****
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.
- 2622 ***** 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
- 2658 ***** 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.

- 2749 ***** 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)
THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
- 2830 ***** TEST 23 *****
THIS TEST WILL PROVE THAT EACH RECEIVING LINE CAN BE DISABLED BY SETTING THE RCVON BIT TO ZERO FOR EACH LINE IN THE LPR REGISTER. IT ALSO VERIFIES THAT MASTER CLEAR WILL ZERO DVALID FOR CHARACTERS STORED IN THE SILO.
- 2915 ***** TEST 24 *****
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 AN H325, H3271, OR H3190 CONNECTOR
- 2982 ***** TEST 25 *****
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).
- 3051 ***** TEST 26 *****
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.
- 3124 ***** TEST 27 *****
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
- 3234 ***** TEST 30 *****
TEST TO VERIFY THAT 'RDONE DOES NOT SET IF THE SCANNER IS DISABLED.
TURN ON SCANNER, WAIT FOR TRDY.
TURN OFF SCANNER, TRANSMIT A CHARACTER
'RDONE SHOULD NOT SET.

- 3280 ***** TEST 31 *****
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
- 3285 EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
CHAR PULLED OUT OF THE SILO.
USING SWITCH NINE FOR THIS TEST SENDC 20. CHARACTERS
ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
USED TO SCOPE SILO ALARM PULSES, ETC.
- 3415 ***** TEST 32 *****
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.
- 3500 ***** TEST 33 *****
THIS TEST RUNS ALL LINES FULL BORE
BASED UPON QUALIFIED LINES
..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
TRANSMITTER
- 3644 ***** TEST 34 *****
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.
- 3743 ***** TEST 35 *****
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 "STAGGERED" MODE.
40(8) CHARS ARE USED FOR THIS TEST.
ALL SELECTED LINES WILL BE ENABLED
AT THE SAME TIME!

3800 ***** TEST 36 *****
 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!

3985 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

4022 PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTE

4208

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!

ENDCOM	132#													
ERROR	26#	2094	2112	2125	2144	2150	2157	2176	2182	2189	2208	2214	222	
ESCAPE	132#													
GETPRI	132#													
GETSWR	132#													
MULT	132#													
NEWST	132#	2059	2102	2134	2166	2198	2230	2262	2293	2336	2390	2428	246	
PASEND	1#	1095												
POP	132#	1349	1350	1736	1737									
PRGEND	1#	1082												
PRGFRT	1#	3												
PUSH	132#	1310	1312	1333	1717	1723								
REPORT	1#	132#												
SC	1#	1161												
SCOPE	27#	1095	2060	2103	2135	2167	2199	2231	2263	2294	2337	2391	242	
SC1	1#	1201												
SETPRI	132#													
SETUP	132#													
SKIP	132#													
SLASH	132#													
SPACE	132#													
STARS	132#	332	354	405	408	543	545	552	1089	1150	1226	1305	171	
SWRSU	132#													
TYPBIN	132#													
TYPDEC	132#													
TYPNAM	132#													
TYPNUM	132#													
TYPOCS	132#													
TYPOCT	132#													
TYPTXT	132#													
\$BUFFE	1#	1787												
\$CYCLE	1#	1814												
\$EOP	1#	1082												
\$GETFL	1#	896												
\$GETPA	1#	835	848	861	903	959	992	1050	1862					

. ABS. 031110 000

ERRORS DETECTED: 0

DZDZAE, DZDZAE/SOL/CRF/NL: TOC=DZDZAE.P11
RUN-TIME: 29 20 2 SECONDS
RUN-TIME RATIO: $246/52=4.6$
CORE USED: 36K (71 PAGES)

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

```

.TITLE MD-11-DZDZA-E
.*COPYRIGHT (C) 1977
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
.*
$TN=1
000001 .STARTING PROCEDURE
        .LOAD PROGRAM
        .LOAD ADDRESS 000200
        .PRESS START
        .PROGRAM WILL TYPE "MAINDEC-11-DZDZAE/<200>/EIGHT LINE ASYNC MUX TESTS"
        .PROGRAM WILL TYPE "RUNNING" TO INDICATE THAT TESTING HAS STARTED
        .AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
        .AND THEN RESUME TESTING

.SBTTL BASIC DEFINITIONS
001120 .*INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***
        STACK= 1120
        .EQUIV EMT,ERROR          ;;BASIC DEFINITION OF ERROR CALL
        .EQUIV IOT,SCOPE          ;;BASIC DEFINITION OF SCOPE CALL

.*MISCELLANEOUS DEFINITIONS
000011 HT= 11          ;;CODE FOR HORIZONTAL TAB
000012 LF= 12          ;;CODE FOR LINE FEED
000015 CR= 15          ;;CODE FOR CARRIAGE RETURN
000200 CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776 PS= 177776     ;;PROCESSOR STATUS WORD
        .EQUIV PS,PSW
177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
177772 PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR= 177570   ;;HARDWARE SWITCH REGISTER
177570 DDISP= 177570  ;;HARDWARE DISPLAY REGISTER

.*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0= %0          ;;GENERAL REGISTER
000001 R1= %1          ;;GENERAL REGISTER
000002 R2= %2          ;;GENERAL REGISTER
000003 R3= %3          ;;GENERAL REGISTER
000004 R4= %4          ;;GENERAL REGISTER
000005 R5= %5          ;;GENERAL REGISTER
000006 R6= %6          ;;GENERAL REGISTER
000007 R7= %7          ;;GENERAL REGISTER
000006 SP= %6          ;;STACK POINTER
000007 PC= %7          ;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
000000 PRO= 0          ;;PRIORITY LEVEL 0
000040 PR1= 40         ;;PRIORITY LEVEL 1
000100 PR2= 100        ;;PRIORITY LEVEL 2

```

```

57      000140      PR3=      140      ::PRIORITY LEVEL 3
58      000200      PR4=      200      ::PRIORITY LEVEL 4
59      000240      PR5=      240      ::PRIORITY LEVEL 5
60      000300      PR6=      300      ::PRIORITY LEVEL 6
61      000340      PR7=      340      ::PRIORITY LEVEL 7
62
63      :*"SWITCH REGISTER" SWITCH DEFINITIONS
64      100000      SW15=     100000
65      040000      SW14=     40000
66      020000      SW13=     20000
67      010000      SW12=     10000
68      004000      SW11=     4000
69      002000      SW10=     2000
70      001000      SW09=     1000
71      000400      SW08=     400
72      000200      SW07=     200
73      000100      SW06=     100
74      000040      SW05=     40
75      000020      SW04=     20
76      000010      SW03=     10
77      000004      SW02=     4
78      000002      SW01=     2
79      000001      SW00=     1
80      .EQUIV      SW09,SW9
81      .EQUIV      SW08,SW8
82      .EQUIV      SW07,SW7
83      .EQUIV      SW06,SW6
84      .EQUIV      SW05,SW5
85      .EQUIV      SW04,SW4
86      .EQUIV      SW03,SW3
87      .EQUIV      SW02,SW2
88      .EQUIV      SW01,SW1
89      .EQUIV      SW00,SW0
90
91      :*DATA BIT DEFINITIONS (BIT00 TO BIT15)
92      100000      BIT15=    100000
93      040000      BIT14=    40000
94      020000      BIT13=    20000
95      010000      BIT12=    10000
96      004000      BIT11=    4000
97      002000      BIT10=    2000
98      001000      BIT09=    1000
99      000400      BIT08=    400
100     000200      BIT07=    200
101     000100      BIT06=    100
102     000040      BIT05=    40
103     000020      BIT04=    20
104     000010      BIT03=    10
105     000004      BIT02=    4
106     000002      BIT01=    2
107     000001      BIT00=    1
108     .EQUIV      BIT09,BIT9
109     .EQUIV      BIT08,BIT8
110     .EQUIV      BIT07,BIT7
111     .EQUIV      BIT06,BIT6
112     .EQUIV      BIT05,BIT5

```

```

113 .EQUIV BIT04,BIT4
114 .EQUIV BIT03,BIT3
115 .EQUIV BIT02,BIT2
116 .EQUIV BIT01,BIT1
117 .EQUIV BIT00,BIT0
118
119 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
120 000004 ERRVEC= 4 ; TIME OUT AND OTHER ERRORS
121 000010 RESVEC= 10 ; RESERVED AND ILLEGAL INSTRUCTIONS
122 000014 TBITVEC=14 ; "T" BIT
123 000014 TRTVEC= 14 ; TRACE TRAP
124 000014 BPTVEC= 14 ; BREAKPOINT TRAP (BPT)
125 000020 IOTVEC= 20 ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
126 000024 PWRVEC= 24 ; POWER FAIL
127 000030 EMTVEC= 30 ; EMULATOR TRAP (EMT) **ERROR**
128 000034 TRAPVEC=34 ; "TRAP" TRAP
129 000060 TKVEC= 60 ; TTY KEYBOARD VECTOR
130 000064 TPVEC= 64 ; TTY PRINTER VECTOR
131 000240 PIRQVEC=240 ; PROGRAM INTERRUPT REQUEST VECTOR
132
133
134 ; INSTRUCTION DEFINITIONS
135 ;-----
136
137 005746 PUSH1SP=5746 ; DECREMENT PROCESSOR STACK 1 WORD
138 005726 POP1SP=5726 ; INCREMENT PROCESSOR STACK 1 WORD
139 010046 PUSHRO=10046 ; SAVE RO ON STACK
140 012600 POPRO=12600 ; RESTORE RO FROM STACK
141 024646 PUSH2SP=24646 ; DECREMENT STACK TWICE
142 022626 POP2SP=22626 ; INCREMENT STACK TWICE
143
144 ; DZ11 CONTROL AND STATUS REGISTER DEFINITIONS
145 ; (DZCSR) BIT DEFINITIONS
146 ;-----
147
148 000010 MAINT = BIT3 ; MAINTENANCE MODE ENABLE
149 000020 DCLR=BIT4 ; DEVICE CLEAR
150 000040 MSENAB=BIT5 ; MASTER SCAN ENABLE
151 000100 RIE=BIT6 ; RECEIVER INTERRUPT ENABLE
152 000200 RDONE=BIT7 ; RECEIVER DONE
153 010000 SILOEN= BIT12 ; SILO ALARM ENABLE
154 020000 SILOAL = BIT13 ; SILO ALARM
155 040000 TIE=BIT14 ; TRANSMITTER INTERRUPT ENABLE
156 100000 TRDY=BIT15 ; TRANSMITTER READY
157
158 ; DZCSR WORD DEFINITIONS
159 ;-----
160 000000 TLO=0 ; TRANSMIT LINE 0
161 000400 TL1=BIT8 ; TRANSMIT LINE 1
162 001000 TL2=BIT9 ; TRANSMIT LINE 2
163 001400 TL3=BIT9:BIT8 ; TRANSMIT LINE 3
164 002000 TL4=BIT10 ; TRANSMIT LINE 4
165 002400 TL5=BIT10:BIT8 ; TRANSMIT LINE 5
166 003000 TL6=BIT10:BIT9 ; TRANSMIT LINE 6
167 003400 TL7=BIT10:BIT9:BIT8 ; TRANSMIT LINE 7
168
    
```



```

169
170
171
172
173 010000
174 020000
175 040000
176 100000
177
178
179
180
181 000000
182 000400
183 001000
184 001400
185 002000
186 002400
187 003000
188 003400
189
190
191
192
193 000000
194 000001
195 000002
196 000003
197 000004
198 000005
199 000006
200 000007
201
202 000000
203 000010
204 000020
205 000030
206 000040
207 000050
208 000060
209 000070
210
211 000100
212 000200
213 000000
214 000040
215 000000
216 010000
217
218 000000
219 000400
220 001000
221 001400
222 002000
223 002400
224 003000

```

```

;DZRBUF BIT DEFINITIONS
-----
PARER=BIT12 ;PARITY ERROR
FRMERR=BIT13 ;FRAME ERROR
OVRRUN=BIT14 ;OVERRUN ERROR
DVALID=BIT15 ;DATA VALID

;DZRBUF WORD DEFINITIONS
-----
RL0=0 ;RECEIVER LINE 0
RL1=BIT8 ;RECEIVER LINE 1
RL2=BIT9 ;RECEIVER LINE 2
RL3=BIT9!BIT8 ;RECEIVER LINE 3
RL4=BIT10 ;RECEIVER LINE 4
RL5=BIT10!BIT8 ;RECEIVER LINE 5
RL6=BIT10!BIT9 ;RECEIVER LINE 6
RL7=BIT10!BIT9!BIT8 ;RECEIVER LINE 7

;DZLPR WORD DEFINITIONS
-----
LP0=0 ;LINE PARAMETER 0
LP1=BIT0 ;LINE PARAMETER 1
LP2=BIT1 ;LINE PARAMETER 2
LP3=BIT1!BIT0 ;LINE PARAMETER 3
LP4=BIT2 ;LINE PARAMETER 4
LP5=BIT2!BIT0 ;LINE PARAMETER 5
LP6=BIT2!BIT1 ;LINE PARAMETER 6
LP7=BIT2!BIT1!BIT0 ;LINE PARAMETER 7

FIVE=0 ;FIVE BITS/CHAR, 1 STOP BIT
SIX=BIT3 ;SIX BITS/CHAR, 1 STOP BIT
SEVEN=BIT4 ;SEVEN BITS/CHAR, 1 STOP BIT
EIGHT=BIT4!BIT3 ;EIGHT BITS/CHAR, 1 STOP BIT
FIVES=BIT5 ;FIVE BITS/CHAR, 2 STOP BITS
SIXS=BIT5!BIT3 ;SIX BITS/CHAR, 2 STOP BITS
SEVENS=BIT5!BIT4 ;SEVEN BITS/CHAR, 2 STOP BITS
EIGHTS=BIT5!BIT4!BIT3 ;EIGHT BITS/CHAR, 2 STOP BITS

PARITY=BIT6 ;PARITY ENABLED
ODDPAR=BIT7 ;ODD PARITY ENABLED
ONESTOP=0 ;ONE STOP BIT ENABLED
TWOSTOP=BITS ;TWO STOP BITS ENABLED
EVEPAR=0 ;EVEN PARITY ENABLED
RCVON=BIT12 ;ENABLE RECEIVER (RECEIVER ON)

SS0=0 ;SPEED 50 BAUD
S75=BIT8 ;SPEED 75 BAUD
S110=BIT9 ;SPEED 110 BAUD
S134=BIT9!BIT8 ;SPEED 134.5 BAUD
S150=BIT10 ;SPEED 150 BAUD
S300=BIT10!BIT8 ;SPEED 300 BAUD
S600=BIT10!BIT9 ;SPEED 600 BAUD

```

225	003400	S1200=BIT10!BIT9!BIT8	:SPEED 1200 BAUD
226	004000	S1800=BIT11	:SPEED 1800 BAUD
227	004400	S2000=BIT11!BIT8	:SPEED 2000 BAUD
228	005000	S2400=BIT11!BIT9	:SPEED 2400 BAUD
229	005400	S3600=BIT11!BIT9!BIT8	:SPEED 3600 BAUD
230	006000	S4800=BIT11!BIT10	:SPEED 4800 BAUD
231	006400	S7200=BIT11!BIT10!BIT8	:SPEED 7200 BAUD
232	007000	S9600=BIT11!BIT10!BIT9	:SPEED 9600 BAUD
233	007400	S19200=BIT11!BIT10!BIT9!BIT8	:SPEED 19200 BAUD

:DZTCR BIT DEFINITIONS

234		-----	
235			
236			
237	000001	TCR0=BIT0	:TCR0
238	000002	TCR1=BIT1	:TCR1
239	000004	TCR2=BIT2	:TCR2
240	000010	TCR3=BIT3	:TCR3
241	000020	TCR4=BIT4	:TCR4
242	000040	TCR5=BIT5	:TCR5
243	000100	TCR6=BIT6	:TCR6
244	000200	TCR7=BIT7	:TCR7
245	000400	DTR0=BIT8	:DTR0
246	001000	DTR1=BIT9	:DTR1
247	002000	DTR2=BIT10	:DTR2
248	004000	DTR3=BIT11	:DTR3
249	010000	DTR4=BIT12	:DTR4
250	020000	DTR5=BIT13	:DTR5
251	040000	DTR6=BIT14	:DTR6
252	100000	DTR7=BIT15	:DTR7

:DZMSR BIT DEFINITIONS

253		-----	
254			
255			
256	000001	RING0=BIT0	:RING INDICATED ON LINE 0
257	000002	RING1=BIT1	:RING INDICATED ON LINE 1
258	000004	RING2=BIT2	:RING INDICATED ON LINE 2
259	000010	RING3=BIT3	:RING INDICATED ON LINE 3
260	000020	RING4=BIT4	:RING INDICATED ON LINE 4
261	000040	RING5=BIT5	:RING INDICATED ON LINE 5
262	000100	RING6=BIT6	:RING INDICATED ON LINE 6
263	000200	RING7=BIT7	:RING INDICATED ON LINE 7
264	000400	C00=BIT8	:CARRIER PRESENT ON LINE 0
265	001000	C01=BIT9	:CARRIER PRESENT ON LINE 1
266	002000	C02=BIT10	:CARRIER PRESENT ON LINE 2
267	004000	C03=BIT11	:CARRIER PRESENT ON LINE 3
268	010000	C04=BIT12	:CARRIER PRESENT ON LINE 4
269	020000	C05=BIT13	:CARRIER PRESENT ON LINE 5
270	040000	C06=BIT14	:CARRIER PRESENT ON LINE 6
271	100000	C07=BIT15	:CARRIER PRESENT ON LINE 7

:DZTDR BIT DEFINITIONS

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

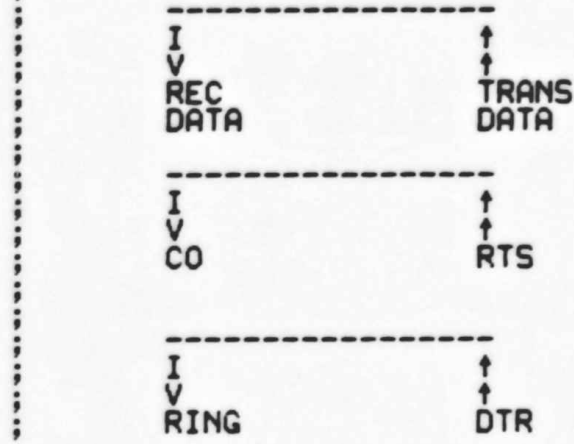
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303

020000
040000
100000

BRK5=BIT13
BRK6=BIT14
BRK7=BIT15

;BREAK FOR LINE 5
;BREAK FOR LINE 6
;BREAK FOR LINE 7

:TABLE OF LOOP AROUND FUNCTIONS (H325)



```

304 ;*****
305 ;-----
306 ; TRAPCATCHER FOR ILLEGAL INTERRUPTS
307 ; THE STANDARD "TRAP CATCHER" IS PLACED
308 ; BETWEEN ADDRESS 0 TO ADDRESS 776.
309 ; IT LOOKS LIKE "PC+2 HALT".
310 ;-----
311 ;*****
312
313         000000
314         .=0
315         ; STANDARD INTERRUPT VECTORS
316         ;-----
317         .=10
318         SET.PS           ;FAKE "MTPS" INSTRUCTION TRAP
319         PR7              ;MAKE SURE PS IS PRIORITY 7
320
321         .=20
322         .SCOPE           ;SCOPE LOOP HANDLER
323         PR7              ;HANDLE AT PRIORITY 7
324         $PWRDN          ;POWER FAIL HANDLER
325         340             ;SERVICE AT PRIORITY LEVEL 7
326         $ERROR         ;ERROR HANDLER
327         340             ;SERVICE AT PRIORITY LEVEL 7
328         .TRPSRV        ;GENERAL HANDLER DISPATCH SERVICE
329         340             ;SERVICE AT PRIORITY LEVEL 7
330         .SBTTL ACT11 HOOKS
331
332 ;*****
333 ;HOOKS REQUIRED BY ACT11
334         $$VPC=.         ;SAVE PC
335         .=46           ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
336         $ENDAD         ;;2)SET LOC.52 TO ZERO
337         .=52           ;; RESTOPE PC
338         .WORD 0
339         .=$VPC
340
341         .=174
342         DISPREG:0      ;SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 115
343         SWREG: 0       ;SOFTWARE SWITCH REGISTER FOR SWITCHLESS 115
344         .=200
345         JMP .START    ;GO TO START OF PROGRAM
346         .=210
347         JMP XSTART    ;GOTO CABLE TEST/ECHO TEST
348
349
350         .=1000
351         MTITLE: .ASCIZ <200><12>/MAINDEC-11-DZDZAE/<200>/EIGHT LINE ASYNC MUX TESTS/<200>
(2)
    
```

```

352          .SBTTL  COMMON TAGS
353
354          ;:*****
355          ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
356          ;*USED IN THE PROGRAM.
357
358          001120          .=1120          ;; START OF COMMON TAGS
359          001120          000000          $CMTAG: .WORD 0
360          001122          000          $STNM: .BYTE 0
361          001122          000          $ERFLG: .BYTE 0
362          001123          000          $ICNT: .WORD 0
363          001124          000000          $LPADR: .WORD 0
364          001126          000000          $LPERR: .WORD 0
365          001130          000000          $ERTTL: .WORD 0
366          001132          000000          $ITEMB: .BYTE 0
367          001134          000          $ERMAX: .BYTE 1
368          001135          001          $ERRPC: .WORD 0
369          001136          000000          $GDADR: .WORD 0
370          001140          000000          $BDADR: .WORD 0
371          001142          000000          $GDDAT: .WORD 0
372          001144          000000          $BDDAT: .WORD 0
373          001146          000000          .WORD 0
374          001150          000000          .WORD 0
375          001152          000000          .WORD 0
376          001154          000          $AUTOB: .BYTE 0
377          001155          000          $INTAG: .BYTE 0
378          001156          000000          .WORD 0
379          001160          177570          $SWR: .WORD DSWR
380          001162          177570          $DISPLAY: .WORD DDISP
381          001164          177560          $TKS: 177560
382          001166          177562          $TKB: 177562
383          001170          177564          $TPS: 177564
384          001172          177566          $TPB: 177566
385          001174          000          $NULL: .BYTE 0
386          001175          002          $FILLS: .BYTE 2
387          001176          012          $FILLC: .BYTE 12
388          001177          000          $TPFLG: .BYTE 0
389          001200          000000          $REGAD: .WORD 0
390
391          001202          000000          $REG0: .WORD 0
392          001204          000000          $REG1: .WORD 0
393          001206          000000          $REG2: .WORD 0
394          001210          000000          $REG3: .WORD 0
395          001212          000000          $REG4: .WORD 0
396          001214          000000          $REG5: .WORD 0
397          001216          000000          $TMP0: .WORD 0
398          001220          000000          $TMP1: .WORD 0
399          001222          000000          $TMP2: .WORD 0
400          001224          000000          $TMP3: .WORD 0
401          001226          000000          $TIMES: 0
402          001230          077          $QUES: .ASCII /?/
403          001231          015          $CRLF: .ASCII <15>
404          001232          000012          $LF: .ASCIZ <12>
405          ;:*****
406          .SBTTL  APT MAILBOX-ETABLE
407

```

```

408 ;:*****
409 .EVEN
410 $MAIL: .WORD AMSGTY :; APT MAILBOX
411 $MSGTY: .WORD AFATAL :; MESSAGE TYPE CODE
412 $FATAL: .WORD ATESTN :; FATAL ERROR NUMBER
413 $TESTN: .WORD APASS :; TEST NUMBER
414 $PASS: .WORD ADEVCT :; PASS COUNT
415 $DEVCT: .WORD AUNIT :; DEVICE COUNT
416 $UNIT: .WORD AMSGAD :; I/O UNIT NUMBER
417 $MSGAD: .WORD AMSGLG :; MESSAGE ADDRESS
418 $MSGLG: .WORD AMSGLG :; MESSAGE LENGTH
419 $ETABLE: .WORD AENV :; APT ENVIRONMENT TABLE
420 $ENV: .BYTE AENVM :; ENVIRONMENT BYTE
421 $ENVM: .BYTE ASWREG :; ENVIRONMENT MODE BITS
422 $SWREG: .WORD AUSWR :; APT SWITCH REGISTER
423 $USWR: .WORD ACPUOP :; USER SWITCHES
424 $CPUOP: .WORD ACPUOP :; CPU TYPE, OPTIONS
425 *
426 * BIT 15-11=CPU TYPE
427 * 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
428 * 11/70=06, PDQ=07, Q=10
429 *
430 * BIT 10=REAL TIME CLOCK
431 * BIT 9=FLOATING POINT PROCESSOR
432 * BIT 8=MEMORY MANAGEMENT
433 $MAMS1: .BYTE AMAMS1 :; HIGH ADDRESS, M.S. BYTE
434 $MTYP1: .BYTE AMTYP1 :; MEM. TYPE, BLK#1
435 * MEM. TYPE BYTE -- (HIGH BYTE)
436 * 900 NSEC CORE=001
437 * 300 NSEC BIPOLAR=002
438 * 500 NSEC MOS=003
439 $MADR1: .WORD AMADR1 :; HIGH ADDRESS, BLK#1
440 * MEM.LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
441 $MAMS2: .BYTE AMAMS2 :; HIGH ADDRESS, M.S. BYTE
442 $MTYP2: .BYTE AMTYP2 :; MEM. TYPE, BLK#2
443 $MADR2: .WORD AMADR2 :; MEM.LAST ADDRESS, BLK#2
444 $MAMS3: .BYTE AMAMS3 :; HIGH ADDRESS, M.S. BYTE
445 $MTYP3: .BYTE AMTYP3 :; MEM. TYPE, BLK#3
446 $MADR3: .WORD AMADR3 :; MEM.LAST ADDRESS, BLK#3
447 $MAMS4: .BYTE AMAMS4 :; HIGH ADDRESS, M.S. BYTE
448 $MTYP4: .BYTE AMTYP4 :; MEM. TYPE, BLK#4
449 $MADR4: .WORD AMADR4 :; MEM.LAST ADDRESS, BLK#4
450 $SVECT1: .WORD AVECT1 :; INTERRUPT VECTOR#1, BUS PRIORITY#1
451 $SVECT2: .WORD AVECT2 :; INTERRUPT VECTOR#2, BUS PRIORITY#2
452 $BASE: .WORD ABASE :; BASE ADDRESS OF EQUIPMENT UNDER TEST
453 $DEVN: .WORD ADEVN :; DEVICE MAP
454 $SCDW1: .WORD ACDW1 :; CONTROLLER DESCRIPTION WORD#1
455 $SCDW2: .WORD ACDW2 :; CONTROLLER DESCRIPTION WORD#2
456 $SDDW0: .WORD ADDW0 :; DEVICE DESCRIPTOR WORD#0
457 $SDDW1: .WORD ADDW1 :; DEVICE DESCRIPTOR WORD#1
458 $SDDW2: .WORD ADDW2 :; DEVICE DESCRIPTOR WORD#2
459 $SDDW3: .WORD ADDW3 :; DEVICE DESCRIPTOR WORD#3
460 $SDDW4: .WORD ADDW4 :; DEVICE DESCRIPTOR WORD#4
461 $SDDW5: .WORD ADDW5 :; DEVICE DESCRIPTOR WORD#5
462 $SDDW6: .WORD ADDW6 :; DEVICE DESCRIPTOR WORD#6
463 $SDDW7: .WORD ADDW7 :; DEVICE DESCRIPTOR WORD#7
464 $SDDW8: .WORD ADDW8 :; DEVICE DESCRIPTOR WORD#8
465 $SDDW9: .WORD ADDW9 :; DEVICE DESCRIPTOR WORD#9

```


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
512

.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

001360

\$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 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


```

513
514
515
516
517 001414 000
518 001415 000
519 001416 000
520 001417 000
521 001420 000
522 001422
523
524 001422 000000
525 001424 000000
526 001426 000000
527 001430 000000
528 001432 000000
529 001434 000000
530 001436 000000
531 001440 000000
532 001442 000000
533 001444 000000
534 001446 000000
535 001450 000000
536 001452 000000
537 001454 000000
538 001456 000000
539 001460 000000
540 001462
541
542
543
544
545
546 001462
547 000024 000024
548 000024 000200
549 000044 000044
550 000044 001462
551 001462
552
553
554
555
556 001462
557 001462 000000
558 001464 001234
559 001466 000132
560 001470 000137
561 001472 000137
562 001474 000052
563
564
565
566 001500 001500
567
568

```

```

;PROGRAM CONTROL FLAGS
-----
EIAFLG: .BYTE 0 ;0=EIA 100000=20MA
INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
HDRFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
MNTFLG: .BYTE 0 ;MAINTENANCE BIT SET FLAG
DNMFLG: .BYTE 0 ;TRANSMISSION COMPLETION FLAG
.EVEN
;DATA VARIABLES
TD0: .WORD 0
TD1: .WORD 0
TD2: .WORD 0
TD3: .WORD 0
TD4: .WORD 0
TD5: .WORD 0
TD6: .WORD 0
TD7: .WORD 0
TR0: .WORD 0
TR1: .WORD 0
TR2: .WORD 0
TR3: .WORD 0
TR4: .WORD 0
TR5: .WORD 0
TR6: .WORD 0
TR7: .WORD 0
STOP:
.SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.SX= . ;SAVE CURRENT LOCATION
.=24 . ;SET POWER FAIL TO POINT TO START OF PROGRAM
200 . ;FOR APT START UP
.=44 . ;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR . ;POINT TO APT HEADER BLOCK
.=.SX . ;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.
$APTHD:
$HIBTS: .WORD 0 ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT: .WORD 90 ;RUN TIM OF LONGEST TEST
$PASTM: .WORD 95 ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 95 ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD $ETEND-$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)
;DZ11 STATUS TABLE AND ADDRESS ASSIGNMENTS
-----
.=1500
DZ.MAP:

```

569	001500	000001	DZCR0:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 0
570	001502	000001	DZVC0:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 0
571	001504	000001	DZLV0:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
572	001506	000001	LINE0:	.BLKW	1	:ALL LINES SELECTED
573	001510	000001	PAR0:	.BLKW	1	:PARAMETERS
574	001512	000001	MANT0:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
575						
576	001514	000001	DZCR1:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 1
577	001516	000001	DZVC1:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 1
578	001520	000001	DZLV1:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
579	001522	000001	LINE1:	.BLKW	1	:ALL LINES SELECTED
580	001524	000001	PAR1:	.BLKW	1	:PARAMETERS
581	001526	000001	MANT1:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
582						
583	001530	000001	DZCR2:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 2
584	001532	000001	DZVC2:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 2
585	001534	000001	DZLV2:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
586	001536	000001	LINE2:	.BLKW	1	:ALL LINES SELECTED
587	001540	000001	PAR2:	.BLKW	1	:PARAMETERS
588	001542	000001	MANT2:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
589						
590	001544	000001	DZCR3:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 3
591	001546	000001	DZVC3:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 3
592	001550	000001	DZLV3:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
593	001552	000001	LINE3:	.BLKW	1	:ALL LINES SELECTED
594	001554	000001	PAR3:	.BLKW	1	:PARAMETERS
595	001556	000001	MANT3:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
596						
597	001560	000001	DZCR4:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 4
598	001562	000001	DZVC4:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 4
599	001564	000001	DZLV4:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
600	001566	000001	LINE4:	.BLKW	1	:ALL LINES SELECTED
601	001570	000001	PAR4:	.BLKW	1	:PARAMETERS
602	001572	000001	MANT4:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
603						
604	001574	000001	DZCR5:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 5
605	001576	000001	DZVC5:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 5
606	001600	000001	DZLV5:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
607	001602	000001	LINE5:	.BLKW	1	:ALL LINES SELECTED
608	001604	000001	PAR5:	.BLKW	1	:PARAMETERS
609	001606	000001	MANT5:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
610						
611	001610	000001	DZCR6:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 6
612	001612	000001	DZVC6:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 6
613	001614	000001	DZLV6:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
614	001616	000001	LINE6:	.BLKW	1	:ALL LINES SELECTED
615	001620	000001	PAR6:	.BLKW	1	:PARAMETERS
616	001622	000001	MANT6:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
617						
618	001624	000001	DZCR7:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZ11 NUMBER 7
619	001626	000001	DZVC7:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 7
620	001630	000001	DZLV7:	.BLKW	1	:PRIORITY LEVEL AND EIA FLAG SELECTOR
621	001632	000001	LINE7:	.BLKW	1	:ALL LINES SELECTED
622	001634	000001	PAR7:	.BLKW	1	:PARAMETERS
623	001636	000001	MANT7:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
624						

625	001640	000001	DZCR10: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 10
626	001642	000001	DZVC10: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 10
627	001644	000001	DZLV10: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
628	001646	000001	LINE10: .BLKW	1	: ALL LINES SELECTED
629	001650	000001	PAR10: .BLKW	1	: PARAMETERS
630	001652	000001	MANT10: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
631					
632	001654	000001	DZCR11: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 11
633	001656	000001	DZVC11: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 11
634	001660	000001	DZLV11: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
635	001662	000001	LINE11: .BLKW	1	: ALL LINES SELECTED
636	001664	000001	PAR11: .BLKW	1	: PARAMETERS
637	001666	000001	MANT11: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
638					
639	001670	000001	DZCR12: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 12
640	001672	000001	DZVC12: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 12
641	001674	000001	DZLV12: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
642	001676	000001	LINE12: .BLKW	1	: ALL LINES SELECTED
643	001700	000001	PAR12: .BLKW	1	: PARAMETERS
644	001702	000001	MANT12: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
645					
646	001704	000001	DZCR13: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 13
647	001706	000001	DZVC13: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 13
648	001710	000001	DZLV13: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
649	001712	000001	LINE13: .BLKW	1	: ALL LINES SELECTED
650	001714	000001	PAR13: .BLKW	1	: PARAMETERS
651	001716	000001	MANT13: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
652					
653	001720	000001	DZCR14: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 14
654	001722	000001	DZVC14: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 14
655	001724	000001	DZLV14: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
656	001726	000001	LINE14: .BLKW	1	: ALL LINES SELECTED
657	001730	000001	PAR14: .BLKW	1	: PARAMETERS
658	001732	000001	MANT14: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
659					
660	001734	000001	DZCR15: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 15
661	001736	000001	DZVC15: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 15
662	001740	000001	DZLV15: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
663	001742	000001	LINE15: .BLKW	1	: ALL LINES SELECTED
664	001744	000001	PAR15: .BLKW	1	: PARAMETERS
665	001746	000001	MANT15: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
666					
667	001750	000001	DZCR16: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 16
668	001752	000001	DZVC16: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 16
669	001754	000001	DZLV16: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
670	001756	000001	LINE16: .BLKW	1	: ALL LINES SELECTED
671	001760	000001	PAR16: .BLKW	1	: PARAMETERS
672	001762	000001	MANT16: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
673					
674	001764	000001	DZCR17: .BLKW	1	: CONTROL STATUS REGISTER FOR DZ11 NUMBER 17
675	001766	000001	DZVC17: .BLKW	1	: RECEIVER AND BASE VECTOR FOR DZ11 NUMBER 17
676	001770	000001	DZLV17: .BLKW	1	: PRIORITY LEVEL AND EIA FLAG SELECTOR
677	001772	000001	LINE17: .BLKW	1	: ALL LINES SELECTED
678	001774	000001	PAR17: .BLKW	1	: PARAMETERS
679	001776	000001	MANT17: .BLKW	1	: MAINTENANCE MODE FOR THIS DEVICE
680					

E04

MD-11-DZDZA-E MACY11 30(1046) 03-OCT-77 09:43 PAGE 15
DZDZAE.P11 03-OCT-77 09:39 APT PARAMETER BLOCK

SEQ 0043

681 002000 177777

DZ.END: 177777

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
723

; 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
INSTER=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
;-----
; *****

```

```

724                                     ;DZ11 VECTOR AND REGISTER INDIRECT POINTERS
725                                     ;WORKING AREA
726
727 002042 160040 DZCSR: 160040 ;R/W
728 002044 160041 HDZCSR: 160041 ;R/W
729 002046 160042 DZRBUF: 160042 ;READ ONLY
730 002050 160043 HDZRBUF: 160043 ;READ ONLY
731 002052 160042 DZLPR: 160042 ;WRITE ONLY
732 002054 160043 HDZLPR: 160043 ;WRITE ONLY
733 002056 160044 DZTCR: 160044 ;R/W
734 002060 160045 HDZTCR: 160045 ;R/W
735 002062 160046 DZMSR: 160046 ;READ ONLY
736 002064 160047 HDZMSR: 160047 ;READ ONLY
737 002066 160046 DZTDR: 160046 ;WRITE ONLY
738 002070 160047 HDZTDR: 160047 ;WRITE ONLY
739                                     ;DEFAULT DZ VECTORS
740 002072 000300 DZRIV: 300 ;REC INTR VECTOR
741 002074 000302 DZRIS: 302 ;REC INTR STATUS
742 002076 000304 DZTIV: 304 ;XMIT INTR VECTOR
743 002100 000306 DZTIS: 306 ;XMIT INTR STATUS
744
745
    
```



```

770
771      :PROGRAM INITIALIZATION
772      :LOCK OUT INTERRUPTS
773      :SET UP PROCESSOR STACK
774      :SET UP POWER FAIL VECTOR
775      :CLEAR PROGRAM CONTROL FLAGS AND COUNTS
776      :TYPE TITLE MESSAGE
777
778      .START:
779      002150 000005      RESET      ;CLEAR THE WORLD. START NEW ENVIRONMENT
780      002152 012706 001120      MOV      #STACK,SP      ;SET UP STACK
781      002156 106427 000340      MTPS     #PR7          ;LOCK OUT INTERRUPTS
782      002162 012737 007646 000024      MOV      #SPWRDN,2#24    ;SET UP POWER FAIL VECTOR
783      002170 113737 001410 001411      MOV      DZNUM,SANUM     ;SAVE NUMBER OF DEVICES IN SYSTEM.
784      002176 005037 001242      CLR      $PASS          ;CLEAR PASS COUNT
785      002202 105037 001123      CLRB    $ERFLG         ;CLEAR ERROR FLAG
786      002206 012737 001500 001412      MOV      #DZ.MAP,ACTIVE ;GET MAP POINTER.
787      002214 012737 000001 001406      MOV      #1,RUN         ;POINT POINTER TO FIRST DEVICE.
788      002222 005037 001132      CLR      $ERTTL        ;CLEAR ERROR COUNT
789      002226 005037 001136      CLR      $ERRPC        ;CLEAR LAST ERROR POINTER
790      002232 005037 001122      CLR      $TSTNM        ;SET UP FOR TEST 1
791      002236 012737 002150 001126      MOV      #.START,$LPADR ;SET UP FOR POWER FAIL BEFORE
792
793      :SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
794      002244 013746 000006      MOV      6,-(SP)        ;SAVE BUS ERROR PS
795      002250 013746 000004      MOV      4,-(SP)        ;SAVE BUS ERROR PC
796      002254 012737 002274 000004      MOV      #20$,4         ;SET UP TO TRAP TO THIS ROUTINE
797      002262 022777 177777 176670      CMP      #-1,$SWR       ;CAN 177570 BE REFERENCED?
798      002270 001402      BEQ      22$            ;IF SO AND IT IS -1, TREAT LIKE SWITCHLESS
799      002272 000407      BR      21$            ;IF YES, SKIP AROUND THE SETUP
800      002274 022626      POP2SP   ;REMOVE THE TRAP FROM THE STACK
801      002276 012737 000176 001160 20$:      MOV      #SWREG,SWR     ;IF NO TRAP COMES HERE, POINT TO SOFTWARE SWR
802      002304 012737 000174 001162 22$:      MOV      #DISPREG,DISPLAY ;POINT TO SOFTWARE DISPLAY REGISTER
803      002312 012637 000004      MOV      (SP)+,4        ;RESTORE THE BUS ERROR VECTOR
804      002316 012637 000006      MOV      (SP)+,6
805      002322 105737 001415      TSTB    INIFLG         ;TITLE ALREADY PRINTED?
806      002326 001010      BNE     29$            ;BRANCH IF YES
807      002330 023727 000042 004726      CMP      2#42,$SENDAD   ;RUNNING UNDER ACT?
808      002336 001402      BEQ     31$            ;IF YES DONT PRINT TITLE
809      002340 104402 001000      TYPE    $MTITLE        ;PRINT THE DIAGNOSTIC'S TITLE
810      002344 105337 001415 31$:      DECB    INIFLG         ;SET THE ONCE ONLY FLAG
811      002350 105737 001255 29$:      TSTB    $ENVM          ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
812      002354 100006      BPL     30$            ;IF NOT, GO CHECK FOR AUTO-SIZING
813      002356 004737 011440      JSR     PC,SETAPT      ;OTHERWISE, GO DO APT SIZING FROM ETABLE
814      002362 105037 001416      CLRB    HDRFLG         ;MAKE SURE STATUS TABLE IS PRINTED
815      002366 000137 004270      JMP     16$            ;GO PRINT DZ STATUS TABLE
816      002372 032777 000001 176560 30$:      BIT     #SW00,$SWR     ;RESELECT ?
817      002400 001011      BNE     32$            ;IF YES, GO SET UP THE INFORMATION
818      002402 122737 000377 001415      CMPB    #377,INIFLG    ;ON 1ST START, MUST ANSWER QUESTION
819      002410 001003      BNE     +10           ;IF NOT ANSWERING QUESTIONS
820      002412 105777 176542      TSTB    $SWR           ;ARE U AUTO SIZING?
821      002416 100402      BMI     32$            ;NO AUTO SIZE! NO SW00=1 ON 1ST START!
822      002420 000137 003114      JMP     73$            ;IF NO, SKIP THE INTERROGATION
823      002424 012700 001500 32$:      MOV      #DZ.MAP,RO    ;POINT TO THE BEGINNING OF THE MAP TABLE
824      002430 105037 001416      CLRB    HDRFLG        ;MAKE SURE A MAP GETS PRINTED
825      002434 005020 65$:      CLR     (RO)+         ;CLEAR A TABLE LOCATION

```



```

826 002436 020027 002000          CMP      RO,#DZ.END      ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
827 002442 001374                BNE      65$            ;IF NOT, CLEAR THE NEXT LOCATION IN THE TABLE
828 002444 105337 001415          DECB     INIFLG         ;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!
829
830                                ;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
831                                ;TABLE AND SET UP THE DIAGNOSTIC.
832
833                                ;GET THE BASE ADDRESS OF THE DZ11'S
834
835 002450          33$:          INSTR     ;CALL THE STRING INPUT ROUTINE
836 002450          104403        66$      ;POINTER TO MESSAGE TO BE PRINTED
837 002452          003334        PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
838 002454          104405        160000    ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
839 002456          160000        163770    ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
840 002460          163770        DZCRO    ;POINTER TO MAP LOCATION TO BE FILLED
841 002462          001500        .BYTE    7      ;MASK OF INVALID BITS FOR THIS PARAMETER
842 002464          007          .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
843 002465          001          MOV      DZCRO,$BASE ;COPY BASE ADDRESS TO ETABLE
844 002466          013737 001500 001310
845
846                                ;GET THE BASE VECTOR ADDRESS
847
848 002474          34$:          INSTR     ;CALL THE STRING INPUT ROUTINE
849 002474          104403        67$      ;POINTER TO MESSAGE TO BE PRINTED
850 002476          003400        PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
851 002500          104405        300      ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
852 002502          000300        776      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
853 002504          000776        DZVCO    ;POINTER TO MAP LOCATION TO BE FILLED
854 002506          001502        .BYTE    3      ;MASK OF INVALID BITS FOR THIS PARAMETER
855 002510          003          .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
856 002511          001          MOV      DZVCO,$VECT1 ;COPY VECTOR TO ETABLE
857 002512          013737 001502 001304
858
859                                ;GET THE BUS REQUEST LEVEL
860
861 002520          104403        INSTR     ;CALL THE STRING INPUT ROUTINE
862 002522          003441        68$      ;POINTER TO MESSAGE TO BE PRINTED
863 002524          104405        PARAM     ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
864 002526          000004        4      ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
865 002530          000007        7      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
866 002532          001504        DZLVO    ;POINTER TO MAP LOCATION TO BE FILLED
867 002534          000          .BYTE    0      ;MASK OF INVALID BITS FOR THIS PARAMETER
868 002535          001          .BYTE    1      ;NUMBER OF PARAMETERS TO STORE
869 002536          113737 001504 001305    MOVB     DZLVO,$VECT1+1 ;GET BUS REQUEST LEVEL INTO ETABLE
870 002544          106337 001305          ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
871 002550          106337 001305          ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
872 002554          106337 001305          ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
873 002560          106337 001305          ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
874 002564          106337 001305          ASLB     $VECT1+1      ;ALIGN THE BITS PROPERLY
875
876                                ;FIND OUT IF MODULE IS EIA OR 20 MA.
877
878 002570          104402 004130          TYPE     74$          ;PRINT EIA MESSAGE
879 002574          005037 001220          CLR      $TMP1        ;USE $TMP1
880 002600          105777 176360          TSTB     @$TKS        ;IS KEYBOARD DONE?
881 002604          100375          BPL      80$          ;IF NOT, WAIT FOR IT
    
```

```

882 002606 017746 176354      MOV      @STKB, -(SP)      ; IF YES, PUT CHARACTER ON STACK
883 002612 042716 000240      BIC      #240, (SP)      ; STRIP DOWN CHARACTER
884 002616 122726 000015      CMPB     #15, (SP)+      ; IS IT ?
885 002622 001414              BEQ      #15, GET OUT    ; IF SO, GET OUT
886 002624 014677 176342      MOV      -(SP), @STPB    ; IF NOT, PRINT CHARACTER
887 002630 042737 100000 001504  BIC      #BIT15, DZLVO   ; CLEAR EIA FLAG
888 002636 122726 000102      CMPB     #102, (SP)+    ; IS IT A B?
889 002642 001356              BNE      #05, GO BACK   ; IF NOT, GO BACK FOR INPUT
890 002644 052737 100000 001504  BIS      #BIT15, DZLVO   ; IF SO, SET FLAG
891 002652 000752              BR       #05, GET MORE  ; GET MORE INPUT
892 002654              81$:
893
894              ;GET THE MODE OF OPERATION (E,I,S)
895
896 002654 104403      INSTR    ;CALL THE STRING INPUT ROUTINE
897 002656 003652      72$     ;POINTER TO THE MESSAGE TO BE PRINTED
898 002660 104406      SETFLG   ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
899 002662 001512      MANTO    ;THIS IS THE FLAG BEING SETUP
900
901              ;GET THE NUMBER OF DZ11'S RUNNING
902
903 002664 104403      INSTR    ;CALL THE STRING INPUT ROUTINE
904 002666 003610      71$     ;POINTER TO MESSAGE TO BE PRINTED
905 002670 104405      PARAM   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
906 002672 000001      1       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
907 002674 000020      16      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
908 002676 001220      $TMP1   ;POINTER TO MAP LOCATION TO BE FILLED
909 002700      000    ;MASK OF INVALID BITS FOR THIS PARAMETER
910 002701      001    ;NUMBER OF PARAMETERS TO STORE
911
912 002702 012737 000377 001506  MOV      #377, LINEO     ;SET UP DEFAULT LINES
913 002710 012737 017470 001510  MOV      #17470, PARO    ;SET UP DEFAULT LPR PARAMETER
914
915 002716 012737 000001 006722  MOV      #1, DLYCNT      ;RECEIVER ON: 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
916 002724 032777 000010 176226  BIT      #SW03, @SWR     ;INITIALIZE DELAY COUNT
917 002732 001402              BEQ      #40$, DO YOU   ;DO YOU WANT PARAMETERS?
918 002734 004737 003144              JSR      PC, 23$        ;IF NO, SKIP THE PARAMETER CALL
919 002740 012737 000001 001312 40$:   MOV      #1, $DEVN      ;GET PARAMETERS
920 002746 113737 001220 001410      MOVVB   $TMP1, DZNUM    ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
921 002754 113737 001220 001411      MOVVB   $TMP1, SAVNUM   ;COPY THE NUMBER OF DEVICES
922 002762 005337 001220      62$:   DEC      $TMP1          ;COPY A BACKUP NUMBER
923 002766 001404              BEQ      #61$, $TMP1    ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
924 002770 000261              SEC                      ;SELECTED DEVICES
925 002772 006137 001312      ROL     $DEVN          ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
926 002776 000771              BR       #62$, POINT   ;POINT TO THE NEXT DEVICE
927 003000 013737 001312 001222 61$:   MOV      $DEVN, $TMP2   ;GO DO THIS PROCEDURE AGAIN
928 003006 013737 001312 001404      MOV     $DEVN, DZACTV   ;# OF TIMES
929 003014 012700 001500      MOV     #DZCR0, R0     ;COPY THE ACTIVE DEVICE PARAMETER
930 003020 012701 001514      MOV     #DZCR1, R1     ;SET A POINTER TO THE SPECIFIED INFORMATION
931 003024 012702 001320      MOV     #SDDW0, R2     ;POINT R1 TO THE REST OF THE MAP TABLE
932 003030 000241              CLC                      ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
933 003032 006037 001222      ROR     $TMP2          ;INITIALIZE THE "C" BIT FOR A ROTATION
934 003036 006237 001222      64$:   ASR     $TMP2          ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
935 003042 103404              BCS     #41$, ISOLATE  ;ISOLATE A SELECTION FLAG IN THE "C" BIT
936 003044 012711 177777      MOV     #-1, (R1)      ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
937 003050 000137 004244      JMP     #63$, TERMIN   ;TERMINATE THE LIST
                       ;GO TO THE NEXT BLOCK

```

```

938 003054 012011          41$:  MOV      (R0)+,(R1)      ;ADDRESS
939 003056 062721 000010  ADD      #10,(R1)+      ;POINT TO THE NEXT DZ11 ADDRESS VALUE
940 003062 012011          MOV      (R0)+,(R1)      ;VECTOR
941 003064 062721 000010  ADD      #10,(R1)+      ;POINT TO THE NEXT VECTOR VALUE
942 003070 012021          MOV      (R0)+,(R1)+     ;LEVEL
943 003072 012021          MOV      (R0)+,(R1)+     ;LINES
944 003074 016012 177774  MOV      -4(R0),(R2)     ;GET THE EIA FLAG FROM THE PRIORITY WORD
945 003100 042712 077777  BIC      #77777,(R2)     ;ISOLATE THAT FLAG
946 003104 051022          BIS      (R0),(R2)+     ;ADD PARAMETERS TO DEVICE DESCRIPTOR WORD
947 003106 012021          MOV      (R0)+,(R1)+     ;PARAMETERS
948 003110 012021          MOV      (R0)+,(R1)+     ;MAINTENANCE MODE
949 003112 000751          BR       64$
950 003114 032777 000010 176036 73$:  BIT      #SW03,DSWR      ;ASK PARAMETERS ?
951 003122 001002          BNE      42$            ;IF NO, GO DO AUTO SIZING
952 003124 000137 004244  JMP      63$            ;GO SET UP FOR AUTO SIZING
953 003130 004737 003144 42$:  JSR      PC,23$         ;GO ASK PARAMETERS
954 003134 105337 001415  DECB     INIFLG         ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
955 003140 000137 004270  JMP      16$            ;GO TO THE NEXT BLOCK
956
957
958
959
960 003144          23$:  INSTR
961 003146 104403          69$
962 003148 003464          PARAM
963 003150 104400          1
964 003152 000001          377
965 003154 000377          LINED
966 003156 001506          .BYTE   0
967 003160 000          .BYTE   1
968 003161 001          CLR     HDRFLG
969 003162 105037 001416
970
971
972
973
974 003166 005737 001512  TST      MANTO          ;IS STAGGERED THE MODE OF OPERATION?
975 003172 100021          BPL      26$            ;IF NOT, SKIP THIS SEGMENT
976 003174 013703 001506  MOV      LINED,R3       ;GET A SCRATCH COPY OF THE ACTIVE LINES
977 003200 006003          24$:  ROR      R3             ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
978 003202 103410          BCS      25$            ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
979 003204 001414          BEQ      26$            ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
980 003206 006203          ASR      R3             ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
981 003210 103373          BCC      24$            ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
982 003212 104402 001230  TYPE     ,SQUES         ;THIS IS AN INCORRECT PARAMETER
983 003216 104402 010424  TYPE     ,MBADLN        ;LET THE USER KNOW ABOUT IT
984 003222 000750          BR       23$            ;GO GET THE CORRECT PARAMETER
985 003224 001772          BEQ      27$            ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
986 003226 006203          ASR      R3             ;GET THE NEXT FLAG
987 003230 103370          BCC      27$            ;IF IT ISN'T SET, THERE'S AN ERROR
988 003232 000241          CLC
989 003234 000761          BR       24$            ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
990
991
992
993 003236          ;GET THE LINE PARAMETER REGISTER ARGUMENT
994 003236 104403          26$:  INSTR
995
996
997
998
999
1000
    ;CALL THE STRING INPUT ROUTINE
    ;POINTER TO MESSAGE TO BE PRINTED
    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
    ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
    ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
    ;POINTER TO MAP LOCATION TO BE FILLED
    ;MASK OF INVALID BITS FOR THIS PARAMETER
    ;NUMBER OF PARAMETERS TO STORE
    ;MAKE SURE THE CHANGES ARE PRINTED
    ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
    ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
    ;IS STAGGERED THE MODE OF OPERATION?
    ;IF NOT, SKIP THIS SEGMENT
    ;GET A SCRATCH COPY OF THE ACTIVE LINES
    ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
    ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
    ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
    ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
    ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
    ;THIS IS AN INCORRECT PARAMETER
    ;LET THE USER KNOW ABOUT IT
    ;GO GET THE CORRECT PARAMETER
    ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
    ;GET THE NEXT FLAG
    ;IF IT ISN'T SET, THERE'S AN ERROR
    ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
    ;GO TEST THE NEXT PAIR OF FLAGS
    ;CALL THE STRING INPUT ROUTINE
    
```

994	003240	003540			70\$: POINTER TO MESSAGE TO BE PRINTED
995	003242	104405			PARAM		: CALL THE OCTAL TO ASCII CONVERT ROUTINE
996	003244	000000			0		: LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
997	003246	000017			17		: HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
998	003250	001510			PARQ		: POINTER TO MAP LOCATION TO BE FILLED
999	003252	000			.BYTE 0		: MASK OF INVALID BITS FOR THIS PARAMETER
1000	003253	001			.BYTE 1		: NUMBER OF PARAMETERS TO STORE
1001	003254	012702	001506		MOV #LINEO,R2		: POINT TO THE LINE SELECTION PARAMETER
1002	003260	012703	001510		MOV #PARQ,R3		: POINT TO THE CHOSEN PARAMETERS
1003	003264	011304			MOV (R3),R4		: USE BAUD RATE AS AN INDEX IN DELAY TABLE
1004	003266	006304			ASL R4		: ALIGN INDEX ON WORD BOUNDARY
1005	003270	016437	031050	006722	MOV DLYTBL(R4),DLYCNT		: SET THE DELAY COUNT FOR THIS BAUD RATE
1006	003276	000313			SWAB (R3)		: PLACE IN HIGH BYTE
1007	003300	052713	010070		BIS #10070,(R3)		: PLACE EXTRA PARAMETERS INTO LOC
1008	003304	011262	000014	28\$:	MOV (R2),14(R2)		: LOAD THE LINES
1009	003310	011363	000014		MOV (R3),14(R3)		: LOAD THE PARAMETERS
1010	003314	062702	000014		ADD #14,R2		: POINT TO THE NEXT SET
1011	003320	062703	000014		ADD #14,R3		: OF BOTH PARAMETERS
1012	003324	020327	001774		CMP R3,#PAR17		: HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
1013	003330	001365			BNE 28\$: IF NOT, GO LOAD SOME MORE PARAMETERS
1014	003332	000207			RTS PC		: RETURN TO CALLING BLOCK
1015	003334	030600	052123	041440	66\$:	.ASCIZ <200>/1ST CSR ADDRESS (160000:163700): /	
(1)	003400	030600	052123	053040	67\$:	.ASCIZ <200>/1ST VECTOR ADDRESS (300:770): /	
(1)	003441	200	051102	046040	68\$:	.ASCIZ <200>/BR LEVEL (4:6): /	
(1)	003464	046200	047111	051505	69\$:	.ASCIZ <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:377): /	
(1)	003540	042200	043105	052501	70\$:	.ASCIZ <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17): /	
(1)	003610	021600	047440	020106	71\$:	.ASCIZ <200>/# OF DZ11'S <IN OCTAL> (1:20): /	
(1)	003652	046600	044501	052116	72\$:	.ASCIZ <200>/MAINTENANCE MODE/	
(1)	003673	200	055440	054105	.ASCII <200>/ [EXTERNAL <H325>-EIA ONLY (E)]/		
(1)	003741	200	055440	047111	.ASCII <200>/ [INTERNAL <DZCSR03=1> (I)]/		
(1)	004007	200	055440	052123	.ASCII <200>/ [STAGGERED <H3271>-EIA ONLY (S)]:/		
(1)	004057	200	055440	052123	.ASCII <200>/ [STAGGERED <H3190>-20MA ONLY (S)]:/		
(1)	004130	052200	050131	020105	74\$:	.ASCIZ <200>/TYPE "A" FOR EIA MODULE OR "B" FOR 20 MA (A:B): /	
(1)	004212	042600	052116	051105	75\$:	.ASCIZ <200>/ENTER DELAY PARAMETER: /	
(1)	004244	004244			63\$:	.EVEN	
1016	004244	122737	000377	001415	CMPB #377,INIFLG		: ONLY DO AUTO SIZE ON 1ST START
1017	004252	001006			BNE 16\$		
1018	004254	032777	000200	174676	BIT #BIT7,@SWR		: BIT7=1??
1019	004262	001002			BNE 16\$: BR IF NO AUTO SIZE
1020	004264	004737	011612		JSR PC,AUTO.SIZE		: GO DO THE AUTO SIZE
1021	004270	105737	001416	16\$:	TSTB HDRFLG		: HAS THE TABLE BEEN TYPED YET?
1022	004274	001021			BNE 1\$: IF SO, DON'T TYPE IT AGAIN
1023	004276	105337	001416		DECB HDRFLG		: INDICATE THAT THE TABLE WILL BE TYPED
1024	004302	104402	010377		TYPE XHEAD		: TYPE MAP HEADER
1025	004306	012700	001500		MOV #DZ.MAP,R0		: SET POINTER
1026	004312	010037	001220	5\$:	MOV R0,\$TMP1		: POINT TO THE MAP LOCATION
1027	004316	012037	001222		MOV (R0)+,\$TMP2		: SET DATA
1028	004322	022737	177777	001222	CMP #-1,\$TMP2		: END OF LIST?
1029	004330	001403			BEQ 1\$: BR IF YES
1030	004332	104411		17\$:	CONVRT		: CALL THE OCTAL TO ASCII CONVERSION ROUTINE
1031	004334	010466			XSTATQ		: CONVERT THE DATA AT THIS ADDRESS
1032	004336	000765			BR 5\$: GO PRINT THE NEXT PARAMETER
1033	004340	005737	000042	1\$:	TST @#42		: IS PROGRAM RUNNING UNDER MONITOR
1034	004344	001026			BNE 3\$: YES
1035	004346	032777	000100	174604	BIT #SW06,@SWR		: DESELECT SPECIFIC DEVICES??

1036	004354	001422				BEG	3\$:BR IF NO.
1037	004356	104402	010320			TYPE	MNEW		:TYPE THE MESSAGE.
1038	004362	005000				CLR	RO		:ZERO DATA DISPLAY
1039	004364	000000				HALT			:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1040	004366	027737	174566	001312		CMP	2\$SWR,\$DEV		:IS THE NUMBER VALID?
1041	004374	101404				BLOS	2\$:BR IF NUMBER IS OK.
1042	004376	104402	010172			TYPE	,MERR3		:TELL USER OF INVALID NUMBER.
1043	004402	000000			9\$:	HALT			:STOP EVERY THING.
1044	004404	000776				BR	9\$:RESTART THE PROGRAM AGAIN.
1045	004406	017737	174546	001404	2\$:	MOV	2\$SWR,DZACTV		:GET NEW DEVICE PATTERN
1046	004414	013700	001404			MOV	DZACTV,RO		:SHOW THE USER WHAT HE SELECTED.
1047	004420	000000				HALT			:CONTINUE DYNAMIC SWITCHES.
1048	004422	032777	000020	174530	3\$:	BIT	#SW04,2\$SWR		:CHECK TO SEE IF DELAY COUNT CHANGES
1049	004430	001407				BEG	18\$:IF NOT, GO CLEAR VECTOR AREA
1050	004432	104403				INSTR			:CALL THE STRING INPUT ROUTINE
1051	004434	004212				75\$:POINTER TO MESSAGE TO BE PRINTED
1052	004436	104405				PARAM			:CALL THE OCTAL TO ASCII CONVERT ROUTINE
1053	004440	000001				1			:LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1054	004442	177777				177777			:HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1055	004444	006722				DLYCNT			:POINTER TO MAP LOCATION TO BE FILLED
1056	004446	000				.BYTE	0		:MASK OF INVALID BITS FOR THIS PARAMETER
1057	004447	001				.BYTE	1		:NUMBER OF PARAMETERS TO STORE
1058	004450	012700	000300		18\$:	MOV	#300,RO		:PREPARE TO CLEAR THE FLOATING
1059	004454	012701	000302			MOV	#302,R1		:VECTOR AREA. 300-776
1060	004460	010120			4\$:	MOV	R1,(R0)+		:START PUTTING "PC+2 - HALT"
1061	004462	005021				CLR	(R1)+		:IN VECTOR AREA.
1062	004464	022021				CMP	(R0)+,(R1)+		:POP POINTERS
1063	004466	022700	001000			CMP	#1000,RO		:ALL DONE??
1064	004472	001372				BNE	4\$:BR IF NO.
1065									
1066									
1067									
1068									
1069	004474	012706	001120			.BEGIN:	MOV	#STACK,SP	:SET UP STACK
1070	004500	106427	000340			MTPS	#PR7		:LOCK OUT INTERRUPTS
1071	004504	005737	000042			TST	2\$42		:IS PROGRAM UNDER MONITOR CONTROL
1072	004510	001015				BNE	2\$:BR IF YES
1073	004512	032777	000004	174440		BIT	#BIT2,2\$SWR		:CHECK FOR LOCK ON TEST
1074	004520	001406				BEG	1\$:BR IF NO LOCK DESIRED.
1075	004522	104402	010216			TYPE	MLOCK		:TYPE LOCK SELECTED.
1076	004526	012737	000240	005010		MOV	#NOP,TTST		:ADJUST SCOPE ROUTINE.
1077	004534	000403				BR	2\$:CONTINUE ALONG.
1078	004536	013737	005232	005010	1\$:	MOV	BRW,TTST		:PREPARE NORMAL SCOPE ROUTINE
1079	004544	012737	011070	001126	2\$:	MOV	#CYCLE,\$LPADR		:START AT "CYCLE" FIND WHICH DEVICE TO TEST
1080	004552	104402	010107			TYPE	MR		:TYPE "RUNNING"
1081	004556	000177	174344			JMP	2\$LPADR		:START TESTING

:TEST START AND RESTART

```

1082                                     ;END OF PASS
1083                                     ;TYPE NAME OF TEST
1084                                     ;UPDATE PASS COUNT
1085                                     ;CHECK FOR EXIT TO ACT-11
1086                                     ;RESTART TEST
1087 .SBTTL END OF PASS ROUTINE
1088
1089                                     ;*****
1090                                     ;#INCREMENT THE PASS NUMBER ($PASS)
1091                                     ;*IF THERES A MONITOR GO TO IT
1092                                     ;*IF THERE ISN'T JUMP TO CYCLE
1093
1094 004562                                SEOP:
1095 004562 000004                        SCOPE
1096 004564 005037 001136                CLR $ERRPC ;CLEAR LAST ERROR PC
1097 004570 105037 001123                CLR $ERFLG ;CLEAR ERROR FLAG
1098 004574 104402 010063                TYPE ,MEPASS ;TYPE END PASS
1099 004600 104402 010245                TYPE ,MCSRX ;TYPE CSR
1100 004604 104412 004742                CNVRT ,XCSR ;SHOW IT
1101 004610 104402 010253                TYPE ,MVECX ;TYPE VECTOR
1102 004614 104412 004750                CNVRT ,XVEC ;SHOW IT
1103 004620 005237 001242                INC $PASS ;RAISE PASS COUNT
1104 004624 104402 010261                TYPE ,MPASSX ;TYPE PASSES
1105 004630 104412 004756                CNVRT ,XPASS ;SHOW IT
1106 004634 005337 001242                DEC $PASS ;RESTORE PASS COUNT
1107 004640 104402 010272                TYPE ,MERRX ;TYPE ERRORS
1108 004644 104412 004764                CNVRT ,XERR ;SHOW IT
1109 004650 105337 001411                DECB $AVNUM ;ARE ALL DEVICES TESTED?
1110 004654 001030                       BNE $DOAGN ;BR IF NO.
1111 004656 113737 001410 001411        MOV $DZNUM,$AVNUM ;RESTORE THE COUNT
1112 004664 005037 001226                CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
1113 004670 005237 001242                INC $PASS ;INCREMENT THE PASS NUMBER
1114 004674 042737 100000 001242        BIC $100000,$PASS ;DON'T ALLOW A NEG. NUMBER
1115 004702 005327                       DEC (PC)+ ;LOOP?
1116 004704 000001                        SEOPCT: .WORD 1
1117 004706 003013                       BGT $DOAGN ;YES
1118 004710 012737                       MOV (PC)+,$(PC)+ ;RESTORE COUNTER
1119 004712 000001                        SENDCT: .WORD 1
1120 004714 004704                        SEOPCT
1121 004716 013700 000042                $GET42: MOV $42,R0 ;GET MONITOR ADDRESS
1122 004722 001405                       BEQ $DOAGN ;BRANCH IF NO MONITOR
1123 004724 000005                       RESET ;CLEAR THE WORLD
1124 004726 004710                       SENDAD: JSR PC,(R0) ;GO TO MONITOR
1125 004730 000240                       NOP ;SAVE ROOM
1126 004732 000240                       NOP ;FOR
1127 004734 000240                       NOP ;ACT11
1128 004736                                SDOAGN:
1129 004736 000137                       JMP $$(PC)+ ;RETURN
1130 004740 011070                       SRTNAD: .WORD CYCLE
1131
1132 004742 000001                        XCSR: 1
1133 004744 006 002                       .BYTE 6,2
1134 004746 002042                       DZCSR
1135 004750 000001                        XVEC: 1
1136 004752 003 002                       .BYTE 3,2
1137 004754 002072                       DZRIV

```

```

1138 004756 000001          XPASS: 1
1139 004760          006      002      .BYTE 6,2
1140 004762 001242          $PASS
1141 004764 000001          XERR: 1
1142 004766          006      002      .BYTE 6,2
1143 004770 001132          $ERTTL
1144
1145          ;SCOPE LOOP AND ITERATION HANDLER
1146          ;-----
1147
1148          .SBTTL SCOPE HANDLER ROUTINE
1149
1150          ;*****
1151          ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1152          ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1153          ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1154          ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1155          ;$SW14=1 LOOP ON TEST
1156          ;$SW11=1 INHIBIT ITERATIONS
1157          ;$CALL
1158          ; SCOPE          ;;SCOPE=IOT
1159
1160 004772          $$SCOPE:
1161 004772 004737 007360          .SCOPE: JSR PC SERV.G          ;FIND OUT IF <IG> WAS HIT
1162 004776 005037 001136          CLR $ERRPC          ;CLEAR LAST ERROR PC.
1163 005002 022716 012376          CMP #TST1+2,(SP)    ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
1164 005006 001413          BEQ $XTSTR          ;IF SO, DON'T LOOP ON IT
1165 005010 000406          BR 1$              ;GOTO 1$ (IF LOCK SW02=1; THIS LOC =240)
1166 005012 105777 174146          TSTB #STKS          ;KEYBOARD DONE?
1167 005016 100067          BPL $OVER          ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
1168 005020 017766 174142 177776          MOV #STKB,-2(SP)    ;CLEAR DONE BIT
1169 005026 032777 040000 174124          1$: BIT #BIT14,$SWR  ;LOOP ON PRESENT TEST?
1170 005034 001060          BNE $OVER          ;YES IF SW14=1
1171
1172 005036 000416          ;*****START OF CODE FOR THE XOR TESTER*****
1173          $XTSTR: BR 6$
1174 005040 013746 000004          MOV #ERRVEC,-(SP)   ;IF RUNNING ON THE "XOR" TESTER CHANGE
1175 005044 012737 005064 000004          MOV #SS,#ERRVEC    ;THIS INSTRUCTION TO A "NOP" (NOP=240)
1176 005052 005737 177060          TST #177060        ;SAVE THE CONTENTS OF THE ERROR VECTOR
1177 005056 012637 000004          MOV (SP)+,#ERRVEC  ;SET FOR TIMEOUT
1178 005062 000436          BR $SVLAD          ;TIME OUT ON XOR?
1179 005064 022626          SS: CMP (SP)+,(SP)+ ;RESTORE THE ERROR VECTOR
1180 005066 012637 000004          MOV (SP)+,#ERRVEC  ;GO TO THE NEXT TEST
1181 005072 000441          BR $OVER          ;CLEAR THE STACK AFTER A TIME OUT
1182 005074          6$;*****END OF CODE FOR THE XOR TESTER*****
1183 005074 105737 001123          2$: TSTB $ERFLG     ;HAS AN ERROR OCCURRED?
1184 005100 001404          BEQ 3$            ;BR IF NO
1185 005102 105037 001123          4$: CLRB $ERFLG   ;ZERO THE ERROR FLAG
1186 005106 005037 001226          CLR $TIMES        ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1187 005112 032777 004000 174040          3$: BIT #BIT11,$SWR ;INHIBIT ITERATIONS?
1188 005120 001011          BNE 1$            ;BR IF YES
1189 005122 005737 001242          TST $PASS         ;IF FIRST PASS OF PROGRAM
1190 005126 001406          BEQ 1$            ;INHIBIT ITERATIONS
1191 005130 005237 001124          INC $ICNT         ;INCREMENT ITERATION COUNT
1192 005134 023737 001226 001124          CMP $TIMES,$ICNT  ;CHECK THE NUMBER OF ITERATIONS MADE
1193 005142 002015          BGE $OVER         ;BR IF MORE ITERATION REQUIRED

```

```

1194 005144 012737 000001 001124 1$: MOV #1,$ICNT ;:REINITIALIZE THE ITERATION COUNTER
1195 005152 013737 005234 001226 MOV $MXCNT,$TIMES ;:SET NUMBER OF ITERATIONS TO DO
1196 005160 105237 001122 $SVLAD: INCB $STNM ;:COUNT TEST NUMBERS
1197 005164 113737 001122 001240 MOV $STNM,$TESTN ;:SET TEST NUMBER IN APT MAILBOX
1198 005172 011637 001126 MOV (SP),$LPADR ;:SAVE SCOPE LOOP ADDRESS
1199 005176 013777 001122 173756 $OVER: MOV $STNM,$DISPLAY ;:DISPLAY TEST NUMBER
1200 005204 013716 001126 MOV $LPADR,(SP) ;:FUDGE RETURN ADDRESS
1201 005210 105037 001417 3$: CLRB MNTFLG ;:CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
1202 005214 005737 001370 TST MODE ;:HAS THE MODE BEEN CHANGED?
1203 005220 001003 BNE 4$ ;:IF NOT INTERNAL, GO DO A TEST
1204 005222 112737 000010 001417 MOVB #MAINT,MNTFLG ;:IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
1205 005230 000002 4$: RTI ;:GO DO THE TEST
1206 005232 000406 BRW: 406
1207 005234 000005 $MXCNT: 5 ;:MAX. NUMBER OF ITERATIONS
1208
1209 ;:CHECK FOR FREEZE ON CURRENT DATA
1210 ;-----
1211
1212 005236 032777 001000 173714 .SCOPI: BIT #SW09,$SWR ;:IS SW09=1(SET)?
1213 005244 001405 1$ BEQ 1$ ;:BR IF NOT SET.
1214 005246 005737 001362 TST LOCK ;:IS THER A TIGHT LOOP SPECIFIED?
1215 005252 001402 BEQ 1$ ;:IF NO, RETURN
1216 005254 013716 001362 MOV LOCK,(SP) ;:IF YES, GOTO THE ADDRESS IN LOCK.
1217 005260 000002 1$: RTI ;:GO BACK.
1218
1219 005262 032777 010000 173670 .TYPE: BIT #SW12,$SWR ;:INHIBIT ALL PRINTOUT??
1220 005270 001403 1$ BEQ 1$ ;:IF NOT, GO TYPE
1221 005272 062716 000002 ADD #2,(SP) ;:SKIP OVER MESSAGE POINTER
1222 005276 000002 1$: RTI ;:RETURN TO WHERE PROCEDURE WAS INVOKED
1223 005300
1224 .SBTTL TYPE ROUTINE
1225
1226 ;:*****
1227 ;:ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1228 ;:THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1229 ;:NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1230 ;:NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1231 ;:NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1232 ;:
1233 ;:CALL:
1234 ;:1) USING A TRAP INSTRUCTION
1235 ;: TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1236 ;:OR
1237 ;: TYPE
1238 ;: MESADR
1239 ;:
1240
1241 005300 105737 001177 $TYPE: TSTB $TPFLG ;:IS THERE A TERMINAL?
1242 005304 100002 1$ BPL 1$ ;:BR IF YES
1243 005306 000000 HALT ;:HALT HERE IF NO TERMINAL
1244 005310 000430 BR 3$ ;:LEAVE
1245 005312 010046 1$: MOV RO,-(SP) ;:SAVE RO
1246 005314 017600 000002 MOV @2(SP),RO ;:GET ADDRESS OF ASCIZ STRING
1247 005320 122737 000001 001254 CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
1248 005326 001011 BNE 62$ ;:NO GO CHECK FOR APT CONSOLE
1249 005330 132737 000100 001255 BITB #APTPOOL,$ENVM ;:SPOOL MESSAGE TO APT

```



```

1250 005336 001405          BEQ      62$      ; NO GO CHECK FOR CONSOLE
1251 005340 010037 005350    MOV      RD,61$  ; SETUP MESSAGE ADDRESS FOR APT
1252 005344 004737 005570    JSR     PC,$ATY3 ; SPOOL MESSAGE TO APT
1253 005350 000000          .WORD   0        ; MESSAGE ADDRESS
1254 005352 132737 000040 001255 61$:  BITB    #APTCSUP,$ENVM ; APT CONSOLE SUPPRESSED
1255 005360 001003          BNE     60$      ; YES, SKIP TYPE OUT
1256 005362 112046          2$:    MOVB   (RD)+,-(SP) ; PUSH CHARACTER TO BE TYPED ONTO STACK
1257 005364 001005          BNE     4$       ; BR IF IT ISN'T THE TERMINATOR
1258 005366 005726          TST     (SP)+    ; IF TERMINATOR POP IT OFF THE STACK
1259 005370 012600          60$:   MOV     (SP)+,RD  ; RESTORE RD
1260 005372 062716 000002    3$:    ADD     #2,(SP) ; ADJUST RETURN PC
1261 005376 000002          RTI                    ; RETURN
1262 005400 122716 000011    4$:    CMPB   #HT,(SP) ; BRANCH IF <HT>
1263 005404 001430          BEQ     8$       ;
1264 005406 122716 000200    CMPB   #CRLF,(SP) ; ; BRANCH IF NOT <CRLF>
1265 005412 001006          BNE     5$       ;
1266 005414 005726          TST     (SP)+    ; ; POP <CR><LF> EQUIV
1267 005416 104402          TYPE   ; ; TYPE A CR AND LF
1268 005420 001231          $CRLF ;
1269 005422 105037 005556    CLRB   $CHARCNT ; ; CLEAR CHARACTER COUNT
1270 005426 000755          BR      2$       ; ; GET NEXT CHARACTER
1271 005430 004737 005512    5$:    JSR     PC,$TYPEC ; ; GO TYPE THIS CHARACTER
1272 005434 123726 001176    6$:    CMPB   $FILLC,(SP)+ ; ; IS IT TIME FOR FILLER CHARS.?
1273 005440 001350          BNE     2$       ; ; IF NO GO GET NEXT CHAR.
1274 005442 013746 001174    MOV     $NULL,-(SP) ; ; GET # OF FILLER CHARS. NEEDED
1275                                ; ; AND THE NULL CHAR.
1276 005446 105366 000001    7$:    DECB   1(SP)   ; ; DOES A NULL NEED TO BE TYPED?
1277 005452 002770          BLT     6$       ; ; BR IF NO--GO POP THE NULL OFF OF STACK
1278 005454 004737 005512    JSR     PC,$TYPEC ; ; GO TYPE A NULL
1279 005460 105337 005556    DECB   $CHARCNT ; ; DO NOT COUNT AS A COUNT
1280 005464 000770          BR      7$       ; ; LOOP
1281
1282                                ; HORIZONTAL TAB PROCESSOR
1283
1284 005466 112716 000040    8$:    MOVB   #' ,(SP) ; ; REPLACE TAB WITH SPACE
1285 005472 004737 005512    9$:    JSR     PC,$TYPEC ; ; TYPE A SPACE
1286 005476 132737 000007 005556    BITB   #7,$CHARCNT ; ; BRANCH IF NOT AT
1287 005504 001372          BNE     9$       ; ; TAB STOP
1288 005506 005726          TST     (SP)+    ; ; POP SPACE OFF STACK
1289 005510 000724          BR      2$       ; ; GET NEXT CHARACTER
1290 005512 105777 173452    $TYPEC: TSTB   @STPS ; ; WAIT UNTIL PRINTER IS READY
1291 005516 100375          BPL     $TYPEC  ; ;
1292 005520 116677 000002 173444    MOVB   2(SP),@STPB ; ; LOAD CHAR TO BE TYPED INTO DATA REG.
1293 005526 122766 000015 000002    CMPB   #CR,2(SP) ; ; IS CHARACTER A CARRIAGE RETURN?
1294 005534 001003          BNE     1$       ; ; BRANCH IF NO
1295 005536 105037 005556    CLRB   $CHARCNT ; ; YES--CLEAR CHARACTER COUNT
1296 005542 000406          BR      $TYPEX  ; ; EXIT
1297 005544 122766 000012 000002 1$:    CMPB   #LF,2(SP) ; ; IS CHARACTER A LINE FEED?
1298 005552 001402          BEQ     $TYPEX  ; ; BRANCH IF YES
1299 005554 105227          INCB   (PC)+    ; ; COUNT THE CHARACTER
1300 005556 000000          $CHARCNT: .WORD 0 ; ; CHARACTER COUNT STORAGE
1301 005560 000207          $TYPEX: RTS     PC
1302
1303                                ; .SBTTL APT COMMUNICATIONS ROUTINE
1304
1305                                ; ; *****

```

```

1306 005562 112737 000001 006026 $ATY1: MOVB #1,$FFLG ;; TO REPORT FATAL ERROR
1307 005570 112737 000001 006024 $ATY3: MOVB #1,$MFLG ;; TO TYPE A MESSAGE
1308 005576 000403          $ATYC
1309 005600 112737 000001 006026 $ATY4: MOVB #1,$FFLG ;; TO ONLY REPORT FATAL ERROR
1310 005606          $ATYC:
1311 005606 010046          MOV RD,-(SP) ;; PUSH RD ON STACK
1312 005610 010146          MOV R1,-(SP) ;; PUSH R1 ON STACK
1313 005612 105737 006024          TSTB $MFLG ;; SHOULD TYPE A MESSAGE?
1314 005616 001450          BEQ 5$ ;; IF NOT: BR
1315 005620 122737 000001 001254          CMPB #APTENV,$ENV ;; OPERATING UNDER APT?
1316 005626 001031          BNE 3$ ;; IF NOT: BR
1317 005630 132737 000100 001255          BITB #APTPOOL,$ENVM ;; SHOULD SPOOL MESSAGES?
1318 005636 001425          BEQ 3$ ;; IF NOT: BR
1319 005640 017600 000004          MOV #4(SP),RO ;; GET MESSAGE ADDR.
1320 005644 062766 000002 000004          ADD #2,4(SP) ;; BUMP RETURN ADDR.
1321 005652 005737 001234 1$: TST $MSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
1322 005656 001375          BNE 1$ ;; IF NOT: WAIT
1323 005660 010037 001250          MOV RO,$MSGAD ;; PUT ADDR IN MAILBOX
1324 005664 105720          2$: TSTB (RO)+ ;; FIND END OF MESSAGE
1325 005666 001376          BNE 2$
1326 005670 163700 001250          SUB $MSGAD,RO ;; SUB START OF MESSAGE
1327 005674 006200          ASR RO ;; GET MESSAGE LNTH IN WORDS
1328 005676 010037 001252          MOV RO,$MSGLGT ;; PUT LENGTH IN MAILBOX
1329 005702 012737 000004 001234          MOV #4,$MSGTYPE ;; TELL APT TO TAKE MSG.
1330 005710 000413          BR 5$
1331 005712 017637 000004 005736 3$: MOV #4(SP),4$ ;; PUT MSG ADDR IN JSR LINKAGE
1332 005720 062766 000002 000004          ADD #2,4(SP) ;; BUMP RETURN ADDRESS
1333 005726 013746 177776          MOV 177776,-(SP) ;; PUSH 177776 ON STACK
1334 005732 004737 005300          JSR PC,$TYPE ;; CALL TYPE MACRO
1335 005736 000000          4$: .WORD 0
1336 005740          5$:
1337 005740 105737 006026          10$: TSTB $FFLG ;; SHOULD REPORT FATAL ERROR?
1338 005744 001416          BEQ 12$ ;; IF NOT: BR
1339 005746 005737 001254          TST $ENV ;; RUNNING UNDER APT?
1340 005752 001413          BEQ 12$ ;; IF NOT: BR
1341 005754 005737 001234          11$: TST $MSGTYPE ;; FINISHED LAST MESSAGE?
1342 005760 001375          BNE 11$ ;; IF NOT: WAIT
1343 005762 017637 000004 001236          MOV #4(SP),$FATAL ;; GET ERROR #
1344 005770 062766 000002 000004          ADD #2,4(SP) ;; BUMP RETURN ADDR.
1345 005776 005237 001234          INC $MSGTYPE ;; TELL APT TO TAKE ERROR
1346 006002 105037 006026          12$: CLRB $FFLG ;; CLEAR FATAL FLAG
1347 006006 105037 006025          CLRB $LFLG ;; CLEAR LOG FLAG
1348 006012 105037 006024          CLRB $MFLG ;; CLEAR MESSAGE FLAG
1349 006016 012601          MOV (SP)+,R1 ;; POP STACK INTO R1
1350 006020 012600          MOV (SP)+,RO ;; POP STACK INTO RO
1351 006022 000207          RTS PC ;; RETURN
1352 006024 000          $MFLG: .BYTE 0 ;; MESSG. FLAG
1353 006025 000          $LFLG: .BYTE 0 ;; LOG FLAG
1354 006026 000          $FFLG: .BYTE 0 ;; FATAL FLAG
1355 006030          .EVEN
1356 000200          APTSIZE=200
1357 000001          APTENV=001
1358 000100          APTPOOL=100
1359 000040          APTCSUP=040
1360
1361 ;STRING INPUT ROUTINE
    
```

```

1362
1363
1364 006030 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON STACK
1365 006032 010446 MOV R4,-(SP) ;SAVE R4 ON STACK
1366 006034 017637 000004 006052 MOV 4(SP),MSG ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
1367 006042 062766 000002 000004 ADD #2,4(SP) ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
1368 006050 104402 .INST1: TYPE ;PRINT THE MESSAGE
1369 006052 000000 .MSG: 0 ;MESSAGE IS POINTED TO FROM HERE
1370 006054 012704 010620 MOV #INBUF,R4 ;POINT R4 TO THE INPUT BUFFER
1371 006060 012703 000007 MOV #7,R3 ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
1372 006064 105777 173074 1$: TSTB 2$TKS ;HAS A CHARACTER BEEN RECEIVED?
1373 006070 100375 BPL 1$ ;IF NO, KEEP WAITING FOR IT
1374 006072 117714 173070 MOVB 2$TKB,(R4) ;IF YES, SAVE IT IN THE INPUT BUFFER
1375 006076 142714 000200 BICB #200,(R4) ;KEEP ONLY THE 7-BIT ASCII INFORMATION
1376 006102 122427 000015 CMPB (R4)+,#15 ;IS THIS CHARACTER A LINE FEED?
1377 006106 001417 BEQ INSTR2 ;IF SO, TERMINATE THE INPUT SEQUENCE
1378 006110 105777 173054 2$: TSTB 2$TPS ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
1379 006114 100375 BPL 2$ ;IF WE CAN'T, WAIT UNTIL WE CAN
1380 006116 017777 173044 173046 MOV 2$TKB,2$TPB ;ECHO THE CHARACTER BACK
1381 006124 005303 DEC R3 ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
1382 006126 001356 BNE 1$ ;IF WE DON'T HAVE 7, GO GET SOME MORE
1383 006130 012604 MOV (SP)+,R4 ;IF WE HAVE 7, RESTORE R4
1384 006132 012603 MOV (SP)+,R3 ;RESTORE R3
1385 006134 010346 .INSTE: MOV R3,-(SP) ;SAVE R3 ON THE STACK
1386 006136 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1387 006140 104402 001230 TYPE ,QUES ;PRINT A QUESTION MARK... WHAT'S GOING ON?
1388 006144 000741 BR .INST1 ;GO PRINT THE MESSAGE AGAIN
1389 006146 012604 INSTR2: MOV (SP)+,R4 ;RESTORE R4
1390 006150 012603 MOV (SP)+,R3 ;RESTORE R3
1391 006152 000002 RTI ;RETURN TO THE MAIN PROCEDURE
1392
1393 ;CONVERT ASCII STRING TO OCTAL
1394
1395
1396 006154 010546 .PARAM: MOV R5,-(SP) ;SAVE R5 ON THE STACK
1397 006156 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1398 006160 016605 000004 MOV 4(SP),R5 ;GET THE SETUP INFORMATION POINTER
1399 006164 012537 006344 MOV (R5)+,LOLIM ;SET THE LOW LIMIT FOR THE INPUT
1400 006170 012537 006346 MOV (R5)+,HILIM ;SET THE HIGH LIMIT FOR THE INPUT
1401 006174 012537 006350 MOV (R5)+,DEVADR ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
1402 006200 112537 006352 MOVB (R5)+,LOBITS ;GET THE MASK OF THE INCORRECT BITS
1403 006204 112537 006353 MOVB (R5)+,ADRCNT ;GET THE COUNT OF ITEMS TO BE STORED
1404 006210 010566 000004 MOV R5,4(SP) ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
1405 006214 005005 PARAM1: CLR R5 ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
1406 006216 012704 010620 MOV #INBUF,R4 ;POINT TO THE INPUT BUFFER
1407 006222 122714 000015 CMPB #15,(R4) ;IS THIS CHARACTER A CARRIAGE RETURN?
1408 006226 001420 BEQ PARERR ;IF SO, PRINT THE MESSAGE AGAIN
1409 006230 121427 000060 1$: CMPB (R4),#60 ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
1410 006234 002415 BLT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1411 006236 121427 000067 CMPB (R4),#67 ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
1412 006242 003012 BGT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1413 006244 142714 000060 BICB #60,(R4) ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
1414 006250 152405 BISB (R4)+,R5 ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
1415 006252 122714 000015 CMPB #15,(R4) ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
1416 006256 001406 BEQ LIMITS ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
1417 006260 006305 ASL R5 ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
    
```

```

1418 006262 006305      ASL      R5      ; CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
1419 006264 006305      ASL      R5      ; MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
1420                                ; NEXT THREE BITS
1421 006266 000760      BR        1$      ; GO GET THE NEXT CHARACTER
1422 006270 104404      PARERR: INSTER    ; THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
1423 006272 000750      BR        PARAM1 ; TRY GETTING THE PARAMETERS AGAIN
1424
1425                                ; TEST TO SEE IF NUMBER IS WITHIN LIMITS
1426                                ;-----
1427
1428 006274 020537 006346  LIMITS: CMP      R5,HILIM ; DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
1429 006300 101373      BHI      PARERR   ; IF YES, GO PRINT THE MESSAGE AGAIN
1430 006302 020537 006344  CMP      R5,LOLIM ; IS THE RESULT LOWER THAN ALLOWED?
1431 006306 103770      BLO      PARERR   ; IF YES, GO PRINT THE MESSAGE AGAIN
1432 006310 133705 006352  BITB    LOBITS,R5 ; ARE ANY INCORRECT BITS SET IN THE RESULT?
1433 006314 001365      BNE      PARERR   ; IF SO, GO PRINT THE MESSAGE AGAIN
1434
1435                                ; STORE NUMBER AT SPECIFIED ADDRESS
1436
1437 006316 013704 006350  1$:     MOV      DEVADR,R4 ; POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
1438 006322 010524      MOV      R5,(R4)+ ; STORE THE RESULT
1439 006324 062705 000002  ADD      #2,R5      ; CALCULATE THE NEXT DATUM
1440 006330 105337 006353  DECB    ADCNT      ; REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
1441 006334 001372      BNE      1$        ; IF NOT, GO STORE THE NEXT DATUM
1442 006336 012604      MOV      (SP)+,R4  ; RESTORE R4
1443 006340 012605      MOV      (SP)+,R5  ; RESTORE R5
1444 006342 000002      RTI                    ; RETURN TO THE MAIN PROGRAM
1445
1446 006344 000000  LOLIM:  0          ; LOWEST ACCEPTABLE VALUE
1447 006346 000000  HILIM:  0          ; HIGHEST ACCEPTABLE
1448 006350 000000  DEVADR: 0          ; LOCATION WHERE RESULT WILL BE STORED
1449 006352 000      LOBITS: .BYTE 0      ; INCORRECT BITS MASK
1450 006353 000      ADCNT:  .BYTE 0      ; COUNT OF ITEMS TO BE STORED
1451
1452                                ; SAVE PC OF TEST THAT FAILED AND R0-R5
1453                                ;-----
1454
1455 006354 016637 000004 001402 .SAV05: MOV      4(SP),SAVPC ; SAVE R7 (PC)
1456
1457                                ; SAVE R0-R5
1458
1459 006362 010537 001214  SV05:  MOV      R5,$REG5 ; SAVE R5
1460 006366 010437 001212  MOV      R4,$REG4 ; SAVE R4
1461 006372 010337 001210  MOV      R3,$REG3 ; SAVE R3
1462 006376 010237 001206  MOV      R2,$REG2 ; SAVE R2
1463 006402 010137 001204  MOV      R1,$REG1 ; SAVE R1
1464 006406 010037 001202  MOV      R0,$REG0 ; SAVE R0
1465 006412 000002      RTI                    ; LEAVE.
1466
1467                                ; RESTORE R0-R5
1468
1469 006414 013700 001202  .RES05: MOV      $REG0,R0 ; RESTORE R0
1470 006420 013701 001204  MOV      $REG1,R1 ; RESTORE R1
1471 006424 013702 001206  MOV      $REG2,R2 ; RESTORE R2
1472 006430 013703 001210  MOV      $REG3,R3 ; RESTORE R3
1473 006434 013704 001212  MOV      $REG4,R4 ; RESTORE R4
    
```

```

1474 006440 013705 001214      MOV      $REG5,R5      ;RESTORE R5
1475 006444 000002      RTI                    ;LEAVE
1476
1477
1478
1479
1480 006446 104402 001231      .CONVR: TYPE          $CRLF          ;PRINT A CARRIAGE RETURN
1481 006452 010046      .CNVRT: MOV           R0,-(SP)      ;SAVE R0
1482 006454 010146      MOV           R1,-(SP)      ;SAVE R1
1483 006456 010346      MOV           R3,-(SP)      ;SAVE R3
1484 006460 010446      MOV           R4,-(SP)      ;SAVE R4
1485 006462 010546      MOV           R5,-(SP)      ;SAVE R5
1486 006464 017601 000012      MOV           @12(SP),R1     ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
1487 006470 062766 000002 000012      ADD          #2,12(SP)      ;POINT TO WHERE MAIN PROGRAM WILL RESUME
1488 006476 012137 006622      MOV           (R1)+,WRDCNT  ;GET NUMBER OF WORDS TO BE PRINTED
1489 006502 112105      1$: MOVVB        (R1)+,R5     ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
1490 006504 112100      MOVVB        (R1)+,R0     ;GET THE NUMBER OF SPACES TO PRINT
1491 006506 013104      MOV           @2(R1)+,R4    ;COPY THE WORD TO BE CONVERTED
1492 006510 110537 006624      MOVVB        R5,CHRCNT    ;COPY THE CHARACTER COUNT
1493 006514 010403      3$: MOV           R4,R3     ;COPY THE ARGUMENT WORD AGAIN
1494 006516 042703 177770      BIC          #1<7>,R3     ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
1495 006522 062703 000060      ADD          #060,R3     ;MAKE AN ASCII CHARACTER OUT OF THEM
1496 006526 110346      MOVVB        R3,-(SP)     ;SAVE THAT CHARACTER
1497 006530 006004      ROR          R4          ;MOVE THE NEXT THREE BITS INTO PLACE
1498 006532 006204      ASR          R4          ;MOVE THEM AGAIN
1499 006534 006204      ASR          R4          ;AND FINALLY A THIRD TIME
1500 006536 005305      DEC          R5          ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
1501
1502 006540 001365      BNE          3$         ;BUILT?
1503 006542 012703 010724      MOV          #MDATA,R3    ;IF NO, GO BUILD THE NEXT ONE.
1504 006546 112623      4$: MOVVB        (SP)+,(R3)+  ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
1505 006550 105337 006624      DECB        CHRCNT      ;STORE THE CHARACTER, STARTING WITH THE MOST
1506 006554 001374      BNE          4$         ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
1507 006556 105700      TSTB        R0          ;IF NO, GO TRANSFER ANOTHER
1508 006560 001404      BEQ          6$         ;ARE ANY SPACES TO BE PRINTED?
1509 006562 112723 000040      5$: MOVVB        #040,(R3)+  ;IF NO, DON'T SET UP ANY
1510 006566 105300      DECB        R0          ;ADD A SPACE TO THE OUTPUT BUFFER
1511 006570 001374      BNE          5$         ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
1512 006572 105013      6$: CLRB        (R3)     ;IF YES, GO ADD ANOTHER SPACE
1513 006574 104402 010724      TYPE        ,MDATA     ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
1514 006600 005337 006622      DEC          WRDCNT      ;PRINT THE STRING WE JUST BUILT
1515 006604 001336      BNE          1$         ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
1516 006606 012605      MOV          (SP)+,R5     ;IF YES, GO CONVERT THEM
1517 006610 012604      MOV          (SP)+,R4     ;RESTORE R5
1518 006612 012603      MOV          (SP)+,R3     ;RESTORE R4
1519 006614 012601      MOV          (SP)+,R1     ;RESTORE R3
1520 006616 012600      MOV          (SP)+,R0     ;RESTORE R1
1521 006620 000002      RTI                    ;RESTORE R0
1522 006622 000000      WRDCNT: 0              ;RETURN TO THE MAIN PROGRAM
1523 006624      000                  CHRCNT: .BYTE          ;NUMBER OF CHARACTERS TO PRINT
1524 006625      000                  SPACNT: .BYTE          ;NUMBER OF SPACES TO PRINT
1525
1526 006626 000000      BINWRD: 0
1527
1528
1529
;TRAP DISPATCH SERVICE
    
```

```

1530                                     ; ARGUMENT OF TRAP IS EXTRACTED
1531                                     ; AND USED AS OFFSET TO OBTAIN POINTER
1532                                     ; TO SELECTED SUBROUTINE
1533
1534 006630 010046 .TRPSR: MOV      RO, -(SP)          ; SAVE RO, USE RO TO FIND TRAP ROUTINE
1535 006632 016600 000002      MOV      2(SP), RO        ; GET TRAP ADDRESS
1536 006636 005740          TST      -(RO)          ; GET TRAP
1537 006640 111000          MOVB   (RO), RO        ; GET RIGHT BYTE OF TRAP (TRAP OFFSET)
1538 006642 006300          ASL     RO            ; POSITION OFFSET FOR TABLE INDEXING
1539 006644 016000 002002      MOV     TRPTAB(RO), RO    ; PLACE INDEXED ADDRESS OF TABLE IN RO
1540 006650 000200          RTS     RO            ; TRANSFER TO THAT ADDRESS AND RESTORE OLD RO
1541
1542                                     ; DEVICE CLEAR ROUTINE
1543                                     ; ISSUE A DEVICE CLEAR
1544 -----
1545 006652 .DEVICE.CLR:
1546 006652 052777 000020 173162 BIS     #DCLR, @DZCSR    ; SET DCLR
1547 006660 032777 000020 173154 1$: BIT     #DCLR, @DZCSR    ; DID IT CLEAR?
1548 006666 001374          BNE    1$            ; BR IF NO
1549 006670 000002          RTI                    ; EXIT ROUTINE
1550
1551                                     ; ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
1552 -----
1553 006672 104413 .DCLASM: DEVICE.CLR    ; ISSUE A DEVICE CLEAR
1554 006674 153777 001417 173140 BISB   MNTFLG, @DZCSR    ; LOAD THE MAINTENANCE BIT IF IT IS I MODE
1555 006702 000002          RTI                    ; RETURN TO CALLING ROUTINE
1556
1557 .DELAY:
1558 006704 010046      MOV     RO, -(SP)          ; SAVE RO
1559 006706 013700 006722      MOV     DLYCNT, RO        ; SET COUNT
1560 006712 005300      1$: DEC     RO            ; DELAY
1561 006714 001376          BNE    1$            ;
1562 006716 012600          MOV     (SP)+, RO        ; RESTORE RO
1563 006720 000002          RTI                    ; LEAVE ROUTINE
1564 006722 000001      DLYCNT: .WORD    1        ; PATCHABLE LOC FOR MORE TIME
1565
1566                                     ; ADVANCE TO NEXT TEST HANDLER
1567 -----
1568
1569 006724 013716 .ADVANCE: MOV     NEXT, (SP)        ; CRUNCH STACK WITH ADDRESS OF SCOPE CALL
1570 006730 005037 001362      CLR     LOCK          ; RESET TIGHT LOOP ADDRESS
1571 006734 000002          RTI                    ; CHECK TO SEE IF OLD TEST GETS REPEATED
1572
1573                                     ; ERROR HANDLER
1574 -----
1575
1576 006736 004737 $ERROR: JSR     PC, SERV.G        ; FIND OUT IF <IG> WAS HIT
1577 006742 032777 010000 172210 BIT     #SW12, @SWR      ; BELL ON ERROR?
1578 006750 001406          BEQ    XBX          ; BR IF NO BELL
1579 006752 105777 172212          TSTB  @STPS         ; TTY READY.
1580 006756 100003          BPL   XBX          ; DON'T WAIT IF TTY NOT READY.
1581 006760 112777 000207 172204 MOVB   #207, @STPB     ; PUSH A BELL AT THE TTY.
1582 006766 032777 020000 172164 XB$: BIT     #SW13, @SWR      ; DELETE ERROR PRINT OUT?
1583 006774 001113          BNE   HALTS        ; BR IF NO PRINT OUT WANTED.
1584 006776 021637 001136          CMP   (SP), $ERRPC   ; WAS THIS ERROR FOUND LAST TIME?
1585 007002 001404          BEQ   1$            ; BR IF YES
    
```

1586	007004	011637	001136		MOV	(SP), \$ERRPC	: RECORD BEING HERE
1587	007010	105037	001123		CLRB	\$ERFLG	: PREPARE HEADER
1588	007014	104407		1\$:	SAVOS		: SAVE ALL PROC REGISTERS
1589	007016	011605			MOV	(SP), R5	: GET THE PC OF ERROR
1590	007020	162705	000002		SUB	#2, R5	: GET ADDRESS OF TRAP CALL
1591	007024	011504			MOV	(R5), R4	: GET ERROR INSTRUCTION
1592	007026	110437	001134		MOVB	R4, \$ITEMB	: COPY TEST NUMBER FOR APT HANDLING
1593	007032	006304			ASL	R4	: MULT BY TWO
1594	007034	061504			ADD	(R5), R4	: DOUBLE IT
1595	007036	006304			ASL	R4	: MULT AGAIN
1596	007040	042704	177001		BIC	#177001, R4	: CLEAR JUNK
1597	007044	062704	027064		ADD	#.ERRTAB, R4	: GET POINTER
1598	007050	012437	007174		MOV	(R4)+, ERRMSG	: GET ERROR MESSAGE
1599	007054	012437	007206		MOV	(R4)+, DATAHD	: GET DATA HEADER
1600	007060	011437	007220		MOV	(R4), DATABP	: GET DATA TABLE
1601	007064	105737	001123		TSTB	\$ERFLG	: TYPE HEADER
1602	007070	001403			BEQ	TYPMSG	: BR IF YES
1603	007072	005737	007220		TST	DATABP	: DOES DATA TABLE EXIST?
1604	007076	001044			BNE	TYPDAT	: BR IF YES.
1605	007100	104402	001231	TYPMSG:	TYPE	, \$CRLF	: TYPE A CARRIAGE RETURN
1606	007104	104402	001231		TYPE	, \$CRLF	: AND TYPE ANOTHER
1607	007110	005737	001362		TST	LOCK	
1608	007114	001402			BEQ	1\$	
1609	007116	104402	010315		TYPE	, MASTEK	
1610	007122	104402	010303	1\$:	TYPE	, MTSTN	
1611	007126	104412	007352		CNVRT	, XTSTN	: SHOW IT
1612	007132	104402	010372		TYPE	, MERRPC	: TYPE PC.
1613	007136	104412	007344		CNVRT	, ERTABO	: SHOW IT
1614	007142	104402	010245		TYPE	, MCSRX	
1615	007146	104412	004742		CNVRT	, XCSR	
1616	007152	104402	001231		TYPE	, \$CRLF	: GIVE A CR/LF
1617	007156	112737	177777	001123	MOVB	#-1, \$ERFLG	: NO MORE HEADER UNLESS NO DATA TABLE.
1618	007164	005737	007174		TST	ERRMSG	: IS THERE AN ERROR MESSAGE?
1619	007170	001402			BEQ	WTBS.FM	: BR IF NO.
1620	007172	104402			TYPE		: TYPE
1621	007174	000000		ERRMSG:	0		: ERROR MESSAGE
1622	007176			WTBS.FM:			
1623	007176	005737	007206		TST	DATAHD	: DATA HEADER?
1624	007202	001402			BEQ	TYPDAT	: BR IF NO
1625	007204	104402			TYPE		: TYPE
1626	007206	000000		DATAHD:	0		: DATA HEADER
1627	007210	005737	007220	TYPDAT:	TST	DATABP	: DATA TABLE?
1628	007214	001402			BEQ	RESREG	: BR IF NO.
1629	007216	104411			CONVRT		: SHOW
1630	007220	000000		DATABP:	0		: DATA TABLE
1631	007222	104410		RESREG:	RESOS		: RESTORE PROC REGISTERS
1632	007224	122737	000001	001254	HALTS:	CMPB	: IS APT RUNNING?
1633	007232	001007			BNE	2\$,	: SKIP APT CALL IF NOT
1634	007234	113737	001134	007246	MOVB	\$ITEMB, 7\$: COPY ERROR NUMBER
1635	007242	004737	005600		JSR	PC, \$ATY4	: CALL APT SERVICE
1636	007246	000000		7\$:	.WORD	0	: ERROR NUMBER STUCK HERE
1637	007250	000777		8\$:	BR	8\$: LOCK UP HERE
1638	007252	022737	004726	000042	2\$:	CMP	: CHECK TO SEE IF IN ACT-11 MODE
1639	007260	001403			BEQ	1\$: IF SO, HANDLE ACCORDINGLY
1640	007262	005777	171672		TST	\$SWR	: HALT ON ERROR?
1641	007266	100004			BPL	EXITER	: BR IF NO HALT ON ERROR

1642	007270	016677	000002	171664	1\$:	MOV	2(SP),@DISPLAY	:SHOW ERROR PC IN DATA DISPLAY
1643	007276	000000				HALT		:HALT
1644	007300	005237	001132		EXITER:	INC	\$ERTTL	:UPDATE ERROR COUNT
1645	007304	032777	000400	171646		BIT	#SW08,@SWR	:GOTO TOP OF TEST?
1646	007312	001007				BNE	1\$:BR IF YES
1647	007314	032777	002000	171636		BIT	#SW10,@SWR	:GOTO NEXT TEST?
1648	007322	001407				BEQ	2\$:BR IF NO
1649	007324	013737	001360	001126		MOV	NEXT,\$LPADR	:SET FOR NEXT TEST
1650	007332	012706	001120		1\$:	MOV	#STACK,SP	:RESET SP
1651	007336	000177	171564			JMP	@\$LPADR	:GOTO SPECIFIED TEST
1652	007342	000002			2\$:	RTI		:RETURN
1653	007344	000001			ERTABO:	1		
1654	007346	006	002			.BYTE	6,2	
1655	007350	001402				SAVPC		
1656	007352	000001			XTSTN:	1		
1657	007354	002	002			.BYTE	2,2	
1658	007356	001122				\$TSTNM		
1659	007360	022737	177570	001160	SERV.G:	CMP	#177570,\$SWR	:IS THE SWITCH REGISTER HARDWIRED?
1660	007366	001513				BEQ	6\$:IF SO, IGNORE ↑G
1661	007370	017746	171572			MOV	@\$TKB,-(SP)	:OTHERWISE, GET THE LAST CHARACTER TYPED
1662	007374	042716	000200			BIC	#BIT7,(SP)	:STRIP PARITY(EIGHTH) BIT
1663	007400	122726	000007			CMPB	#7,(SP)+	:IS IT ↑G?
1664	007404	001104				BNE	6\$:IF NOT, IGNORE INPUT
1665	007406	032777	004000	171550		BIT	#4000,\$STKS	:RX BUSY?
1666	007414	001361				BNE	SERV.G	:BR IF YES
1667	007416	017737	171536	007640		MOV	@\$SWR,90\$:SAVE (SWR).
1668	007424	013777	007640	171526	1\$:	MOV	90\$,@\$SWR	
1669	007432	104402	007620			TYPE	,89\$	
1670	007436	104412	007632			CNVRT	,88\$	
1671	007442	104402	007642			TYPE	,91\$	
1672	007446	105777	171512			TSTB	@\$TKS	:WAIT FOR DONE.
1673	007452	100375				BPL	,-4	
1674	007454	017746	171506			MOV	@\$TKB,-(SP)	
1675	007460	042716	000200			BIC	#BIT7,(SP)	
1676	007464	122726	000015			CMPB	#15,(SP)+	
1677	007470	001450				BEQ	5\$	
1678	007472	005077	171462			CLR	@\$SWR	
1679	007476	105777	171466		2\$:	TSTB	@\$TPS	
1680	007502	100375				BPL	,-4	
1681	007504	016677	177776	171460		MOV	-2(SP),@\$TPB	
1682	007512	000241				CLC		
1683	007514	006177	171440			ROL	@\$SWR	
1684	007520	006177	171434			ROL	@\$SWR	
1685	007524	006177	171430			ROL	@\$SWR	
1686	007530	103735				BCS	1\$:ERROR
1687	007532	026627	177776	000060		CMP	-2(SP),#60	
1688	007540	002731				BLT	1\$	
1689	007542	026627	177776	000067		CMP	-2(SP),#67	
1690	007550	003325				BGT	1\$	
1691	007552	042766	177770	177776		BIC	#↑C(7),-2(SP)	
1692	007560	056677	177776	171372		BIS	-2(SP),@\$SWR	
1693	007566	105777	171372			TSTB	@\$TKS	
1694	007572	100375				BPL	,-4	
1695	007574	017746	171366			MOV	@\$TKB,-(SP)	
1696	007600	042716	000200			BIC	#BIT7,(SP)	
1697	007604	122726	000015			CMPB	#15,(SP)+	


```

1698 007610 001332
1699 007612 104402 001231 5$: BNE 2$
1700 007616 000207 6$: TYPE $CRLF
1701 1702 007620 020200 051450 051127 89$: .ASCIZ <200>? (SWR)=/?
1703 007626 036451 000057
1704 .EVEN
1705 007632 000001 88$: 1
1706 007634 006 000 .BYTE 6,0
1707 007636 007640 90$:
1708 007640 000000 91$: .WORD 0
1709 007642 036457 000057 91$: .ASCIZ ?/=/?
1710 .EVEN
1711 .SBTTL POWER DOWN AND UP ROUTINES
1712
1713 ;*****
1714 ;POWER DOWN ROUTINE
1715 007646 012737 010012 000024 $PWRDN: MOV $SILLUP, @#PWRVEC ; SET FOR FAST UP
1716 007654 012737 000340 000026 MOV #340, @#PWRVEC+2 ; PRIO:7
1717 007662 010046 MOV R0, -(SP) ; PUSH R0 ON STACK
1718 007664 010146 MOV R1, -(SP) ; PUSH R1 ON STACK
1719 007666 010246 MOV R2, -(SP) ; PUSH R2 ON STACK
1720 007670 010346 MOV R3, -(SP) ; PUSH R3 ON STACK
1721 007672 010446 MOV R4, -(SP) ; PUSH R4 ON STACK
1722 007674 010546 MOV R5, -(SP) ; PUSH R5 ON STACK
1723 007676 017746 171256 MOV @SWR, -(SP) ; PUSH @SWR ON STACK
1724 007702 010637 010016 MOV SP, $SAVR6 ; SAVE SP
1725 007706 012737 007720 000024 MOV $PWRUP, @#PWRVEC ; SET UP VECTOR
1726 007714 000000 HALT
1727 007716 000776 BR -2 ; HANG UP
1728
1729 ;*****
1730 ;POWER UP ROUTINE
1731 007720 012737 010012 000024 $PWRUP: MOV $SILLUP, @#PWRVEC ; SET FOR FAST DOWN
1732 007726 013706 010016 MOV $SAVR6, SP ; GET SP
1733 007732 005037 010016 CLR $SAVR6 ; WAIT LOOP FOR THE TTY
1734 007736 005237 010016 1$: INC $SAVR6 ; WAIT FOR THE INC
1735 007742 001375 BNE 1$ ; OF WORD
1736 007744 012677 171210 MOV (SP)+, @SWR ; POP STACK INTO @SWR
1737 007750 012605 MOV (SP)+, R5 ; POP STACK INTO R5
1738 007752 012604 MOV (SP)+, R4 ; POP STACK INTO R4
1739 007754 012603 MOV (SP)+, R3 ; POP STACK INTO R3
1740 007756 012602 MOV (SP)+, R2 ; POP STACK INTO R2
1741 007760 012601 MOV (SP)+, R1 ; POP STACK INTO R1
1742 007762 012600 MOV (SP)+, R0 ; POP STACK INTO R0
1743 007764 012737 007646 000024 MOV $PWRDN, @#PWRVEC ; SET UP THE POWER DOWN VECTOR
1744 007772 012737 000340 000026 MOV #340, @#PWRVEC+2 ; PRIO:7
1745 010000 104402 SPWRMG: .WORD MPFAIL ; REPORT THE POWER FAILURE
1746 010002 010020 .WORD MPFAIL ; POWER FAIL MESSAGE POINTER
1747 010004 012716 SPWRAD: MOV (PC)+, (SP) ; RESTART AT RESTART
1748 010006 011434 .WORD RESTART ; RESTART ADDRESS
1749 010010 000002 RTI
1750 010012 000000 $SILLUP: HALT ; THE POWER UP SEQUENCE WAS STARTED
1751 010014 000776 BR -2 ; BEFORE THE POWER DOWN WAS COMPLETE
1752 010016 000000 $SAVR6: 0 ; PUT THE SP HERE
1753 010020 050200 051127 043040 MPFAIL: .ASCIZ <200>/PWR FAILED. RESTART AT LAST TEST /

```

```

(2) 010063      200 047105 020104 MEPASS: .ASCIZ <200>/END PASS DZDZA-E /
(2) 010107      200 052522 047116 MR: .ASCIZ <200>/RUNNING /
(2) 010123      200 051120 043517 MERR2: .ASCIZ <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 010172      044600 051516 043125 MERR3: .ASCIZ <200>/INSUFFICIENT DATA!/
(2) 010216      046200 041517 020113 MLOCK: .ASCIZ <200>/LOCK ON SELECTED TEST/
(2) 010245      103 051123 020072 MCSRX: .ASCIZ /CSR: /
(2) 010253      126 041505 020072 MVECX: .ASCIZ /VEC: /
(2) 010261      120 051501 042523 MPASSX: .ASCIZ /PASSES: /
(2) 010272      051105 047522 051522 MERRX: .ASCIZ /ERRORS: /
(2) 010303      124 051505 020124 MTSTN: .ASCIZ /TEST NO: /
(2) 010315      052 000040 MASTEK: .ASCIZ /* /
(2) 010320      051600 052105 051440 MNEW: .ASCIZ <200>/SET SWITCH REG TO DZ11'S DESIRED ACTIVE./
(2) 010372      041520 020072 000 MERRPC: .ASCIZ /PC: /
(2) 010377      200 040515 020120 XHEAD: .ASCIZ <200>/MAP OF DZ11 STATUS/<200>
(2) 010424      044600 046114 043505 MBADLN: .ASCIZ <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2) 010466      000002 XSTATQ: 2
1754 010470      006 003 .BYTE 6,3
1755 010472      001220 $TMP1
1756 010474      006 002 .BYTE 6,2
1757 010476      001222 $TMP2
1758 .EVEN
1759 ; THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
1760 ;-----
1761 ;E=EXTERNAL LOOP BACK
1762 ;I=INTERNAL LOOP BACK
1763 ;S=STAGGERED LOOP BACK
1764 010500      017605 000000 .SETFLG: MOV 2(SP),R5 ; PICK UP ADDRESS OF TAG
1765 010504      042737 000040 010620 BIC #40,INBUF ; STRIP LOWER CASE
1766 010512      122737 000105 010620 CMPB #'E,INBUF ; IS IT EXTERNAL LOOP BACK ?
1767 010520      001005 BNE 4$ ; NO
1768 010522      013715 010612 MOV 1$(R5) ; YES STORE INFO
1769 010526      105037 001417 CLRB MNTFLG ; SET MAINT BIT =0
1770 010532      000422 BR 7$ ; GET OUT
1771 010534      122737 000111 010620 4$: CMPB #'I,INBUF ; IS IT INTERNAL LOOP BACK ?
1772 010542      001006 BNE 5$ ; NO
1773 010544      013715 010614 MOV 2$(R5) ; YES STORE INFO
1774 010550      112737 000010 001417 MOVB #MAINT,MNTFLG ; SET UP THE MAINTENANCE FLAG LOADER
1775 010556      000410 BR 7$ ; GET OUT
1776 010560      122737 000123 010620 5$: CMPB #'S,INBUF ; IS IT STAGGERED LOOP BACK ?
1777 010566      001007 BNE 6$ ; WHAT ?
1778 010570      013715 010616 MOV 3$(R5) ; YES STORE INFO
1779 010574      105037 001417 CLRB MNTFLG ; ZERO BITS
1780 010600      062716 000002 7$: ADD #2,(SP) ; POP AROUND
1781 010604      000002 RTI
1782 010606      104404 6$: INSTER ; RETRY
1783 010610      000733 BR .SETFLG ; DITTO
1784 010612      000200 1$: .WORD 200 ; EXTERNAL = E
1785 010614      000000 2$: .WORD 0 ; INTERNAL = I
1786 010616      100000 3$: .WORD 100000 ; STAGGERED = S
1787
1788 ; BUFFERS FOR INPUT-OUTPUT
1789
1790 010620      000000 INBUF: 0
1791 010662      010662 .+.40
1792 010662      000000 TEMP: 0

```

1793		010724				. = +40			
1794	010724	000000				MDATA:	0		
1795		010766				. = +40			
1796									
1797	010766	011637	011064			SET.PS:	MOV	(SP), 3\$	
1798	010772	162737	000002	011064			SUB	#2, 3\$	
1799	011000	017737	000060	011066			MOV	23\$, 4\$	
1800	011006	022737	106427	011066			CMP	#106427, 4\$	
1801	011014	001003					BNE	1\$	
1802	011016	011637	011064				MOV	(SP), 3\$	
1803	011022	000412					BR	2\$	
1804	011024	022737	106437	011066	1\$:		CMP	#106437, 4\$	
1805	011032	001401					BEQ	. +4	
1806	011034	000000					HALT		; RESERVED INSTRUCTION NOT "MTPS"
1807	011036	011637	011064				MOV	(SP), 3\$	
1808	011042	017737	000016	011064			MOV	23\$, 3\$	
1809	011050	062716	000002		2\$:		ADD	#2, (SP)	
1810	011054	017766	000004	000002			MOV	23\$, 2(SP)	
1811	011062	000002					RTI		
1812	011064	000000			3\$:		0		
1813	011066	000000			4\$:		0		

```

1814
1815
1816
1817
1818
1819
1820
1821
1822
1823 011070 005737 001404          CYCLE: TST      DZACTV          ;ARE ANY DZ11'S TO BE TESTED?
1824 011074 001004                    BNE      1$          ;BR IF OK.
1825 011076 104402 010123          TYPE     ,MERR2     ;NO DZ11'S SELECTED!!
1826 011102 000000                    HALT                    ;STOP THE SHOW.
1827 011104 000776                    BR      -2          ;DISQUALIFY CONT. SW.
1828 011106 013737 005234 001226 1$: MOV     $MXCNT,$TIMES ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
1829 011114 033737 001406 001404 BIT     RUN,DZACTV  ;IS THIS ONE "ACTIVE"
1830 011122 001020                    BNE     2$          ;BR IF GOOD ONE FOUND.
1831 011124 000241                    CLC
1832 011126 006137 001406          ROL     RUN          ;UPDATE POINTER
1833 011132 005537 001406          ADC     RUN          ;CATCH CARRY FROM RUN
1834 011136 062737 000014 0C1412 ADD     #14,ACTIVE  ;UPDATE ADDRESS POINTER.
1835 011144 022737 002000 001412 CMP     #DZ.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
1836 011152 001355                    BNE     1$          ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
1837 011154 012737 001500 001412 MOV     #DZ.MAP,ACTIVE ;RESET ADDRESS POINTER.
1838 011162 000751                    BR      1$          ;KEEP LOOKING FOR ACTIVE DZ11
1839 011164 000241                    CLC
1840 011166 006137 001406          ROL     RUN          ;UPDATE POINTER.
1841 011172 005537 001406          ADC     RUN          ;CATCH CARRY.
1842 011176 013700 001412          MOV     ACTIVE,RO   ;GET ADDRESS POINTER.
1843 011202 062737 000014 001412 ADD     #14,ACTIVE  ;UPDATE.
1844 011210 022737 002000 001412 CMP     #DZ.END,ACTIVE ;ALL DONE?
1845
1846 011216 001003                    BNE     3$          ;BR IF NO.
1847 011220 012737 001500 001412 MOV     #DZ.MAP,ACTIVE ;RESTORE POINTER.
1848 011226 012037 001310          MOV     (RO)+,$BASE ;LOAD SYSTEM CTRL. REG
1849 011232 012037 002072          MOV     (RO)+,DZRIV ;LOAD VECTOR
1850 011236 012037 027060          MOV     (RO)+,DZPRT ;LOAD PRIORITY
1851 011242 113737 027061 001414 MOVB   DZPRT+1,EIAFLG ;EIA OR ZOMA
1852 011250 042737 100000 027060 BIC     #BIT15,DZPRT ;CLEAR FLAG
1853 011256 012037 001364          MOV     (RO)+,LINE  ;SET UP LINE DZ LINES ACTIVE
1854 011262 012037 001366          MOV     (RO)+,PAR   ;SET UP PARAMETERIZATION
1855 011266 012037 001370          MOV     (RO)+,MODE  ;SET UP MAINTENANCE MODE
1856 011272 004737 026652          JSR    PC,DZLEV    ;SET UP
1857 011276 005737 000042          TST    @#42        ;ARE WE UNDER MONITOR CONTROL?
1858 011302 001051                    BNE     4$          ;IF YES, SKIP THIS SETUP
1859 011304 032777 000002 167646 BIT     #SW01,@SWR  ;IF SW01=1, GET STARTING TEST #
1860 011312 001445                    BEQ     4$          ;BR IF NO TEST IS TO BE INPUTTED
1861 011314 104402 001231          TYPE     ,$CRLF
1862 011320 104403                    INSTR
1863 011322 010303                    MTSTN
1864 011324 104405                    PARAM
1865 011326 000001                    I
1866 011330 001000                    1000
1867 011332 001122                    $TSTNM
1868 011334 000                    .BYTE 0
1869 011335 001                    .BYTE 1
    
```

1870	011336	012700	012374			MOV	#TST1,RO	
1871	011342	022710	000004		5S:	CMP	#4,(RO)	
1872	011346	001020				BNE	6S	
1873	011350	022760	012737	000002		CMP	#12737,2(RO)	
1874	011356	001014				BNE	6S	
1875	011360	023760	001122	000004		CMP	\$TSTNM,4(RO)	; IS THIS THE TEST ?
1876	011366	001010				BNE	6S	; IF NOT, DON'T PROCESS NUMBER
1877	011370	010037	001126			MOV	RO,\$LPADR	; SAVE PC
1878	011374	062737	000002	001126		ADD	#2,\$LPADR	; POP OVER SCOPE
1879	011402	104402	001231			TYPE	\$CRLF	
1880	011406	000412				BR	6S	
1881	011410	005720			6S:	TST	(RO)+	
1882	011412	020027	023002			CMP	RO,#TLAST+10	
1883	011416	001351				BNE	5S	
1884	011420	104402	001230			TYPE	\$QUES	
1885	011424	000733				BR	7S	
1886	011426	012737	012374	001126	4S:	MOV	#TST1,\$LPADR	; PREPARE TEST ADDRESS
1887	011434				8S:			
1888	011434	000177	167466			RESTART:JMP	2\$LPADR	; GO START TESTING.***WARNING!***
1889								; THIS JUMP IS USED BY POWER UP ROUTINE!!!!
1890								

```

1891 ;--ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
1892 ; IF BIT7 IN THE ENVIRONMENT MODE ($ENVM) BYTE IS SET.
1893 ; THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLÉ.
1894
1895 011440 012700 001500      SETAPT: MOV      #DZ.MAP,R0      ;POINT TO THE DEVICE MAP TABLE
1896 011444 013701 001310      MOV      $BASE,R1      ;BUILD DEVICE ADDRESSES IN R1
1897 011450 013702 001304      MOV      $VECT1,R2     ;BUILD DEVICE VECTORS IN R2
1898 011454 042702 177007      BIC      #1C<770>,R2   ;STRIP AWAY OTHER INFORMATION
1899
1900 011460 113703 001305      MOVB     $VECT1+1,R3   ;LOAD THE INTERRUPT PRIORITY FROM R3
1901 011464 106003              RORB     R3            ;ALIGN THE NUMBER
1902 011466 106003              RORB     R3            ;ALIGN THE NUMBER
1903 011470 106003              RORB     R3            ;ALIGN THE NUMBER
1904 011472 106003              RORB     R3            ;ALIGN THE NUMBER
1905 011474 106003              RORB     R3            ;ALIGN THE NUMBER
1906 011476 042703 177770      BIC      #1C<7>,R3    ;REMOVE ALL BUT BUS LEVEL NUMBER
1907 011502 012704 001320      MOV      #SDDWO,R4    ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
1908 011506 013705 001312      MOV      $DEVM,R5     ;GET THE MAP OF ACTIVE DEVICES
1909 011512 010537 001404      MOV      R5,DZACTV    ;SAVE THE BIT MAP
1910 011516 006005              1$:     ROR      R5      ;GET A DEVICE SELECTION BIT
1911 011520 103407              BCS     3$            ;IF IT IS SELECTED, GO SET UP A MAP
1912 011522 001425              BEQ     5$            ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
1913 011524 005724              TST     (R4)+         ;POINT TO NEXT DEVICE DESCRIPTOR
1914 011526 062701 000010      2$:     ADD     #10,R1   ;SET UP THE NEXT ADDRESS
1915 011532 062702 000010      ADD     #10,R2        ;SET UP THE NEXT VECTOR GROUP
1916 011536 000767              BR      1$            ;GO SEE IF MORE DEVICES REMAIN
1917 011540 010120      3$:     MOV     R1,(R0)+    ;LOAD DEVICE ADDRESS
1918 011542 010220      MOV     R2,(R0)+    ;LOAD THE VECTOR ADDRESS
1919 011544 010320      MOV     R3,(R0)+    ;LOAD THE INTERRUPT PRIORITY LEVEL
1920 011546 013720 001314      MOV     $CDW1,(R0)+  ;GET THE NUMBER OF LINES IN OPERATION
1921 011552 012420      MOV     (R4)+,(R0)+  ;LOAD DEVICE PARAMETERS
1922 011554 100006              BPL     4$            ;IF 20MA MODE SELECTED, SET IT UP
1923 011556 052760 100000 177772      BIS     #100000,-6(R0);SET THE 20MA FLAG IN DZLVN
1924 011564 042760 100000 177776      BIC     #100000,-2(R0);CLEAR THE FLAG IN DZPARN
1925 011572 005020      4$:     CLR     (R0)+     ;DEFAULT OPERATION TO INTERNAL MAINTENANCE MODE
1926 011574 000754              BR      2$            ;GO BUILD THE NEXT ADDRESS
1927 011576 012710 177777      5$:     MOV     #-1,(R0)  ;TERMINATE THE DEVICE MAP
1928 011602 012737 001256 001160      MOV     #SSWREG,SWR  ;SET TO SOFTWARE APT SWITCH REGISTER
1929 011610 000207              RTS     PC            ;RETURN TO PRINT STATUS TABLE
1930
1931
1932 ;*ROUTINE USED TO "AUTO SIZE" THE DZ11
1933 ;*CSR AND VECTOR.
1934 ;*NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1935 ;* ADDRESS RANGE (160000:163700)
1936 ;* AND THE VECTOR MAY BE ANY WHERE IN THE
1937 ;* FLOATING VECTOR RANGE (300:770)
1938 ;*
1939
1940 011612              AUTO.SIZE:
1941 011612 000005              RESET
1942 011614 105337 001415      DECB
1943 011620 012702 001500      CSRMAP: MOV     #DZ.MAP,R2 ;INSURE A BUS INIT.
1944 011624 012703 001320      MOV     #SDDWO,R3    ;SHOW THAT I WAS HERE
1945 011630 005022              1$:     CLR     (R2)+     ;LOAD MAP POINTER.
1946 011632 022702 002000      CMP     #DZ.END,R2  ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
                          ;ZERO ENTIRE MAP
                          ;ALL DONE?
    
```

1947	011636	001374			BNE	1\$;BR IF NO
1948	011640	105037	001410		CLRB	DZNUM		;SET OCTAL NUMBER OF DZ11'S TO 0
1949	011644	012702	001500		MOV	#DZ.MAP,R2		
1950	011650	012701	160000		MOV	#160000,R1		;SET FOR FIRST ADDRESS TO BE TESTED
1951	011654	012737	012174	000004	MOV	#6\$,2#4		;SET FOR NON-EXISTENT DEVICE TIME OUT
1952	011662	052711	000040		BIS	#BITS,(R1)	2\$:	;TRY TO SET MASTER SCAN ENABLE
1953	011666	052761	000200	000004	BIS	#BIT7,4(R1)		;TRY TO TRANSMIT ON LINE 7
1954	011674	005000			CLR	RO		;USE RO AS A COUNTER
1955	011676	005711			TST	(R1)	7\$:	;HAS TRANSMITTER READY COME UP?
1956	011700	100403			BMI	8\$;IF SO, GO GET A FINAL CHECK
1957	011702	005300			DEC	RO		;REDUCE COUNT. TIME UP?
1958	011704	001374			BNE	7\$;IF NOT, KEEP WAITING
1959	011706	000463			BR	3\$;ASSUME IT'S NOT A DZ11
1960	011710	032761	000200	000004	BIT	#BIT7,4(R1)	8\$:	;IS LINE 7 ENABLE STILL SET? IT SHOULD BE
1961	011716	001457			BEQ	3\$;IF IT'S NOT, ASSUME IT'S NOT A DZ11
1962	011720	032711	000040		BIT	#BITS,(R1)		;IS MASTER SCAN ENABLE STILL SET?
1963	011724	001454			BEQ	3\$;IF NOT, ASSUME IT'S NOT A DZ11
1964	011726	005000			CLR	RO		
1965	011730	052711	000020		BIS	#20,(R1)		;SET DEVICE CLEAR
1966	011734	032711	000020		BIT	#20,(R1)		;SHOULD STAY SET FOR A WHILE IF DZ
1967	011740	001446			BEQ	3\$;BR IF NOT DZ11
1968	011742	032711	000020		BIT	#20,(R1)		;WAIT FOR BIT TO CLEAR
1969	011746	001404			BEQ	+.12		;BR WHEN CLEARED
1970	011750	104414			DELAY			
1971	011752	005200			INC	RO		
1972	011754	001372			BNE	.-12		
1973	011756	000437			BR	3\$;BIT NOT CLEARED! MUST NOT BE DZ11
1974	011760	005011			CLR	(R1)		;GET RID OF MASTER SCAN ENABLE
1975	011762	005061	000004		CLR	4(R1)		;GET RID OF LINE 7 ENABLE
1976								;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZ11 CSR ADDRESS.
1977	011766	010122			MOV	R1,(R2)+		;STORE CSR IN CORE TABLE.
1978	011770	005722			TST	(R2)+		;POP OVER VECTOR STORE AREA
1979	011772	012722	000005		MOV	#5,(R2)+		;SET THE DEFAULT BUS LEVEL
1980	011776	052761	177400	000004	BIS	#177400,4(R1)		;TRY TO SET ALL DTR BITS
1981	012004	032761	177400	000004	BIT	#177400,4(R1)		;IF ANY SET ASSUME EIA BOARD
1982	012012	001003			BNE	9\$;IF NONE SET ASSUME BOARD IS
1983	012014	052762	100000	177776	BIS	#BIT15,-2(R2)		;20 MA, SET 20 MA FLAG
1984	012022	012722	000377		MOV	#377,(R2)+	9\$:	;SET THE DEFAULT LINE SELECTION PARAMETER
1985	012026	012712	017470		MOV	#17470,(R2)		;SET THE DEFAULT PARAMETERS
1986	012032	012223			MOV	(R2)+,(R3)+		;COPY PARAMETERS INTO ETABLE DESCRIPTOR
1987	012034	005022			CLR	(R2)+		;SET THE DEFAULT MODE OF OPERATION
1988	012036	012712	177777		MOV	#-1,(R2)		;TERMINATE LIST
1989	012042	105237	001410		INCB	DZNUM		;UPDATE DEVICE COUNTER
1990	012046	122737	000020	001410	CMPB	#20,DZNUM		;ARE MAX. NO. OF DEV FOUND?
1991	012054	001405			BEQ	100\$;YES DON'T LOOK FOR ANY MORE.
1992	012056	062701	000010		ADD	#10,R1	3\$:	;UPDATE CSR POINTER ADDRESS
1993	012062	022701	163700		CMP	#163700,R1		
1994	012066	001275			BNE	2\$;BR IF MORE ADDRESS TO CHECK.
1995	012070						100\$:	
1996	012070	105737	001410		TSTB	DZNUM		;WERE ANY DZ11'S FOUND AT ALL?
1997	012074	001432			BEQ	5\$;ERROR AUTO SIZER FOUND NO DZ11'S IN THIS SYS.
1998	012076	113701	001410		MOVB	DZNUM,R1		
1999	012102	110137	001411		MOVB	R1,SAVNUM		;SAVE NUMBER OF DEVICES
2000	012106	012737	000001	001404	MOV	#1,DZACTV		
2001	012114	005301			DEC	R1	4\$:	
2002	012116	001404			BEQ	98\$		

H06

MD-11-DZDZA-E MACY11 30(1046) 03-OCT-77 09:43 PAGE 44
DZDZAE.P11 03-OCT-77 09:39

DZ11 DEVICE DIAGNOSTICS. COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0072

```
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
```

```
***** TEST 1 *****
*THIS TEST PROVES THE SLAVE SYNC RESPONSE
*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
* DZCSR, DZRBUF, DZTCR, DZMSR
::* TEST 1
*****
↑TST1: SCOPE
MOV #1,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST2,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
1$: MOV DZCSR,R0 ;SET ADDRESS TO TEST
MOV (R0),R1 ;READ THE ADDRESS
NOP ;WASTE TIME
CLR (R0) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #25,LOCK ;SET RETURN ADDRESS FOR SW09
2$: MOV DZRBUF,R0 ;SET ADDRESS TO TEST
MOV (R0),R1 ;READ THE ADDRESS
NOP ;WASTE TIME
CLR (R0) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #35,LOCK ;SET RETURN ADDRESS FOR SW09
3$: MOV DZTCR,R0 ;SET ADDRESS TO TEST
MOV (R0),R1 ;READ THE ADDRESS
NOP ;WASTE TIME
CLR (R0) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #45,LOCK ;SET RETURN ADDRESS
4$: MOV DZMSR,R0 ;SET ADDRESS TO TEST
MOV (R0),R1 ;READ FROM ADDRESS
NOP ;WASTE TIME
CLR (R0) ;WRITE THE ADDRESS
NOP ;WASTE TIME
MOV #6,4 ;SET TRAP CATCHER BACK TO NORMAL
5$: 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
*****
↑TST2: SCOPE
MOV #2,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST3,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
```



```

2165 ;:* TEST 4
2166 ;:*****
2167 012742 000004          TST4: SCOPE
2168 012744 012737 000004 001122  MOV #4,STSTNM ;LOAD THE NUMBER OF THIS TEST
2169 012752 012737 013034 001360  MOV #TST5,NEXT ;POINT TO THE START OF THE NEXT TEST
2170 012760 013700 002042  MOV DZCSR,R0 ;GET BASE ADDRESS
2171 012764 012705 000040  MOV #MSENAB,R5 ;SET BIT
2172 012770 010510  MOV R5,(R0) ;SET SET IN DEVICE
2173 012772 011004  MOV (R0),R4 ;READ THE BIT FROM DEVICE
2174 012774 020504  CMP R5,R4 ;WAS BIT SET?
2175 012776 001401  BEQ 1$ ;BR IF YES
2176 013000 104002  ERROR 2 ;*BIT R/W FAILURE
2177 013002 040510 1$: BIC R5,(R0) ;CLEAR THE BIT.
2178 013004 011004  MOV (R0),R4 ;READ DEVICE
2179 013006 001404  BEQ 2$ ;BR IF BITS WERE CLEARED.
2180 013010 010546  MOV R5,-(SP) ;SAVE THE BIT
2181 013012 005005  CLR R5 ;SET EXPECTED RESULTS TO 0
2182 013014 104002  ERROR 2 ;*BIT FAILED TO CLEAR
2183 013016 012605  MOV (SP)+,R5 ;RESTORE THE BIT.
2184 013020 010510 2$: MOV R5,(R0) ;SET THE BIT AGAIN
2185 013022 104413  DEVICE.CLR ;ISSUE DEVICE CLEAR
2186 013024 011004  MOV (R0),R4 ;READ THE BIT.
2187 013026 001402  BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
2188 013030 005005  CLR R5 ;SET EXPECTED TO ZERO
2189 013032 104002  ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
2190 013034
2191 ;:***** TEST 5 *****
2192 ;*TEST TO VERIFY THAT BIT "SILOEN" CAN
2193 ;*BE SET. THEN VERIFY THAT BIT "SILOEN" CAN
2194 ;*BE CLEARED (WRITTEN TO A ZERO). AND FINALLY
2195 ;*VERIFY THAT AFTER BEING SET AGAIN IT CAN BE
2196 ;*CLEARED BY A "DEVICE CLEAR"
2197 ;:*****
2198 ;:* TEST 5
2199 ;:*****
2200 013034 000004          TST5: SCOPE
2201 013036 012737 000005 001122  MOV #5,STSTNM ;LOAD THE NUMBER OF THIS TEST
2202 013044 012737 013126 001360  MOV #TST6,NEXT ;POINT TO THE START OF THE NEXT TEST
2203 013052 013700 002042  MOV DZCSR,R0 ;GET BASE ADDRESS
2204 013056 012705 010000  MOV #SILOEN,R5 ;SET BIT
2205 013062 010510  MOV R5,(R0) ;SET SET IN DEVICE
2206 013064 011004  MOV (R0),R4 ;READ THE BIT FROM DEVICE
2207 013066 020504  CMP R5,R4 ;WAS BIT SET?
2208 013070 001401  BEQ 1$ ;BR IF YES
2209 013072 104002  ERROR 2 ;*BIT R/W FAILURE
2210 013074 040510 1$: BIC R5,(R0) ;CLEAR THE BIT.
2211 013076 011004  MOV (R0),R4 ;READ DEVICE
2212 013100 001404  BEQ 2$ ;BR IF BITS WERE CLEARED.
2213 013102 010546  MOV R5,-(SP) ;SAVE THE BIT
2214 013104 005005  CLR R5 ;SET EXPECTED RESULTS TO 0
2215 013106 104002  ERROR 2 ;*BIT FAILED TO CLEAR
2216 013110 012605  MOV (SP)+,R5 ;RESTORE THE BIT.
2217 013112 010510 2$: MOV R5,(R0) ;SET THE BIT AGAIN
2218 013114 104413  DEVICE.CLR ;ISSUE DEVICE CLEAR
2219 013116 011004  MOV (R0),R4 ;READ THE BIT.
2220 013120 001402  BEQ 3$ ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
2220 013122 005005  CLR R5 ;SET EXPECTED TO ZERO

```

2221 013124 104002
2222 013126

3\$: ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR

2223
2224
2225
2226
2227
2228

***** 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"

2229
2230

::* TEST 6

2231 013126 000004
2232 013130 012737 000006 001122
2233 013136 012737 013220 001360
2234 013144 013700 002042
2235 013150 012705 000100
2236 013154 010510
2237 013156 011004
2238 013160 020504
2239 013162 001401
2240 013164 104002
2241 013166 040510
2242 013170 011004
2243 013172 001404
2244 013174 010546
2245 013176 005005
2246 013200 104002
2247 013202 012605
2248 013204 010510
2249 013206 104413
2250 013210 011004
2251 013212 001402
2252 013214 005005
2253 013216 104002
2254 013220

TST6: SCOPE ; LOAD THE NUMBER OF THIS TEST
MOV #6,STSTNM ; POINT TO THE START OF THE NEXT TEST
MOV #TST7,NEXT ; GET BASE ADDRESS
MOV DZCSR,R0 ; SET SET IN DEVICE
MOV #RIE,R5 ; SET BIT ; READ THE BIT FROM DEVICE
MOV R5,(R0) ; WAS BIT SET?
MOV (R0),R4 ; BR IF YES
CMP R5,R4 ; *BIT R/W FAILURE
BEQ 1\$; CLEAR THE BIT.
ERROR 2 ; READ DEVICE
1\$: BIC R5,(R0) ; BR IF BITS WERE CLEARED.
MOV (R0),R4 ; SAVE THE BIT
BEQ 2\$; SET EXPECTED RESULTS TO 0
MOV R5,-(SP) ; *BIT FAILED TO CLEAR
CLR R5 ; RESTORE THE BIT
ERROR 2 ; SET THE BIT AGAIN
2\$: MOV (SP)+,R5 ; ISSUE DEVICE CLEAR
MOV R5,(R0) ; READ THE BIT.
DEVICE.CLR ; BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
MOV (R0),R4 ; SET EXPECTED TO ZERO
BEQ 3\$; *BIT NOT CLEARED BY DEVICE CLEAR
CLR R5
ERROR 2
3\$:

2255
2256
2257
2258
2259
2260
2261

***** 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"

2262

::* TEST 7

2263 013220 000004
2264 013222 012737 000007 001122
2265 013230 012737 013312 001360
2266 013236 013700 002042
2267 013242 012705 040000
2268 013246 010510
2269 013250 011004
2270 013252 020504
2271 013254 001401
2272 013256 104002
2273 013260 040510
2274 013262 011004
2275 013264 001404
2276 013266 010546

TST7: SCOPE ; LOAD THE NUMBER OF THIS TEST
MOV #7,STSTNM ; POINT TO THE START OF THE NEXT TEST
MOV #TST10,NEXT ; GET BASE ADDRESS
MOV DZCSR,R0 ; SET SET IN DEVICE
MOV #TIE,R5 ; SET BIT ; READ THE BIT FROM DEVICE
MOV R5,(R0) ; WAS BIT SET?
MOV (R0),R4 ; BR IF YES
CMP R5,R4 ; *BIT R/W FAILURE
BEQ 1\$; CLEAR THE BIT.
ERROR 2 ; READ DEVICE
1\$: BIC R5,(R0) ; BR IF BITS WERE CLEARED.
MOV (R0),R4 ; SAVE THE BIT
BEQ 2\$; SET EXPECTED RESULTS TO 0
MOV R5,-(SP) ; RESTORE THE BIT
CLR R5 ; SET THE BIT AGAIN
ERROR 2 ; ISSUE DEVICE CLEAR
2\$: MOV (SP)+,R5 ; READ THE BIT.
MOV R5,(R0) ; BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
BEQ 3\$; SET EXPECTED TO ZERO
MOV (R0),R4 ; *BIT NOT CLEARED BY DEVICE CLEAR
BEQ 3\$
3\$:

```

2277 013270 005005 CLR R5 ;SET EXPECTED RESULTS TO 0
2278 013272 104002 ERROR 2 ;*BIT FAILED TO CLEAR
2279 013274 012605 MOV (SP)+,R5 ;RESTORE THE BIT
2280 013276 010510 2S: MOV R5,(R0) ;SET THE BIT AGAIN
2281 013300 104413 DEVICE.CLR ;ISSUE DEVICE CLEAR
2282 013302 011004 MOV (R0),R4 ;READ THE BIT
2283 013304 001402 BEQ 3S ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
2284 013306 005005 CLR R5 ;SET EXPECTED TO ZERO
2285 013310 104002 ERROR 2 ;*BIT NOT CLEARED BY DEVICE CLEAR
2286 013312
2287
2288 ;***** TEST 10 *****
2289 ;*THIS TESTS THAT ALL OF THE FOLLOWING
2290 ;*BITS CAN BE: SET, CLEARED, CLEARED BY "DEVICE CLEAR "
2291 ;*BITS TESTED ARE:
2292 ;* TCR0, TCR1, TCR2, TCR3, TCR4, TCR5, TCR6, TCR7
2293
2294 ;* TEST 10
2295 ;*****
2296 TST10: SCOPE
2297 MOV #10,$STSTM ;LOAD THE NUMBER OF THIS TEST
2298 MOV #TST11,NEXT ;POINT TO THE START OF THE NEXT TEST
2299 MOV DZTCR,R0 ;SET DEVICE ADDRESS
2300 MOV #TCR0,R5 ;SET EXPECTED RESULTS
2301 MOV #1,$LOCK ;SET FOR SW09
2302 1S: MOV R5,(R0) ;SET THE BIT
2303 MOV (R0),R4 ;READ THE BIT FROM THE DEVICE
2304 BIC #1C<377>,R4 ;CLEAR HIGH BYTE
2305 CMP R5,R4 ;WAS BIT OK?
2306 BEQ 2S ;BR IF YES
2307 ERROR 2 ;*BIT FAILED TO SET.
2308 2S: BIC R5,(R0) ;CLEAR THE BIT
2309 MOV (R0),R4 ;READ THE REGISTER
2310 BIC #1C<377>,R4 ;CLEAR HIGH BYTE
2311 TST R4 ;BITS CLEAR?
2312 BEQ 3S ;BR IF YES
2313 MOV R5,-(SP) ;SAVE GOOD RESULTS
2314 CLR R5 ;SET EXPECTED TO 0
2315 ERROR 2 ;*REPORT BIT NOT CLEAR
2316 3S: MOV (SP)+,R5 ;RESTORE R5
2317 MOV R5,(R0) ;SET THE BIT AGAIN.
2318 DEVICE.CLR ;ISSUE DEVICE CLEAR
2319 MOV (R0),R4 ;READ THE REGISTER
2320 BIC #1C<377>,R4 ;CLEAR HIGH BYTE
2321 TST R4 ;BITS CLEAR?
2322 BEQ 4S ;BR IF YES
2323 MOV R5,-(SP) ;SAVE GOOD RESULTS
2324 CLR R5 ;SET EXPECTED TO 0
2325 ERROR 2 ;*REPORT BIT NOT CLEAR
2326 4S: MOV (SP)+,R5 ;RESTORE R5
2327 SCOPE1 ;LOCK ON BIT? SET SW09=1
2328 ASLB R5 ;CHANGE TO NEXT BIT
2329 BNE 1S ;CONTINUE TESTING
2330 CLR LOCK ;MAKE SURE TIGHT LOOP IS CLEANED UP
2331 ;***** TEST 11 *****
2332 ;*THIS TESTS THAT ALL OF THE FOLLOWING
;*BITS CAN BE: SET, CLEARED, CLEARED BY "RESET INSTR *NOT* DEVICE CLEAR "
;*BITS TESTED ARE:

```

M06

```

2333          ;* DTR0, DTR1, DTR2, DTR3, DTR4, DTR5, DTR6, DTR7
2334          ;* THIS TEST IS NOT DONE IF MODULE IS 20MA VERSION
2335          ;* TEST 11
2336          ;*****
2337 013450 000004 000011 001122 001360
2338 013452 012737 000011 001122 001360
2339 013460 012737 013632 002056 001360
2340 013466 013700 002056
2341 013472 012705 000400
2342 013476 012737 013514 001362
2343 013504 105737 001414
2344 013510 100001
2345 013512 104400
2346 013514 010510 1S:
2347 013516 011004
2348 013520 105004
2349 013522 020504
2350 013524 001401
2351 013526 104002
2352 013530 040510 2S:
2353 013532 011004
2354 013534 105004
2355 013536 005704
2356 013540 001404
2357 013542 010546
2358 013544 005005
2359 013546 104002
2360 013550 012605
2361 013552 010510 3S:
2362 013554 104413
2363 013556 011004
2364 013560 105004
2365 013562 030510
2366 013564 001001
2367 013566 104002
2368 013570 104401 4S:
2369 013572 006305
2370 013574 001347
2371 013576 012710 177400
2372 013602 005005
2373 013604 005227 000000 5S:
2374 013610 001375
2375 013612 000005
2376 013614 011004
2377 013616 105004
2378 013620 005704
2379 013622 001401
2380 013624 104002
2381 013626 005037 001362
2382          ;***** TEST 12 *****
2383          ;* THIS TEST PERFORMS RESET TESTING &
2384          ;* TESTING OF WRITE ONLY OR READ ONLY BIT
2385          ;* TEST BITS "RDONE, BIT11, BIT10, BIT9, BIT8, BIT2, BIT1
2386          ;* BIT0, SILOAL" ARE READ ONLY AND THAT TRDY IS
2387          ;* ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.
2388          ;*

```

```

2389
2390
2391 013632 000004
2392 013634 012737 000012 001122
2393 013642 012737 013750 001360
2394 013650 013700 002042
2395 013654 005005
2396 013656 012710 027607
2397
2398 013662 011004
2399 013664 001401
2400 013666 104002
2401 013670 012710 100000
2402 013674 011004
2403 013676 001401
2404 013700 104002
2405 013702 012705 100000
2406 013706 005077 166140
2407 013712 052777 000001 166136
2408 013720 052710 000040
2409 013724 052705 000040
2410 013730 005002
2411 013732 011004
2412 013734 020504
2413 013736 001404
2414 013740 104414
2415 013742 005202
2416 013744 001372
2417 013746 104002
2418 013750
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429 013750 000004
2430 013752 012737 000013 001122
2431 013760 012737 014100 001360
2432 013766 104413
2433 013770 013700 002042
2434 013774 012710 177757
2435 014000 012705 050150
2436 014004 011004
2437 014006 020405
2438 014010 001401
2439 014012 104002
2440 014014 105010
2441 014016 105005
2442 014020 011004
2443 014022 020405
2444 014024 001401

```

```

::* TEST 12
*****
TST12: SCOPE
MOV #12,$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST13,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;SET ADDRESS TO R0
CLR R5 ;SET EXPECTED TO 0
MOV #RDONE+BIT11+BIT10+BIT9+BIT8+BIT2+BIT1+BIT0+$ILOAL,(R0) ;WRITE THE BITS
MOV (R0),R4 ;READ BACK THE BITS
BEQ 1$ ;BR IF NONE ARE SET.
ERROR 2 ;*BITS WERE SET.
MOV #TRDY,(R0) ;ATTEMPT TO WRITE TRDY
MOV (R0),R4 ;READ TRDY
BEQ 2$ ;BR IF NOT SET
ERROR 2 ;*
MOV #TRDY,R5 ;SET EXPECTED BIT
CLR @DZLPA ;LOAD LINE 0
BIS #TCRO,@DZTCR ;SET TCR BIT
BIS #MSENAB,(R0) ;
BIS #MSENAB,R5 ;SET SCAN ENABLE
CLR R2 ;SET COUNTER TO ZERO
MOV (R0),R4 ;READ THE REGISTER
CMP R5,R4 ;BIT SET?
BEQ 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 VERIFY THAT A RESET WILL CLEAR THESE BITS
*THIS TEST ALSO CHECKS BYTE OPERATIONS ON THE CSR

```

```

::* TEST 13
*****
TST13: SCOPE
MOV #13,$TSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST14,NEXT ;POINT TO THE START OF THE NEXT TEST
MOV DZCSR,R0 ;SET UP FOR ERROR MESSAGE
CLR R5 ;TRY TO WRITE
MOV #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
MOV (R0),R4 ;ACTUAL
CMP R4,R5 ;CMP EXPECTED VS ACTUAL
BEQ 1$ ;YES
ERROR 2 ;*NO
CLR (R0) ;CLEAR LOWER BYTE OF CSR
CLRB R5 ;SET EXPECTED
MOV (R0),R4 ;READ CSR BITS
CMP R4,R5 ;COMPARE ACTUAL TO EXPECTED
BEQ 3$ ;BRANCH IF SAME
3$:

```

```

2445 014026 104002          ERROR      2          ; OTHERWISE PRINT ERROR
2446 014030 012710 177757 3$:  MOV      #1C<DCLR>, (R0) ; RESET CSR BITS
2447 014034 105077 166004 CLR      #HDZCSR      ; CLEAR HIGH BYTE OF CSR
2448 014040 012705 000150 MOV      #RIE!MSENAB!MAINT R5 ; SET R5 TO EXPECTED RESULTS
2449                                     ; READ CSR
2450 014044 011004          MOV      (R0), R4      ; ACTUAL = EXPECTED?
2451 014046 020405          CMP      R4, R5      ; BRANCH IF SAME
2452 014050 001401          BEQ      4$          ; OTHERWISE PRINTOUT ERROR
2453 014052 010402          ERROR     2          ; RESET CSR BITS
2454 014054 012710 177757 4$:  MOV      #1C<DCLR>, (R0) ; SET R5 TO EXPECTED RESULTS
2455 014060 005005          CLR      R5          ; DELAY TIMER FOR
2456 014062 005227 000000 5$:  INC      #0          ; ACT-11 COMPATIBILITY
2457 014066 001375          BNE     5$          ; ISSUE BUS INIT
2458 014070 000005          RESET                     ; READ CSR REGISTER
2459 014072 011004          MOV      (R0), R4      ; BRANCH IF CSR IS CLEAR
2460 014074 001401          BEQ      2$          ; IF NOT PRINT ERROR
2461 014076 104002          ERROR     2          ;
2462 014100                                     ;
2463                                     ; ***** TEST 14 *****
2464                                     ; *THIS TEST PERFORMS RESET TESTING AND
2465                                     ; *TESTING OF READ ONLY REGISTER DZRBUF
2466                                     ; *AND TESTING OF WRITE ONLY REGISTER DZLPR
2467                                     ;
2468                                     ; ** TEST 14
2469                                     ; *****
2470 014100 000004          TST14: SCOPE
2471 014102 012737 000014 001122 MOV      #14, $TSTNM ; LOAD THE NUMBER OF THIS TEST
2472 014110 012737 014170 001360 MOV      #TST15, NEXT ; POINT TO THE START OF THE NEXT TEST
2473 014116 104413          DEVICE.CLR ; CLEAR DZ11
2474 014120 013700 002046 MOV      DZRBUF, R0 ; SET UP FOR ERROR MESSAGE
2475 014124 011005          MOV      (R0), R5 ; SET EXPECTED
2476 014126 012777 177777 165716 MOV      #-1, #DZLPR ; TRY TO WRITE ALL 1'S
2477 014134 011004          MOV      (R0), R4 ; ACTUAL
2478 014136 042705 104000 BIC      #DVALID!BIT11, R5 ; DITTO
2479 014142 020405          CMP      R4, R5 ; CMP ACTUAL VS EXPECTED
2480 014144 001401          BEQ      1$          ; IF YES, GO CONTINUE PROCESSING
2481 014146 104002          ERROR     2          ; *ERROR- BIT PATTERN NOT CORRECT
2482 014150 010403 1$:  MOV      R4, R3 ; GET A COPY OF THE ACTUAL BIT PATTERN
2483 014152 005103          COM     R3          ; GET THE LOGICAL INVERSE OF THE BIT PATTERN
2484 014154 010377 165672 MOV      R3, #DZLPR ; TRY TO WRITE
2485 014160 011004          MOV      (R0), R4 ; ACTUAL
2486 014162 020405          CMP      R4, R5 ; CMP ACTUAL VS EXPECTED
2487 014164 001401          BEQ      2$          ; IF YES, GET OUT OF THIS TEST
2488 014166 104002          ERROR     2          ; *NO
2489 014170                                     ;
2490                                     ; ***** TEST 15 *****
2491                                     ; *THIS TEST PERFORMS RESET TESTING AND
2492                                     ; *TESTING OF READ ONLY REGISTER DZMSR
2493                                     ; *AND TESTING OF WRITE ONLY REGISTER DZTDR
2494                                     ;
2495                                     ; ** TEST 15
2496                                     ; *****
2497 014170 000004          TST15: SCOPE
2498 014172 012737 000015 001122 MOV      #15, $TSTNM ; LOAD THE NUMBER OF THIS TEST
2499 014200 012737 014254 001360 MOV      #TST16, NEXT ; POINT TO THE START OF THE NEXT TEST
2500 014206 104413          DEVICE.CLR ; CLEAR DZ11
2501 014210 013700 002062 MOV      DZMSR, R0 ; SET UP FOR ERROR MESSAGE
2502 014214 011005          MOV      (R0), R5 ; SET EXPECTED

```



```

2501 014216 012777 177777 165642      MOV      #-1,@DZTDR      ;TRY TO WRITE ALL 1'S
2502 014224 011004                      MOV      (R0),R4        ;ACTUAL
2503 014226 020405                      CMP      R4,R5         ;CMP ACTUAL VS EXPECTED
2504 014230 001401                      BEQ      1$           ;IF YES,GO CONTINUE PROCESSING
2505 014232 104002                      ERROR    2           ;*ERROR- BIT PATTERN NOT CORRECT
2506 014234 010403                      1$:      MOV      R4,R3      ;GET A COPY OF THE ACTUAL BIT PATTERN
2507 014236 005103                      COM      R3,R3        ;GET THE LOGICAL INVERSE OF THE BIT PATTERN
2508 014240 010377 165622      MOV      R3,@DZTDR      ;TRY TO WRITE
2509 014244 011004                      MOV      (R0),R4        ;ACTUAL
2510 014246 020405                      CMP      R4,R5         ;CMP ACTUAL VS EXPECTED
2511 014250 001401                      BEQ      2$           ;IF YES, GET OUT OF THIS TEST
2512 014252 104002                      ERROR    2           ;*NO
2513 014254                      2$:

```

```

***** TEST 16 *****
*VERIFY THAT IF WE ARE IN "STAGGERED" MODE
*THAT SETTING "DTR" FOR A LINE WILL
*BRING UP "RING" AND "CARRIER" FOR THE
*ASSOCIATED LINE IN WHICH WE ARE STAGGERED!
* LINE1 DTR= LINE1 RING AND CARRIER
* LINE1 DTR= LINE2 RING AND CARRIER
* LINE2 DTR= LINE3 RING AND CARRIER
* LINE3 DTR= LINE 4 RING AND CARRIER
* ETC...

```

```

::* TEST 16
*****

```

```

2529 014254 000004                      TST16:  SCOPE
2530 014256 012737 000016 001122      MOV      #16,$TSTNM     ;LOAD THE NUMBER OF THIS TEST
2531 014264 012737 014450 001360      MOV      #TST17,NEXT    ;POINT TO THE START OF THE NEXT TEST
2532 014272 012737 014344 001362      MOV      #1$,LOCK       ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2533 014300 105737 001414                      TSTB     EIAFLG         ;EIA OR ZOMA?
2534 014304 100001                      BPL      10$           ;BR IF EIA
2535 014306 104400                      ADVANCE  ;EXIT TEST
2536 014310 013700 002062 10$:      MOV      DZMSR,R0       ;SET REGISTER
2537 014314 104413                      DEVICE.CLR ;INIT DZ11
2538 014316 005003                      CLR      R3            ;ZERO LINE NUMBER
2539 014320 012702 000001                      MOV      #1,R2         ;SET POINTER
2540 014324 005737 001370                      TST      MODE          ;ARE WE IN STAGGERED MODE?
2541 014330 100405                      BMI      1$           ;YES WE ARE!
2542 014332 013737 001360 001126      MOV      NEXT,$LPADR    ;LEAVE THIS TEST! NOT STAGGERED
2543 014340 000177 164562                      JMP      @SLPADR       ;EXIT
2544 014344 130237 001364 1$:      BITB     R2,LINE       ;TEST THIS LINE?
2545 014350 001004                      BNE     3$           ;YES
2546 014352 005203                      2$:      INC      R3            ;LINE #
2547 014354 106302                      ASLB    R2            ;GET NEXT LINE
2548 014356 103372                      BCC     1$           ;KEEP TESTING
2549 014360 104400                      ADVANCE ;ADVANCE THIS TEST
2550 014362 010204 3$:      MOV      R2,R4         ;SAVE BINARY BIT FOR LINE #
2551 014364 032703 000001                      BIT      #BIT0,R3      ;GET STAGGERED COMPANION LINE
2552 014370 001402                      BEQ     4$           ;BR IF LINE EVEN
2553 014372 006204                      ASR     R4            ;ADJUST LINE
2554 014374 000401                      BR      5$           ;ADJUST LINE
2555 014376 006304                      4$:      ASL     R4            ;ADJUST LINE
2556 014400 005005                      5$:      CLR     R5            ;SET EXPECTED

```

```

2557 014402 150405      BISB      R4,R5      ;
2558 014404 000305      SWAB      R5      ;
2559 014406 150405      BISB      R4,R5      ;
2560 014410 150277 165444      BISB      R2,2HDZTCR ; SET DTR
2561 014414 104414      DELAY     ; CABLE DELAY
2562 014416 011004      MOV       (R0),R4   ; READ MSR REGISTER
2563 014420 020504      CMP       R5,R4     ; OK?
2564 014422 001401      BEQ       6$        ; YES
2565 014424 104002      ERROR    2          ; *ERROR IN RING OR CARRIER
2566 014426 140277 165426      BICB      R2,2HDZTCR ; CLEAR DTR
2567 014430 104414      DELAY     ; CABLE DELAY
2568 014434 011004      MOV       (R0),R4   ; READ MSR
2569 014436 001402      BEQ       7$        ; BR IF THEY CLEARED
2570 014440 005005      CLR       R5        ; SET EXPECTED TO 0
2571 014442 104002      ERROR    2          ; *BITS NOT CLEARED
2572 014444 104401      SCOPI    2          ; LOCK ON SIGNAL?
2573 014446 000741      BR        2$        ; CONTINUE TEST
2574
2575      ;***** TEST 17 *****
2576      ;*TEST TO VERIFY THAT IF IN "EXTERNAL"
2577      ;*MODE; SETTING DTR FOR SELECTED LINES
2578      ;*WILL BRING UP "CARRIER" AND "RING"
2579      ;*FOR THAT SAME LINE. NOTE: IF YOU HAVE
2580      ;*SELECTED MODE AS "EXTERNAL"; THE H325 TEST CONNECTER
2581      ;*MUST BE USED ON ALL SPECIFIED LINES.
2582      ;*LINES MAY BE SPECIFIED BY SWR03=1
2583      ;*AND SWR00=1 AT START TIME OR ALTERING
2584      ;*STATUS MAP.
2585      ;:* TEST 17
2586      ;*****
2587 014450 000004      TST17:  SCOPE
2588 014452 012737 000017 001122      MOV       #17,$STSTNM ; LOAD THE NUMBER OF THIS TEST
2589 014460 012737 014606 001360      MOV       #20,$NEXT   ; POINT TO THE START OF THE NEXT TEST
2590 014466 012737 014522 001362      MOV       #3,$LOCK    ; USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2591 014474 105737 001370      TSTB     MODE        ; EXTERNAL?
2592 014500 100401      BMI      2$          ; BR IF YES
2593 014502 104400      ADVANCE  2$          ; EXIT TEST
2594 014504 105737 001414      TSTB     EIAFLG      ; YOU BETTER BE IN
2595 014510 100774      BMI      1$          ; EIA MODE FOR THIS TEST.
2596 014512 013700 002062      MOV       DZMSR,R0   ; SET REGISTER
2597 014516 012702 000001      MOV       #1,R2     ; SET LINE POINTER
2598 014522 130237 001364      BITB     R2,LINE     ; LINE SELECTED?
2599 014526 001003      BNE      5$          ; BR IF YES
2600 014530 106302      ASLB     R2          ; NEXT LINE
2601 014532 103373      BCC      3$          ; CONTINUE TEST
2602 014534 104400      ADVANCE  3$          ; ADVANCE THIS TEST
2603 014536 005005      CLR       R5        ; SET EXPECTED
2604 014540 150205      BISB     R2,R5      ;
2605 014542 000305      SWAB     R5          ;
2606 014544 150205      BISB     R2,R5      ;
2607 014546 150277 165306      BISB     R2,2HDZTCR ; SET DTR
2608 014552 104414      DELAY     ; CABLE DELAY
2609 014554 011004      MOV       (R0),R4   ; READ MSR
2610 014556 020504      CMP       R5,R4     ; BITS OK?
2611 014560 001401      BEQ      6$          ; BR IF YES
2612 014562 104002      ERROR    2          ; CARRIER OR RING ERROR

```

E07

MD-11-DZDZA-E MACY11 30(1046) 03-OCT-77 09:43 PAGE 54
DZDZAE.P11 03-OCT-77 09:39

DZ11 DEVICE DIAGNOSTICS.

COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0082

2613 014564 140277 165270
2614 014570 104414
2615 014572 011004
2616 014574 001402
2617 014576 005005
2618 014600 104002
2619 014602 104401
2620 014604 000751
2621
2622
2623
2624
2625
2626
2627
2628
2629 014606 000004
2630 014610 012737 000020 001122
2631 014616 012737 014732 001360
2632 014624 104413
2633 014626 013700 002042
2634 014632 012705 100040
2635 014636 005037 001372
2636 014642 012702 000001
2637 014646 130237 001364
2638 014652 001420
2639 014654 050277 165176
2640 014660 052710 000040
2641 014664 005004
2642 014666 032710 100000
2643 014672 001004
2644 014674 104414
2645 014676 005204
2646 014700 001372
2647 014702 104003
2648 014704 011004
2649 014706 020405
2650 014710 001401
2651 014712 104002
2652 014714 062705 000400
2653 014720 104413
2654 014722 005237 001372
2655 014726 106302
2656 014730 103346
2657 014732
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668

```
6S: BICB R2, @HDZTCR ;CLEAR DTR
      DELAY ;CABLE DELAY
      MOV (R0), R4 ;READ MSR
      BEQ 7S ;BR IF BITS CLEARED
      CLR R5 ;CLEAR EXPECTED LOC.
      ERROR 2 ;BITS NOT CLEARED.
7S: SCOP1 ;LOCK ON LINE?
     BR 4S ;CONTINUE TEST

***** 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 ;LOAD THE NUMBER OF THIS TEST
        MOV #20, STSTNM ;POINT TO THE START OF THE NEXT TEST
        MOV #ST21, NEXT ;ISSUE A "DEVICE CLEAR" (RESET)
        DEVICE.CLR ;SET POINTER
        MOV DZCSR, R0 ;START THE EXPECTED LINE NUMBER AT 0
        MOV #MSENAB!TRDY, R5 ;SET UP FOR ERROR PRINTOUTS
        CLR SAVLIN ;USING R2 AS A BIT POINTER, POINT TO LINE 0
        MOV #1, R2 ;IS THIS LINE SELECTED?
1S: BITB R2, LINE ;IF NO, SKIP THE STARTUP
     BEQ 5S ;SET THE GO BIT FOR THIS LINE
2S: BIS R2, @DZTCR ;START THE SCANNER
     BIS #MSENAB, (R0) ;SET FOR DELAY
     CLR R4 ;TX READY?
3S: BIT #TRDY, (R0) ;BR IF YES
     BNE 4S ;DELAY
     DELAY ;COUNTER
     INC R4 ;BR IF (<)0!
     BNE 3S ;*TX NOT READY!
4S: MOV (R0), R4 ;GET THE LINE POINTED TO BY THE SCANNER
     CMP R4, R5 ;IS THE LINE NUMBER WHAT IT SHOULD BE?
     BEQ 5S ;IF YES, GO WORK ON THE NEXT LINE
     ERROR 2 ;*LINE NUMBER DID NOT MATCH TCR BIT
5S: ADD #400, R5 ;POINT TO THE NEXT EXPECTED LINE
     DEVICE.CLR ;ISSUE A "DEVICE CLEAR" (RESET)
     INC SAVLIN ;ADJUST FOR NEXT LINE
     ASLB R2 ;POINT TO THE NEXT LINE. ARE ALL LINES TESTED?
6S: BCC 1S ;IF NOT, GO DO THE NEXT LINE

***** 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.
::* TEST 21
*****
```

```

2669 014732 000004 TST21: SCOPE
2670 014734 012737 000021 001122 MOV #21,STSTNM ;LOAD THE NUMBER OF THIS TEST
2671 014742 012737 015270 001360 MOV #TST22,NEXT ;POINT TO THE START OF THE NEXT TEST
2672 014750 012737 015246 001362 MOV #16$,LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2673 014756 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2674 014760 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
2675 014764 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
2676 014770 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
2677 014774 001402 BEQ 2$ ;NO
2678 014776 010177 165050 MOV R1,ADZLPR ;SET UP LINE PARAMETERS
2679 015002 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
2680 015004 106302 ASLB R2 ;GOT 'EM ALL ?
2681 015006 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
2682 015010 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
2683 015014 012702 000001 MOV #1,R2 ;LINE POINTER
2684 015020 052777 000040 165014 BIS #MSENAB,ADZCSR ;START SCANNER
2685 015026 030237 001364 3$: BIT R2,LINE ;VALID LINE ?
2686 015032 001462 BEQ 14$ ;NO SET UP NEXT LINE
2687 015034 010277 165016 MOV R2,ADZTCR ;SET TCR BIT
2688 015040 032777 000200 164774 4$: BIT #RDONE,ADZCSR ;IS REC DONE = 0 ?
2689 015046 001401 BEQ 5$ ;IF YES, ALLOW TIME FOR TRDY TO SET
2690 015050 104020 ERROR 20 ;*REC DONE SHOULD = 0
2691 015052 005005 5$: CLR R5
2692 015054 032777 100000 164760 6$: BIT #TRDY,ADZCSR
2693 015062 001004 BNE 7$
2694 015064 104414 DELAY
2695 015066 105205 INCB R5
2696 015070 001371 BNE 6$
2697 015072 104003 ERROR 3 ;*TRDY FAILED TO SET!
2698 015074 112777 000252 164764 7$: MOV #252,ADZTDR ;LOAD CHARACTER
2699 015102 013705 001372 MOV SAVLIN,R5 ;MAKE EXPECTED LINE #
2700 015106 105737 001371 TSTB MODE+1 ;IS THIS TEST IN STAGGERED MODE?
2701 015112 001406 BEQ 10$ ;IF NOT, SKIP STAGGERED SETUP
2702
2703 ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
2704
2705 015114 006205 ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
2706 015116 103402 BCS 8$ ;IF IT IS SET, GO CLEAR IT
2707 015120 000261 SEC ;IF IT IS CLEAR SET IT HERE
2708 015122 000401 BR 9$ ;SKIP THE CLEARING
2709 015124 000241 8$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2710 015126 006105 9$: ROL R5 ;GET THE NEW BIT BACK INTO R5
2711 015130 000305 10$: SWAB R5 ;MOVE THE LINE NUMBER TO THE UPPER BYTE
2712 015132 152705 000252 BISB #252,R5 ;ADD CHARACTER
2713 015136 052705 100000 BIS #DVALID,R5 ;ADD DATA VALID
2714 015142 005003 CLR R3
2715 015144 032777 000200 164670 11$: BIT #RDONE,ADZCSR
2716 015152 001004 BNE 12$
2717 015154 104414 DELAY
2718 015156 005203 INC R3
2719 015160 001371 BNE 11$
2720 015162 104004 ERROR 4 ;*RDONE FAILED TO SET!
2721 015164 017704 164656 12$: MOV ADZRBUF,R4 ;LOAD THE VALUE ACTUALLY RECEIVED
2722 015170 020405 CMP R4,R5 ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
2723 015172 001401 BEQ 13$ ;IF YES, GO DO THE NEXT LINE
2724 015174 104006 ERROR 6 ;*NO DATA/CONTENTS DID NOT COMPARE

```

```

2725 015176 104401 13$: SCOP1 ;CHECK TO SEE IF SWITCH NINE IS SET
2726 015200 040277 164652 14$: BIC R2,ADZTCR ;CLEAR TCR BIT FOR THAT LINE.
2727 015204 005237 001372 15$: INC SAVLIN ;INC EXPECTED LINE
2728 015210 013700 001372 MOV SAVLIN,R0 ;SET UP CHARACTER OFFSET
2729 015214 006300 ASL R0 ;MAKE THE OFFSET A POWER OF TWO
2730 015216 106302 ASLB R2 ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
2731 015220 103302 BCC 3$ ;IF NO, GO AROUND AGAIN FOR NEXT LINE
2732 015222 005003 CLR R3 ;THIS CODE HAS BEEN INSERTED
2733 015224 104414 17$: DELAY ;TO DETECT A PROBLEM FOUND IN FAULT
2734 015226 105203 INCB R3 ;INSERTION. IF AN ERROR OCCURS MORE
2735 015230 001375 BNE 17$ ;THAN ONE WORD WAS RECIEVED ON
2736 015232 032777 000200 164602 BIT #RDONE,ADZCSR ;LINE 7.
2737 015240 001401 BEQ 18$
2738 015242 104020 ERROR 20
2739 015244 104400 18$: ADVANCE ;GO TO NEXT TEST
2740
2741 ;TIGHT SCOPE LOOP FOR THIS TEST. LOOP TRANSMITS CHARACTERS ONLY
2742
2743 015246 032777 100000 164566 16$: BIT #TROY,ADZCSR ;IS TRANSMITTER READY?
2744 015254 001774 BEQ 16$ ;IF NOT, WAIT FOR IT
2745 015256 112777 000252 164602 MOVB #252,ADZTDR ;LOAD THE CHARACTER
2746 015264 104401 SCOP1 ;LOOP AGIN IF SW09=1
2747 015266 000744 BR 14$ ;OTHERWISE, GO PICK UP THE TEST NORMALLY
2748
2749 ;***** TEST 22 *****
2750 ;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
2751 ;* CHARACTERS (FLAG MODE) AND THE RECEIVER RECEIVES (FLAG MODE)
2752 ;* (ONE LINE AT A TIME BASED UPON VALID LINES)
2753 ;* THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
2754
2755 ;:* TEST 22
2756 ;*****
2757 015270 000004 TST22: SCOPE
2758 015272 012737 000022 001122 MOV #22,$TSTNM ;LOAD THE NUMBER OF THIS TEST
2759 015300 012737 015616 001360 MOV #TST23,NEXT ;POINT TO THE START OF THE NEXT TEST
2760 015314 104417 015422 001362 MOV #4$,LOCK ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
2761 015316 013701 001366 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2762 015322 012702 000001 MOV PAR,R1 ;PICK UP PARAMETERS
2763 015326 030237 001364 1$: MOV #1,R2 ;PICK UP INIT POINTER
2764 015332 001402 BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
2765 015334 010177 164512 BEQ 2$ ;NO
2766 015340 005201 2$: INC R1 ;SET UP LINE PARAMETERS
2767 015342 106302 ASLB R2 ;POSITION POINTER TO THE NEXT LINE
2768 015344 103370 BCC 1$ ;GOT 'EM ALL ?
2769 015346 005037 001372 CLR SAVLIN ;IF NO, GO SET UP THE NEXT LINE
2770 015352 012700 001422 MOV #TDO,R0 ;CLEAR LINE # INDICATOR
2771 015356 005020 CLR (R0)+ ;POINT TO THE DATA AREA
2772 015360 022700 001462 CMP #STOP,R0 ;CLEAR A DATA WORD
2773 015364 001374 BNE -6 ;FINISHED ?
2774 015366 005000 CLR R0 ;NO
2775 015370 013737 002046 001400 MOV DZRBUF,REGIST ;CLEAR OFFSET
2776 015376 012702 000001 MOV #1,R2 ;SAVE FOR ERROR MSG
2777 015402 052777 000040 164432 BIS #MSENAB,ADZCSR ;LINE POINTER
2778 015410 030237 001364 3$: BIT R2,LINE ;START SCANNER
2779 015414 001465 BEQ 14$ ;VALID LINE ?
2780 015416 010277 164434 MOV R2,ADZTCR ;NO SET UP NEXT LINE
;SET TCR BIT

```

H07

2781	015422	032777	000200	164412	4\$:	BIT	#RDONE, #DZCSR	; IS REC DONE = 0 ?
2782	015430	001401				BEQ	5\$; IF YES, ALLOW TIME FOR TRDY TO SET
2783	015432	104020				ERROR	20	; *REC DONE SHOULD = 0
2784	015434	005005				CLR	R5	
2785	015436	032777	100000	164376	5\$:	BIT	#TRDY, #DZCSR	
2786	015444	001004				BNE	7\$	
2787	015446	104414				DELAY		
2788	015450	105205				INCB	R5	
2789	015452	001371				BNE	6\$	
2790	015454	104003				ERROR	3	; *TRDY FAILED TO SET!
2791	015456	116077	001422	164402	7\$:	MOVB	T00(R0), #DZTDR	; LOAD CHARACTER
2792	015464	013705	001372			MOV	SAVLIN, R5	; MAKE EXPECTED LINE #
2793	015470	105737	001371			TSTB	MODE+1	; IS THIS TEST IN STAGGERED MODE?
2794	015474	001406				BEQ	10\$; IF NOT, SKIP STAGGERED SETUP
2795								
2796								
2797								
2798	015476	006205						
2799	015500	103402				ASR	R5	; GET THE LAST BIT INTO THE CARRY BIT
2800	015502	000261				BCS	8\$; IF IT IS SET, GO CLEAR IT
2801	015504	000401				SEC		; IF IT IS CLEAR SET IT HERE
2802	015506	000241			8\$:	BR	9\$; SKIP THE CLEARING
2803	015510	006105			9\$:	CLC		; CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2804	015512	000305			10\$:	ROL	R5	; GET THE NEW BIT BACK INTO R5
2805	015514	156005	001422			SWAB	R5	; MOVE THE LINE NUMBER TO THE UPPER BYTE
2806	015520	052705	100000			BISB	T00(R0), R5	; ADD CHARACTER
2807	015524	005003				BIS	#DVALID, R5	; ADD DATA VALID
2808	015526	032777	000200	164306	11\$:	CLR	R3	
2809	015534	001004				BIT	#RDONE, #DZCSR	
2810	015536	104414				BNE	12\$	
2811	015540	005203				DELAY		
2812	015542	001371				INC	R3	
2813	015544	104004				BNE	11\$	
2814	015546	017704	164274		12\$:	ERROR	4	; *RDONE FAILED TO SET!
2815	015552	020405				MOV	#DZRBUF, R4	; LOAD THE VALUE ACTUALLY RECEIVED
2816	015554	001401				CMP	R4, R5	; COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
2817	015556	104006				BEQ	13\$; IF YES, GO DO THE NEXT LINE
2818	015560	104401				ERROR	6	; *NO DATA/CONTENTS DID NOT COMPARE
2819	015562	105260	001422		13\$:	SCOP1		; CHECK TO SEE IF SWITCH NINE IS SET
2820	015566	001315				INCB	T00(R0)	; INCREMENT BINARY PATTERN FOR THIS LINE
2821	015570	040277	164262		14\$:	BNE	4\$; GO 'ROUND AGAIN FOR NEXT CHARACTER
2822	015574	005237	001372		15\$:	BIC	R2, #DZTCR	; CLEAR TCR BIT FOR THAT LINE.
2823	015600	013700	001372			INC	SAVLIN	; INC EXPECTED LINE
2824	015604	006300				MOV	SAVLIN, R0	; SET UP CHARACTER OFFSET
2825	015606	106302				ASL	R0	; MAKE THE OFFSET A POWER OF TWO
2826	015610	103277				ASLB	R2	; SHIFT THE LINE POINTER. ARE WE ALL DONE?
2827	015612	005037	001362			BCC	3\$; IF NO, GO AROUND AGAIN FOR NEXT LINE
2828						CLR	LOCK	; MAKE SURE LOCK IS CLEAR FOR NEXT TEST

```

***** TEST 23 *****
; THIS TEST WILL PROVE THAT EACH RECEIVING LINE CAN
; BE DISABLED BY SETTING THE RCYON BIT TO ZERO
; FOR EACH LINE IN THE LPR REGISTER. IT ALSO
; VERIFIES THAT MASTER CLEAR WILL ZERO DVALID FOR
; CHARACTERS STORED IN THE SILO.

```

::* TEST 23

2828
2829
2830
2831
2832
2833
2834
2835
2836

```

2837          015616 000004          001122 *****
2838          015620 012737 000023          †TST23: SCOPE
2839          015626 012737 016150 001360      MOV      #23,STSTNM      ;LOAD THE NUMBER OF THIS TEST
2840          015634 105037 001420          MOV      #TST24,NEXT   ;POINT TO THE START OF THE NEXT TEST
2841          015640 005037 001372          CLR      DONFLG       ;INITIALIZE FOR FIRST TEST LOOP
2842          015644 104417          CLR      SAVLIN       ;ZERO LINE NO. FOR ERROR REPORT
2843          015646 013701 001366      DCLASM          ;EXECUTE MASTER CLEAR
2844          015652 042701 010000      MOV      PAR,R1       ;STORE DEFAULT PARAMETERS
2845          015656 012702 000001      BIC      #RCVON,R1    ;CLEAR RCVON BIT
2846          015662 010177 164164      1$:      MOV      #1,R2      ;INIT LINE POINTER
2847          015666 106302          2$:      MOV      R1,ADZLPR    ;LOAD LINE PARAMETER REGISTER
2848          015670 103373          INC      R1           ;SET R1 FOR NEXT LINE
2849          015672 012701 000252      ASLB     R2           ;SHIFT R2 TO NEXT LINE
2850          015674 013702 001364      BCC      2$          ;ALL LINES LOADED?
2851          015700 010277 164146      MOV      #252,R1     ;LOAD TRANSMITTING CHARACTER
2852          015704 052777 000040      MOV      LINE,R2     ;COPY ACTIVE LINE BITS
2853          015710 005005          MOV      R2,ADZTCR   ;LOAD TCR BITS
2854          015716 005777 164116      BIS      #MSENAB,ADZCSR ;SET SCANNER
2855          015724 100404          3$:      CLR      R5           ;INIT DELAY COUNTER
2856          015726 104414          4$:      TST      ADZCSR     ;TRDY SET?
2857          015730 005205          BMI     5$           ;IF YES BRANCH
2858          015732 001372          DELAY   R5           ;IF NOT THEN WAIT
2859          015734 104003          INC     R5           ;INCREMENT DELAY COUNTER
2860          015736 117705 164102      BNE     4$          ;DELAY DONE?
2861          015742 012703 000001      ERROR   3           ;IF YES TRDY FAILED TO SET
2862          015746 042705 177770      5$:      MOV      ADZCSR,R5   ;MOVE LINE NO. INTO R5
2863          015752 001403          MOV     #1,R3        ;INIT TCR POINTER
2864          015754 106303          BIC     #1<7>,R5     ;ISOLATE LINE NO.
2865          015756 005305          BEQ     21$         ;IF LINE 0 GO TEST TRANSM. FLAG
2866          015760 001375          20$:   ASLB     R3        ;POINT R3 TO NEXT TCR BIT
2867          015762 030302          DEC     R5           ;DECREMENT R5 UNTIL R3 POINTS
2868          015764 001007          BNE     20$         ;TO CORRECT TCR BIT
2869          015766 140377 164064      21$:   BIT      R3,R2     ;HAS THIS LINE BEEN SERVICED?
2870          015772 001351          BNE     6$           ;IF NOT GO SEND CHARACTER
2871          015774 105737 001420      BICB   R3,ADZTCR    ;IF YES CLEAR TCR BIT
2872          016000 001040          BNE     3$          ;IF MORE LINES SET BRANCH
2873          016002 000404          TSTB   DONFLG       ;IF ALL LOADED IS THIS SECOND PASS
2874          016004 110177 164056      BR     7$           ;IF YES BRANCH TO SECOND PART OF TEST
2875          016010 040302          BR     7$           ;OTHERWISE CONTINUE WITH FIRST PART
2876          016012 000741          MOV    R1,ADZTDR    ;TRANSMIT CHARACTER
2877          016014 005077 164036      BIC    R3,R2        ;CLEAR FLAG FOR THIS LINE
2878          016020 005005          BR     3$          ;GO WAIT FOR NEXT LINE
2879          016022 104414          CLR    ADZTCR       ;CLEAR TCR BITS
2880          016024 005205          CLR    R5           ;CLEAR DELAY COUNTER
2881          016026 001375          DELAY  R5           ;WAIT FOR LAST CHARACTER
2882          016030 105777 164006      INC    R5           ;INCREMENT DELAY COUNTER
2883          016034 100003          BNE    8$          ;IF NOT FINISHED CONTINUE WAITING
2884          016036 005037 001372      TSTB  ADZCSR        ;RDONE BIT SET?
2885          016042 104020          BPL    10$         ;IF NO CONTINUE
2886          016044 017704 163776      CLR    SAVLIN       ;IF YES SET LINE NO. TO ZERO
2887          016050 100007          ERROR  20          ;AND PRINT ERROR
2888          016052 000304          10$:  MOV    ADZRBUF,R4 ;READ SILO
2889          016054 042704 177770      BPL    11$         ;IF DVALID IS ZERO BRANCH
2890          016060 010437 001372      SWAB  R4            ;IF SET THEN
2891          016064 010437          BIC    #1<7>,R4     ;ISOLATE LINE NO. IN R4
2892          016066 010437          MOV    R4,SAVLIN   ;SET SAVLIN FOR ERROR REPORT

```

```

2893 016064 104017          ERROR 17          ;DATA VALID SHOULD NOT BE SET
2894 016066 000766          BR 10$           ;GO READ SILO AGAIN
2895 016070 105237 001420 11$: INCB DONFLG      ;PREPARE FOR SECOND PART OF TEST
2896 016074 013701 001366  MOV PAR,R1      ;MOVE DEFAULT PARAMETERS TO R1
2897 016100 000666          BR 1$           ;GO LOAD LPR REGISTER
2898 016102 005005          CLR R5          ;INIT DELAY COUNTER
2899 016104 104414          12$: DELAY      ;WAIT FOR LAST CHARACTER
13$:          ;TO BE RECEIVED
2900 016106 005205          INC R5          ;DELAY FINISHED?
2901 016110 001375          BNE 13$        ;IF YES EXECUTE MASTER CLEAR
2902 016112 104413          DEVICE.CLR
2903 016114 000240          NOP
2904 016116 000240          NOP
2905 016120 105777 163716  TSTB 2DZCSR     ;RDONE SET?
2906 016124 100003          BPL 14$        ;IF NOT BRANCH
2907 016126 005037 001372  CLR SAVLIN     ;IF YES THEN PRINT OUT
2908 016132 104020          ERROR 20       ;REPORT
2909 016134 017704 163706 14$: MOV 2DZRBUF,R4 ;READ SILO
2910 016140 100003          BPL 15$        ;DATA VALID SET?
2911 016142 005037 001372  CLR SAVLIN     ;IF YES THEN PRINT OUT
2912 016146 104017          ERROR 17       ;ERROR REPORT
2913 016150
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926 016150 000004          ;***** TEST 24 *****
2927 016152 012737 000024 001122 ;*THIS TEST WILL PROVE THAT:
2928 016160 012737 016426 001360 ;* 1) THE TRANSMITTER "BREAK BIT" WORKS
2929 016166 012737 016264 001362 ;* 2) THE RECEIVER CAN FLAG "FRAMING ERRORS"
2930 016174 005737 001370 ;* 3) THE RECEIVER CAN FLAG "PARITY ERRORS"
2931 016200 001510          ;*ONLY ONE LINE AT A TIME WILL BE EXERCISED.
2932 016202 104417          ;*THIS TEST WILL NOT BE EXERCISED UNLESS
2933 016204 013701 001366 ;*CONNECTED BY AN H325, H3271, OR H3190 CONNECTOR
2934 016210 052701 000300 ;:* TEST 24
2935 016214 012700 000001 ;*****
2936 016220 030037 001364 ;*ST24: SCOPE
2937 016224 001402          MOV #24,$STNM   ;LOAD THE NUMBER OF THIS TEST
2938 016226 010177 163620 1$: MOV $ST25,NEXT ;POINT TO THE START OF THE NEXT TEST
2939 016232 005201 2$: INC R1          ;SET FOR LOOP
2940 016234 106300          TST MODE       ;ARE WE RUNNING IN INTERNAL MODE?
2941 016236 103370          BEQ 12$        ;IF SO, SKIP THIS TEST
2942 016240 005037 001372  DCLASM         ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2943 016244 012702 000001  MOV PAR,R1     ;PICK UP PARAMETERS
2944 016250 052777 000040 163564 ;BIS #ODDPAR,PARITY,R1 ;FORCE ODD PARITY
2945 016256 013737 002046 001400  MOV #1,R0      ;PICK UP INIT POINTER
2946 016264 030237 001364 3$: BIT R0,LINE  ;SHOULD THIS LINE BE SET UP ?
2947 016270 001446          BEQ 2$         ;IF NOT DON'T SET IT UP
2948 016272 010277 163560 2$: MOV R1,2DZLPR ;OTHERWISE, SET UP LINE PARAMETERS
          ASLB R0  ;GOT 'EM ALL ?
          BCC 1$  ;NO
          CLR SAVLIN ;CLEAR LINE #
          MOV #1,R2  ;LINE POINTER
          BIS #MSENAB,2DZCSR ;SET MASTER SCAN ENABLE
          MOV DZRBUF,REGIST ;SAVE FOR ERRR MESSAGE
          BIT R2,LINE
          BEQ 11$
          MOV R2,2DZTCR ;SET TCR BIT
          MOVB R2,2HDZTDR ;SET BREAK BIT

```



```

2949 016302 112777 000377 163556 4$: MOVB #377,ADZTDR ;LOAD CHARACTER
2950 016310 013705 001372 MOV SAVLIN,R5 ;MAKE EXPECTED DATA
2951 016314 105737 001371 TSTB MODE+1 ;IS THIS TEST IN STAGGERED MODE?
2952 016320 001406 BEQ 7$ ;IF NOT, SKIP STAGGERED SETUP
2953
2954 ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
2955
2956 016322 006205 ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
2957 016324 103402 BCS 5$ ;IF IT IS SET, GO CLEAR IT
2958 016326 000261 SEC ;IF IT IS CLEAR SET IT HERE
2959 016330 000401 BR 6$ ;SKIP THE CLEARING
2960 016332 000241 5$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2961 016334 006105 6$: ROL R5 ;GET THE NEW BIT BACK INTO R5
2962 016336 000305 7$: SWAB R5 ;PUT LINE NUMBER IN UPPER BYTE
2963 016340 052705 130000 BIS #DVALID!PARER!FMERR,R5 ;ADD EXPECTED
2964 016344 005004 CLR R4
2965 016346 032777 000200 163466 8$: BIT #RDONE,ADZCSR
2966 016354 001004 BNE 9$
2967 016356 104414 DELAY
2968 016360 005204 INC R4
2969 016362 001371 BNE 8$
2970 016364 104004 ERROR 4 ;#RDONE FAILED TO SET!
2971 016366 017704 163454 9$: MOV ADZRBUF,R4 ;ACTUAL
2972 016372 020405 CMP R4,R5 ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
2973 016374 001401 BEQ 10$ ;IF YES, GO CLEAN UP
2974 016376 104006 ERROR 6 ;#DATA/CONTENTS FAILED TO COMPARE
2975 016400 105077 163464 10$: CLRB ADZTDR ;CLEAR BREAK BITS
2976 016404 104401 SCOP1 ;LOOP?
2977 016406 005237 001372 11$: INC SAVLIN ;INC LINE #
2978 016412 040277 163440 BIC R2,ADZTCR ;CLEAR TCR BIT
2979 016416 106302 ASLB R2
2980 016420 103321 BCC 3$
2981 016422 005037 001362 12$: CLR LOCK ;MAKE SURE LOCK IS CLEAR FOR NEXT TEST
2982
2983 ;***** TEST 25 *****
2984 ;* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
2985 ;*WHILE THE PROCESSOR STATUS IS SET EXACTLY
2986 ;*TO WHAT THE DZ11 PRIORITY IS SET TO.
2987 ;*DEFAULT PRIORITY IS AT 5 (240).
2988 ;::* TEST 25
2989 ;*****
2990 016426 000004 TST25: SCOPE
2991 016430 012737 000025 001122 MOV #25,$STSTM ;LOAD THE NUMBER OF THIS TEST
2992 016436 012737 016736 001360 MOV #TST26,NEXT ;POINT TO THE START OF THE NEXT TEST
2993 016444 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
2994 016446 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
2995 016452 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
2996 016456 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
2997 016462 001402 BEQ 2$ ;NO
2998 016464 010177 163362 MOV R1,ADZLPR ;SET UP LINE PARAMETERS
2999 016470 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
3000 016472 106302 ASLB R2 ;GOT 'EM ALL ?
3001 016474 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
3002 016476 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
3003 016502 106437 027060 MTPS ADZPRT ;SET CPU STATUS TO DZ11 PRIO,
3004 016514 113777 001364 163342 3$: MOVB LINE,ADZTCR ;ENABLE THE VALID LINES

```

```

3005 016514 012777 016604 163354      MOV      #6$,DZTIV      ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3006 016522 012777 016612 163342      MOV      #7$,DZRIV      ;SET UP THE RECEIVER INTERRUPT VECTOR
3007 016530 013777 027060 163336      MOV      DZPRT,DZCRIS   ;SET THE INTERRUPT VECTOR STATUS
3008 016536 013777 027060 163334      MOV      DZPRT,DZTIS    ;SET TRANSMITTER INTERRUPT PRIORITY
3009 016544 052777 040040 163270      BIS      #TIE!MSENAB,DZCSR ;ENABLE THE DEVICE
3010 016552 005005                                CLR      R5
3011 016554 032777 100000 163260 4$:      BIT      #TRDY,DZCSR
3012 016562 001403                                BEQ      5$
3013 016564 000240                                NOP
3014 016566 000240                                NOP
3015 016570 000412                                BR       8$
3016 016572 104414                                5$:     DELAY
3017 016574 005205                                INC      R5
3018 016576 001366                                BNE     4$
3019 016600 104003                                ERROR   3          ;*TRDY NOT SET!
3020 016602 000405                                BR       8$
3021 016604 104010                                6$:     ERROR   10          ;*TRANSMITTER SHOULD NOT INTERRUPT
3022 016606 022626                                CMP     (SP)+,(SP)+ ;POP FOR FAKE RTI
3023 016610 000402                                BR       8$          ;CONTINUE TEST
3024 016612 104012                                7$:     ERROR   12          ;*RECEIVER SHOULD NOT INTERRUPT
3025 016614 022626                                CMP     (SP)+,(SP)+ ;POP FOR FAKE RTI
3026 016616 042777 040000 163216 8$:     BIC     #TIE,DZCSR    ;RESET TRANSMITTER INTERRUPT ENABLE
3027 016624 012777 016722 163244      MOV      #11$,DZTIV     ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3028 016632 012777 016730 163232      MOV      #12$,DZRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
3029 016640 013777 027060 163226      MOV      DZPRT,DZCRIS   ;SET THE INTERRUPT VECTOR STATUS
3030 016646 013777 027060 163224      MOV      DZPRT,DZTIS    ;SET TRANSMITTER INTERRUPT PRIORITY
3031 016654 052777 000140 163160      BIS      #RIE!MSENAB,DZCSR ;ENABLE THE DEVICE
3032 016662 113777 001422 163176      MOV     TD0,DZTDR      ;PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
3033 016670 005005                                CLR      R5
3034 016672 032777 000200 163142 9$:     BIT      #RDONE,DZCSR
3035 016700 001403                                BEQ     10$
3036 016702 000240                                NOP
3037 016704 000240                                NOP
3038 016706 000412                                BR      13$
3039 016710 104414                                10$:    DELAY
3040 016712 005205                                INC     R5
3041 016714 001366                                BNE     9$
3042 016716 104004                                ERROR   4          ;*NO RX DONE! (NOT SET)
3043 016720 000405                                BR      13$          ;CONTINUE TEST
3044 016722 104010                                11$:    ERROR   10          ;*TRANSMITTER SHOULD NOT INTERRUPT
3045 016724 022626                                CMP     (SP)+,(SP)+ ;POP FOR FAKE RTI
3046 016726 000402                                BR      13$          ;CONT TEST
3047 016730 104012                                12$:    ERROR   12          ;*RECEIVER SHOULD NOT INTERRUPT
3048 016732 022626                                CMP     (SP)+,(SP)+ ;POP FOR FAKE RTI
3049 016734                                13$:
3050 016734 104413      DEVICE.CLR      ;ISSUE DEVICE CLEAR (RESET)
3051                                ;***** TEST 26 *****
3052                                ;* THIS TEST VERIFIES THAT THE DEVICE DOES INTERRUPT
3053                                ;*WHILE THE PROCESSOR STATUS IS SET TO EXACTLY
3054                                ;*ONE LEVEL LOWER THAN THE DZ11. DZ11 PRIORITY
3055                                ;*DEFAULT TO LEVEL 5 MINUS ONE LEVEL IS LEVEL 4.
3056                                ;:* TEST 26
3057                                ;*****
3058 016736 000004      †ST26:  SCOPE
3059 016740 012737 000026 001122      MOV     #26,$STNM      ;LOAD THE NUMBER OF THIS TEST
3060 016746 012737 017264 001360      MOV     #TST27,NEXT    ;POINT TO THE START OF THE NEXT TEST

```

3061	016754	104417			DCLASM		; CLEAR DEVICE AND SET MAINT BIT IF I MODE
3062	016756	013701	001366		MOV	PAR,R1	; PICK UP PARAMETERS
3063	016752	012702	000001		MOV	#1,R2	; PICK UP INIT POINTER
3064	016756	030237	001364	1S:	BIT	R2,LINE	; SHOULD THIS LINE BE SET UP ?
3065	016772	001402			BEQ	2S	; NO
3066	016774	010177	163052		MOV	R1,ADZLPR	; SET UP LINE PARAMETERS
3067	017000	005201		2S:	INC	R1	; POSITION POINTER TO THE NEXT LINE
3068	017002	106302			ASLB	R2	; GOT 'EM ALL ?
3069	017004	103370			BCC	1S	; IF NO, GO SET UP THE NEXT LINE
3070	017006	005037	001372		CLR	SAVLIN	; CLEAR LINE # INDICATOR
3071	017012	106437	027062		MTPS	#LESS1	; MAKE CPU ONE LEVEL LOWER THAN DZ11
3072	017016	113777	001364	163032	MOVB	LINE,ADZTCR	; ENABLE THE VALID LINES
3073	017024			3S:			
3074	017024	012777	017116	163044	MOV	#6S,ADZTIV	; SET UP THE TRANSMITTER INTERRUPT VECTOR
3075	017032	012777	017134	163032	MOV	#7S,ADZRIV	; SET UP THE RECEIVER INTERRUPT VECTOR
3076	017040	013777	027060	163026	MOV	DZPRT,ADZRIS	; SET THE INTERRUPT VECTOR STATUS
3077	017046	013777	027060	163024	MOV	DZPRT,ADZTIS	; SET TRANSMITTER INTERRUPT PRIORITY
3078	017054	052777	040040	162760	BIS	#TIE!MSENAB,ADZCSR	; ENABLE THE DEVICE
3079	017062	005005			CLR	R5	
3080	017064	032777	100000	162750	BIT	#TRDY,ADZCSR	
3081	017072	001404			BEQ	5S	
3082	017074	000240			NOP		
3083	017076	000240			NOP		
3084	017100	104007			ERROR	7	; *TRANSMITTER FAILED TO INTERRUPT
3085	017102	000416			BR	8S	
3086	017104	104414		5S:	DELAY		
3087	017106	005205			INC	R5	
3088	017110	001365			BNE	4S	
3089	017112	104003			ERROR	3	; *TRDY NOT SET!
3090	017114	000411			BR	8S	
3091	017116	022626		6S:	POP2SP		; REMOVE THE INTERRUPT FROM THE STACK
3092	017120	042777	040000	162714	BIC	#TIE,ADZCSR	; DON'T LET ANY MORE INTERRUPTS OCCUR
3093	017126	106437	027062		MTPS	#LESS1	; MAKE CPU ONE LEVEL LOWER THAN DZ11
3094	017132	000402			BR	8S	; RETURN TO THE NORMAL FLOW
3095	017134	104012		7S:	ERROR	12	; *RECEIVER SHOULD NOT INTERRUPT
3096	017136	022626			CMP	(SP)+,(SP)+	; POP FOR FAKE RTI
3097	017140	042777	040000	162674	BIC	#TIE,ADZCSR	; RESET TRANSMITTER INTERRUPT ENABLE
3098	017146	012777	017246	162722	MOV	#11S,ADZTIV	; SET UP THE TRANSMITTER INTERRUPT VECTOR
3099	017154	012777	017254	162710	MOV	#12S,ADZRIV	; SET UP THE RECEIVER INTERRUPT VECTOR
3100	017162	013777	027060	162704	MOV	DZPRT,ADZRIS	; SET THE INTERRUPT VECTOR STATUS
3101	017170	013777	027060	162702	MOV	DZPRT,ADZTIS	; SET TRANSMITTER INTERRUPT PRIORITY
3102	017176	052777	000140	162636	BIS	#TIE!MSENAB,ADZCSR	; ENABLE THE DEVICE
3103	017204	113777	001422	162654	MOVB	TDO,ADZTDR	; PUT ANY RANDOM CHARACTER IN TRANSMITTER BUFFER
3104	017212	005005			CLR	R5	
3105	017214	032777	000200	162620	BIT	#RDONE,ADZCSR	
3106	017222	001404			BEQ	10S	
3107	017224	000240			NOP		
3108	017226	000240			NOP		
3109	017230	104011			ERROR	11	; *RECEIVER FAILED TO INTERRUPT
3110	017232	000413			BR	13S	
3111	017234	104414		10S:	DELAY		
3112	017236	005205			INC	R5	
3113	017240	001365			BNE	9S	
3114	017242	104004			ERROR	4	; *NO RX DONE! (NOT SET)
3115	017244	000406			BR	13S	; CONTINUE TEST
3116	017246	104010		11S:	ERROR	10	; *TRANSMITTER SHOULD NOT INTERRUPT

```

3117 017250 022626
3118 017252 000403
3119 017254 022626
3120 017256 005077 162560
3121 017262
3122 017262 104413
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134 017264 000004
3135 017266 012737 000027 001122
3136 017274 012737 017716 001360
3137 017302 104417
3138 017304 013701 001366
3139 017310 012702 000001
3140 017314 030237 001364
3141 017320 001402
3142 017322 010177 162524
3143 017326 005201
3144 017330 106302
3145 017332 103370
3146 017334 005037 001372
3147 017340 012777 017570 162524
3148 017346 013777 027060 162520
3149 017354 012777 017660 162514
3150 017362 013777 027060 162510
3151 017370 052777 000040 162444
3152 017376 012702 000001
3153 017402 030237 001364
3154 017406 001004
3155 017410 005237 001372
3156 017414 106302
3157 017416 000771
3158 017420 106427 000340
3159 017424 000240
3160 017426 000240
3161 017430 110277 162422
3162 017434 005777 162406
3163 017440 100001
3164 017442 104017
3165 017444 105777 162372
3166 017450 100001
3167 017452 104020
3168 017454 005005
3169 017456 005004
3170 017460 005777 162356
3171 017464 100404
3172 017466 104414

```

```

CMP (SP)+,(SP)+ ;POP FOR FAKE RTI
BR 13$ ;CONT TEST
12$: POP2SP ;REMOVE THE INTERRUPT FROM THE STACK
CLR @DZCSR ;DON'T ALLOW ANY MORE INTERRUPTS
13$: DEVICE.CLR ;ISSUE DEVICE CLEAR (RESET)

***** TEST 27 *****
;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 27
*****
TST27: SCOPE
MOV #27,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST30,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,@DZLPR ;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,@DZRIS
MOV #12,@DZTIV
MOV DZPRT,@DZTIS
BIS #MSENAB,@DZCSR
MOV #1,R2 ;LINE POINTER
3$: BIT R2,LINE ;VALID LINE ?
BNE 4$
INC SAVLIN
ASLB R2
BR 3$
4$: MTPS #PR7
NOP
NOP
MOV R2,@DZTCR ;SET TCR BIT
TST @DZRBUF ;VALID DATA?
BPL .+4 ;IT BETTER NOT BE SET
ERROR 17 ;DATA VALID SHOULD NOT BE SET
5$: TSTB @DZCSR ;RECEIVER DONE ?
BPL .+4
ERROR 20 ;RECEIVER DONE BIT SHOULD NOT BE SET
CLR R5
CLR R4
99$: TST @DZCSR ;WAIT FOR TRDY
BMI 100$ ;BR IF READY
DELAY ;STALL TIME

```

```

3173 017470 005204 INC R4 ;
3174 017472 001372 BNE 99$ ;
3175 017474 104003 ERROR 3 ;TRDY FAILED TO SET
3176 017476 105077 162364 100$: CLRB @DZTDR
3177 017502 005004 CLR R4
3178 017504 032777 000200 162330 6$: BIT #RDONE,@DZCSR
3179 017512 001004 BNE 7$
3180 017514 104414 DELAY
3181 017516 005204 INC R4
3182 017520 001371 BNE 6$
3183 017522 104004 ERROR 4 ;*RDONE FAILED TO SET!
3184 017524 005777 162312 7$: TST @DZCSR ;TRANS DONE BIT = 1 ?
3185 017530 100401 BMI +4 ;YES
3186 017532 104003 ERROR 3 ;*NO TRANS DONE FAILED TO SET
3187 ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
3188 ;SET INTERRUPT ENABLES AND WATCH THE FUR FLY
3189 017534 052777 040000 162300 BIS #TIE,@DZCSR
3190 017542 052777 000100 162272 BIS #RIE,@DZCSR
3191 017550 106427 000000 MTPS #0
3192 017554 000240 NOP
3193 017556 000240 NOP
3194 017560 104007 ERROR 7 ;*TRANSMITTER FAILED TO INTERRUPT
3195 017562 104011 ERROR 11 ;*RECEIVER FAILED TO INTERRUPT
3196 ;CHECK BR LEVEL
3197 017564 000137 017664 JMP 13$ ;GET OUT
3198
3199 ;RECEIVER INTERRUPT ROUTINE
3200 017570 017704 162252 8$: MOV @DZRBUF,R4 ;ACTUAL
3201 017574 010403 MOV R4,R3
3202 017576 000303 SWAB R3
3203 017600 042703 177770 BIC #1C<7>,R3 ;STRIP JUNK
3204 017604 105737 001371 TSTB MODE+1 ;IS THIS TEST IN STAGGERED MODE?
3205 017610 001406 BEQ 11$ ;IF NOT, SKIP STAGGERED SETUP
3206
3207 ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3208
3209 017612 006203 ASR R3 ;GET THE LAST BIT INTO THE CARRY BIT
3210 017614 103402 BCS 9$ ;IF IT IS SET, GO CLEAR IT
3211 017616 000261 SEC ;IF IT IS CLEAR SET IT HERE
3212 017620 000401 BR 10$ ;SKIP THE CLEARING
3213 017622 000241 9$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3214 017624 006103 10$: ROL R3 ;GET THE NEW BIT BACK INTO R3
3215 017626 020337 001372 11$: CMP R3,SAVLIN ;IS THIS A VALID LINE
3216 017632 001401 BEQ +4 ;YES
3217 017634 104015 ERROR 15 ;*INVALID LINE
3218 017636 042704 177400 BIC #1C<377>,R4 ;STRIP JUNK
3219 017642 120504 CMPB R5,R4 ;DATA COMPARE ?
3220 017644 001401 BEQ +4 ;YES
3221 017646 104005 ERROR 5 ;*DATA DOES NOT COMPARE
3222 017650 040277 162202 BIC R2,@DZTCR ;CLEAR TCR BIT
3223 017654 022626 POP2SP ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
3224 017656 000402 BR 13$ ;GO GET OUT OF INTERRUPT MODE
3225 ;TRANSMITTER INTERRUPT SVC ROUTINE
3226 017660 104011 12$: ERROR 11 ;THE RECEIVER INTERRUPT FAILED
3227 ;TO OVERRIDE THE TRANSMITTER
3228 017662 022626 POP2SP ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
    
```

```

3229 017664 042777 040100 162150 13$: BIC #TIE!RIE,ADZCSR ;CLEAR INTERRUPT ENABLES
3230 017672 013777 002074 162172 MOV DZBIS,ADZRIV ;RESTORE TRAPCATCHER
3231 017700 005077 162170 CLR ADZRIS
3232 017704 013777 002100 162164 MOV DZTIS,ADZTIV
3233 017712 005077 162162 CLR ADZTIS
3234 ;***** TEST 30 *****
3235 ;*TEST TO VERIFY THAT 'RDONE DOES NOT SET
3236 ;*IF THE SCANNER IS DISABLED.
3237 ;*TURN ON SCANNER, WAIT FOR TRDY.
3238 ;*TURN OFF SCANNER, TRANSMIT A CHARACTER
3239 ;*'RDONE SHOULD NOT SET.
3240 ;::* TEST 30
3241 ;*****
3242 017716 000004 ;*ST30: SCOPE
3243 017720 012737 000030 001122 MOV #30,$STSTM ;LOAD THE NUMBER OF THIS TEST
3244 017726 012737 020104 001360 MOV #TST31,NEXT ;POINT TO THE START OF THE NEXT TEST
3245 017734 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3246 017736 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
3247 017742 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
3248 017746 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
3249 017752 001402 BEQ 2$ ;NO
3250 017754 010177 162072 MOV R1,ADZLPR ;SET UP LINE PARAMETERS
3251 017760 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
3252 017762 106302 ASLB R2 ;GOT 'EM ALL ?
3253 017764 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
3254 017766 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
3255 017772 052777 000040 162042 BIS #MSENAB,ADZCSR ;TURN ON SCANNER
3256 020000 012702 000001 MOV #1,R2 ;INIT LINE COUNTER
3257 020004 030237 001364 3$: BIT R2,LINE ;FIND A VALID LINE
3258 020010 001004 BNE 4$ ;IF WE FOUND ONE GO TO TEST
3259 020012 005237 001372 INC SAVLIN ;IF NOT
3260 020016 106302 ASLB R2 ;KEEP LOOKING
3261 020020 000771 BR 3$
3262 020022 110277 162030 4$: MOVB R2,ADZTCR ;SET TCR BIT
3263 020026 005005 CLR R5
3264 020030 005777 162006 5$: TST ADZCSR ;IS TRDY SET
3265 020034 100404 BMI 6$ ;CON'T TESTING IF IT IS
3266 020036 104414 DELAY ;IF IT NOT WAIT A WHILE
3267 020040 005205 INC R5
3268 020042 001372 BNE 5$
3269 020044 104003 ERROR 3 ;WE WAITED LONG ENOUGH-ERROR
3270 020046 042777 000040 161766 6$: BIC #MSENAB,ADZCSR ;TURN OFF SCANNER
3271 020054 105077 162006 CLR ADZTOR ;TRANSMIT A CHARACTER
3272 020060 005005 CLR R5 ;CLEAR COUNTER
3273 020062 104414 7$: DELAY ;WAIT SUFFICIENT TIME FOR
3274 020064 005205 INC R5 ;RDONE TO SET
3275 020066 001375 BNE 7$
3276 020070 032777 000200 161744 BIT #RDONE,ADZCSR ;RDONE SET
3277 020076 001401 BEQ 8$ ;IT SHOULDN'T BE-CONTINUE
3278 020100 104020 ERROR 20 ;IF IT IS THERE'S AN ERROR
3279 020102 104400 8$: ADVANCE
3280 ;***** TEST 31 *****
3281 ;*THIS TEST VERIFIES OVERRUN AND SILO ALARM
3282 ;*ONE LINE AT A TIME - BASED UPON VALID LINES
3283 ;*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
3284 ;*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN

```

```

3285                                     ;*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
3286                                     ;*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
3287                                     ;*CHAR PULLED OUT OUT THE SILO.
3288                                     ;*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
3289                                     ;*ON DZ LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
3290                                     ;*USED TO SCOPE SILO ALARM PULSES, ETC.
3291                                     ;::* TEST 31
3292                                     ;*****
3293 020104 000004                                †TST31: SCOPE
3294 020106 012737 000031 001122                MOV     #31,STSTNM      ;LOAD THE NUMBER OF THIS TEST
3295 020114 012737 020632 001360                MOV     #TST32,NEXT    ;POINT TO THE START OF THE NEXT TEST
3296 020122 012737 020536 001362                MOV     #18$,LOCK      ;SET FOR LOOP
3297 020130 104417                                DCLASM                 ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3298 020132 013701 001366                MOV     PAR,R1         ;PICK UP PARAMETERS
3299 020136 012702 000001                MOV     #1,R2         ;PICK UP INIT POINTER
3300 020142 030237 001364                1$:    BIT     R2,LINE   ;SHOULD THIS LINE BE SET UP ?
3301 020146 001402                                BEQ     2$,            ;NO
3302 020150 010177 161676                MOV     R1,ADZLPR      ;SET UP LINE PARAMETERS
3303 020154 005201 2$:    INC     R1         ;POSITION POINTER TO THE NEXT LINE
3304 020156 106302                                ASLB    R2             ;GOT 'EM ALL ?
3305 020160 103370                                BCC    1$,            ;IF NO, GO SET UP THE NEXT LINE
3306 020162 005037 001372                CLR     SAVLIN         ;CLEAR LINE # INDICATOR
3307 020166 012700 001422                MOV     #TDO,RO       ;POINT TO THE DATA AREA
3308 020172 005020                                CLR     (RO)+         ;CLEAR A DATA WORD
3309 020174 022700 001462                CMP     #STOP,RO      ;FINISHED ?
3310 020200 001374                                BNE    -6             ;NO
3311 020202 005000                                CLR     RO            ;CLEAR OFFSET
3312 020204 012702 000001                MOV     #1,R2         ;LINE POINTER
3313 020210 052777 010040 161624                BIS     #MSENAB!SILOEN,ADZCSR ;START SCANNER & SET SILO ENABLE
3314 020216 030237 001364                3$:    BIT     R2,LINE   ;VALID LINE?
3315 020222 001002                                BNE    +6             ;YES
3316 020224 000137 020520                JMP     22$,          ;TRY NEXT LINE
3317 020230 013700 001372                MOV     SAVLIN,RO     ;MAKE OFFSET
3318 020234 006300                                ASL     RO            ;MAKE POWER OF TWO
3319 020236 010277 161614                MOV     R2,ADZTCR     ;SET TCR BIT
3320 020242 105777 161574                4$:    TSTB   ADZCSR    ;REC DONE = 1 ?
3321 020246 100001                                BPL    +4             ;REC DONE SHOULD NOT = 1
3322 020250 104020                                ERROR  20             ;SET CHARACTER COUNT
3323 020252 005003                                CLR     R3            ;
3324 020254 005004                                CLR     R4            ;
3325 020256 032777 100000 161556                5$:    BIT     #TRDY,ADZCSR
3326 020264 001004                                BNE    7$,            ;
3327 020266 104414                                DELAY                 ;
3328 020270 105204                                INCB   R4             ;
3329 020272 001371                                BNE    6$,            ;
3330 020274 104003                                ERROR  3              ;
3331 020276 116077 001422 161562                7$:    MOVB   TDO(RO),ADZTDR ;*TRDY FAILED TO SET
3332 020304 005260 001422                INC     TDO(RO)      ;LOAD A CHARACTER
3333 020310 020327 000017                CMP     R3,#15.     ;SET UP NEXT CHARACTER
3334 020314 103006                                BHIS   8$,          ;16 CHARACTERS ?
3335 020316 032777 020000 161516                BIT     #SILOAL,ADZCSR ;SILO ALARM = 0 ?
3336 020324 001401                                BEQ    +4             ;YES
3337 020326 104013                                ERROR  13             ;*SILO ALARM SHOULD NOT = 1
3338                                     ;UNTIL 16. DATA CHARACTERS
3339 020330 000411                                BR     10$,          ;
3340 020332 005004                8$:    CLR     R4

```

```

3341 020334 032777 020000 161500 9$: BIT #SILOAL,ADZCSR
3342 020342 001004 BNE 10$
3343 020344 104414 DELAY
3344 020346 005204 INC R4
3345 020350 001371 BNE 9$
3346 020352 104014 ERROR 14 ;#SILO ALARM FAILED TO SET!
3347 ;SILO ALARM SHOULD =1 AFTER 16.
3348 ;DATA CHARACTERS
3349 020354 005203 10$: INC R3 ;INC CHAR COUNT
3350 020356 022703 000102 CMP #66.,R3 ;FINISHED SENDING CHARACTERS ?
3351 020362 001334 BNE 5$ ;NO
3352 020364 005004 CLR R4
3353 020366 104414 DELAY
3354 020370 105204 INCB R4
3355 020372 001375 BNE -4 ;NOW LETS READ THE SILO
3356 ;MOV SAVLIN,R5 ;MAKE EXPECTED LINE #
3357 020374 013705 001372 MOV SAVLIN,R5 ;IS THIS TEST IN STAGGERED MODE?
3358 020400 105737 001371 TSTB MODE+1 ;IF NOT, SKIP STAGGERED SETUP
3359 020404 001406 BEQ 13$
3360 ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3361
3362
3363 020406 006205 ASR R5 ;GET THE LAST BIT INTO THE CARRY BIT
3364 020410 103402 BCS 11$ ;IF IT IS SET, GO CLEAR IT
3365 020412 000261 SEC ;IF IT IS CLEAR SET IT HERE
3366 020414 000401 BR 12$ ;SKIP THE CLEARING
3367 020416 000241 11$: CLC ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3368 020420 006105 12$: ROL R5 ;GET THE NEW BIT BACK INTO R5
3369 020422 000305 13$: SWAB R5 ;PUT IN UPPER BYTE
3370 020424 052705 100000 BIS #DVALID,R5 ;ADD DATA VALID
3371 020430 017704 161412 14$: MOV ADZRBUF,R4 ;ACTUAL
3372 020434 020405 CMP R4,R5 ;ACTUAL VS. EXPECTED
3373 020436 001401 BEQ 15$ ;YES
3374 020440 104006 ERROR 6 ;#DATA/CONTENTS DID NOT COMPARE
3375 020442 032777 020000 161372 15$: BIT #SILOAL,ADZCSR ;SILO ALARM= 0 ?
3376 020450 001401 BEQ 16$ ;YES
3377 020452 104016 ERROR 16 ;READING DZRBUF DID NOT CLEAR SILO ALARM
3378 020454 005205 16$: INC R5 ;UP CHARACTER
3379 020456 120527 000077 CMPB R5,#63. ;LAST SILO CHAR ?....64TH CHAR
3380 020462 101762 BLOS 14$
3381 020464 005205 INC R5 ;ADD 1 MORE FOR THE CLOBBERED CHAR
3382 020466 052705 040000 BIS #OVRUN,R5 ;ADD OVERRUN TO EXPECTED
3383 020472 120527 000101 CMPB R5,#65. ;LAST CHARACTER ?
3384 020476 001754 BEQ 14$
3385 020500 017704 161342 MOV ADZRBUF,R4 ;FOR GOOD MEASURE
3386 020504 005704 TST R4 ;DATA VALID SHOULD = 0
3387 020506 100001 BPL 17$ ;YES
3388 020510 104017 ERROR 17 ;DATA VALID SHOULD = 0
3389 020512 040277 161340 17$: BIC R2,ADZTCR ;CLR TCR BIT
3390 020516 104401 SCOP1 LOOP?
3391 020520 005237 001372 22$: INC SAVLIN ;INC EXPECTED LINE
3392 020524 106302 ASLB R2 ;NEXT LINE
3393 020526 103402 BCS +6 ;NO
3394 020530 000137 020216 JMP 3$ ;YES
3395 020534 104400 ADVANCE ;GO TO NEXT TEST
3396

```


;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.

```

3397
3398
3399
3400
3401 020536 052777 010040 161276 18$: BIS #MSENAB!SILOEN,ADZCSR ;SETUP DEVICE
3402 020544 012777 020622 161324 MOV #20$,ADZTIV ;SETUP TRANSMITTER VECTOR
3403 020552 012737 000024 001216 MOV #20,$TMPD ;TEMPORARY COUNT OF CHARACTER BURST
3404 020560 050277 161272 BIS R2,ADZTCR ;ENABLE LINE
3405 020564 052777 040000 161250 BIS #TIE,ADZCSR ;ENABLE INTERRUPTS
3406 020572 106427 000000 MTPS #0 ;LOWER PRIORITY
3407 020576 000001 19$: WAIT ;ALLOW INTERRUPTS
3408 020600 005337 001216 DEC $TMPD ;REDUCE COUNT. ALL CHARACTERS SENT?
3409 020604 001374 BNE 19$ ;IF NO, WAIT FOR MORE
3410 020606 042777 050040 161226 BIC #SILOEN!MSENAB!TIE,ADZCSR ;RESET SILO COUNTER, CLEAR STROBE
3411 020614 104401 SCOP1 ;LOOP AGAIN?
3412 020616 000137 020512 JMP 17$ ;IF NOT, RETURN TO WHERE YOU LEFT OFF
3413 020622 112777 000252 161236 20$: MOV #252,ADZTDR ;SEND A CHARACTER
3414 020630 000002 RTI ;ALLOW MORE CHARACTERS TO COME

```

***** TEST 32 *****
;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 32

```

3421
3422
3423 020632 000004 †ST32: SCOPE
3424 020634 012737 000032 001122 MOV #32,$STSTM ;LOAD THE NUMBER OF THIS TEST
3425 020642 012737 021214 001360 MOV #TST33,NEXT ;POINT TO THE START OF THE NEXT TEST
3426 020650 012737 020736 001362 MOV #3$,LOCK ;SET FOR LOOP
3427 020656 104417 DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3428 020660 013701 001366 MOV PAR,R1 ;PICK UP PARAMETERS
3429 020664 012702 000001 MOV #1,R2 ;PICK UP INIT POINTER
3430 020670 030237 001364 1$: BIT R2,LINE ;SHOULD THIS LINE BE SET UP ?
3431 020674 001402 BEQ 2$ ;NO
3432 020676 010177 161150 MOV R1,ADZLPR ;SET UP LINE PARAMETERS
3433 020702 005201 2$: INC R1 ;POSITION POINTER TO THE NEXT LINE
3434 020704 106302 ASLB R2 ;GOT 'EM ALL ?
3435 020706 103370 BCC 1$ ;IF NO, GO SET UP THE NEXT LINE
3436 020710 005037 001372 CLR SAVLIN ;CLEAR LINE # INDICATOR
3437 020714 012700 001422 MOV #TDD,RO ;POINT TO THE DATA AREA
3438 020720 005020 CLR (RO)+ ;CLEAR A DATA WORD
3439 020722 022700 001462 CMP #STOP,RO ;FINISHED ?
3440 020726 001374 BNE -6 ;NO
3441 020730 005000 CLR RO ;CLEAR OFFSET
3442 020732 012702 000001 MOV #1,R2 ;LINE POINTER
3443 020736 012777 021156 161126 3$: MOV #11$,ADZRIV ;SET FOR UNEXPECTED INTER.
3444 020744 012777 000340 161122 MOV #PR7,ADZRIS ;SET PRIO.
3445 020752 052777 010140 161062 BIS #MSENAB!SILOEN!RIE,ADZCSR ;START SCANNER & SET SILO ENABLE
3446
3447 020760 030237 001364 BIT R2,LINE ;VALID LINE?
3448 020764 001002 BNE +6 ;YES
3449 020766 000137 021174 JMP 22$ ;TRY NEXT LINE
3450 020772 005777 161050 TST ADZRBUFF ;EMPTY THE SILO
3451 020776 100775 BMI -4 ;BR IF DATA VALID IS SET!
3452 021000 106427 000000 MTPS #0 ;SET PROCESSOR PRIORITY TO 0

```

3453	021004	013700	001372			MOV	SAVLIN,R0	;MAKE OFFSET
3454	021010	006300				ASL	R0	;MAKE POWER OF TWO
3455	021012	010277	161040			MOV	R2,ADZTCR	;SET TCR BIT
3456	021016	005004			5\$:	CLR	R4	
3457	021020	032777	100000	161014	6\$:	BIT	#TRDY,ADZCSR	
3458	021026	001004				BNE	7\$	
3459	021030	104414				DELAY		
3460	021032	005204				INC	R4	
3461	021034	001371				BNE	6\$	
3462	021036	104003				ERROR	3	;#TRDY FAILED TO SET
3463	021040	116077	001422	161020	7\$:	MOVB	TDO(R0),ADZTDR	;LOAD A CHARACTER
3464	021046	005260	001422			INC	TDO(R0)	;SET UP NEXT CHARACTER
3465	021052	022760	000017	001422		CMP	#15.,TDO(R0)	;15 CHARS YET?
3466	021060	001406				BEQ	8\$	
3467	021062	032777	020000	160752		BIT	#SILOAL,ADZCSR	;SILO ALARM = 0 ?
3468	021070	001401				BEQ	+4	;YES
3469	021072	104013				ERROR	13	;#SILO ALARM SHOULD NOT = 1
3470								;UNTIL 16. DATA CHARACTERS
3471	021074	000750				BR	5\$	
3472	021076	012777	021164	160766	8\$:	MOV	#12\$,ADZRIV	;SET NEW VECTOR
3473	021104	032777	100000	160730		BIT	#TRDY,ADZCSR	;READY FOR 16TH CHAR
3474	021112	001774				BEQ	-6	
3475	021114	016077	001422	160744		MOV	TDO(R0),ADZTDR	;LOAD THE 16TH CHAR.
3476	021122	005004				CLR	R4	
3477	021124	032777	020000	160710	9\$:	BIT	#SILOAL,ADZCSR	
3478	021132	001005				BNE	10\$	
3479	021134	104414				DELAY		
3480	021136	005204				INC	R4	
3481	021140	001371				BNE	9\$	
3482	021142	104014				ERROR	14	;#SILO ALARM FAILED TO SET!
3483	021144	000410				BR	17\$;SILO ALARM SHOULD =1 AFTER 16.
3484								;DATA CHARACTERS
3485	021146	000240			10\$:	NOP		;STALL
3486	021150	000240				NOP		
3487	021152	104000				ERROR		
3488	021154	000404				BR	17\$;SILO ALARM NOT INTERRUPTING.
3489	021156	022626			11\$:	CMP	(SP)+,(SP)+	;CONTINUE TEST.
3490	021160	104012				ERROR	12	;FAKE RTI
3491	021162	000401				BR	17\$;RX SHOULD NOT INTERRUPT
3492	021164	022626			12\$:	CMP	(SP)+,(SP)+	;CONTINUE
3493	021166	040277	160664		17\$:	BIC	R2,ADZTCR	;GOOD INTERRUPT TO HERE.
3494	021172	104401				SCOP1		;CLR TCR BIT
3495	021174	005237	001372		22\$:	INC	SAVLIN	;LOOP?
3496	021200	106302				ASLB	R2	;INC EXPECTED LINE
3497	021202	103402				BCS	+6	;NEXT LINE
3498	021204	000137	020736			JMP	3\$;NO
3499	021210	005037	001362			CLR	LOCK	;YES
								;CLEAR TIGHT LOOP FOR NEXT TEST

```

3500
3501
3502
3503
3504
3505
3506
3507 021214 000004
3508 021216 012737 000033 001122
3509 021224 012737 022022 001360
3510 021232 104417
3511 021234 013737 001364 022020
3512 021242 013701 001366
3513 021246 012700 000001
3514 021252 030037 001364
3515 021256 001402
3516 021260 010177 160566
3517 021264 005201
3518 021266 106300
3519 021270 103370
3520 021272 012700 001422
3521 021276 005020
3522 021300 022700 001462
3523 021304 001374
3524 021306 012777 021542 160556
3525 021314 012777 000340 160552
3526 021322 012777 021444 160546
3527 021330 012777 000340 160542
3528 021336 052777 000100 160476
3529 021344 052777 040000 160470
3530 021352 052777 000040 160462
3531 021360 113777 001364 160470
3532 021366 106437 027062
3533
3534
3535 021372 005037 021442
3536 021376 013727 006722
3537 021402 000000
3538 021404 005337 021402
3539 021410 001375
3540 021412 105737 022020
3541 021416 001002
3542 021420 000137 021720
3543 021424 005237 021442
3544 021430 001362
3545 021432 104007
3546 021434 104011
3547 021436 000137 021772
3548 021442 000000
3549
3550
3551 021444 005777 160372
3552 021450 100401
3553 021452 104003
3554 021454 117703 160364
3555

```

```

***** TEST 33 *****
*THIS TEST RUNS ALL LINES FULL BORE
*BASED UPON QUALIFIED LINES
*..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
*TRANSMITTER
::* TEST 33
*****
TST33: SCOPE
MOV #33,$STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #TST34,NEXT ;POINT TO THE START OF THE NEXT TEST
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV LINE,RXTCR ;SET IMAGE OF TCR BITS
RSTART: MOV PAR,R1 ;PICK UP PARAMETER
MOV #1,R0 ;PICK UP INIT POINTER
INIT: BIT R0,LINE ;SHOULD THIS LINE BE SET UP
;NO
;SET UP LINE PARAM REGISTER
IS: INC R1
ASLB R0 ;GOT 'EM ALL ?
BCC INIT ;NO
MOV #TDO,R0 ;CLEAR TRANS DATA POINTER & REC POINTERS
INIT1: CLR (R0)+
CMP #STOP,R0 ;FINISHED ?
BNE INIT1 ;NO CONTINUE CLEARING
MOV #RXSVC,$DZCRIV ;SET UP REC INTR VECTOR
MOV #PR7,$DZCRIS ;STATUS
MOV #TXSVC,$DZCTIV ;SET UP TRANS INTR VECTOR
MOV #PR7,$DZCTIS ;STATUS
BIS #RIE,$DZCSR ;SET REC INTR ENABLE
BIS #TIE,$DZCSR ;SET TRANS INTR ENABLE
BIS #MSENAB,$DZCSR ;SET MASTER SCAN ENABLE
MOVB LINE,$DZTCR ;SET TCR BITS...UP UP AND AWAY !
MTPS #LESS1 ;ALLOW INTERRUPTS

SNAP: CLR 66$
67$: MOV DLYCNT,(PC)+ ;SET FOR DELAY
68$: 0
DEC 68$
BNE -4
TSTB RXTCR ;WAIT FOR ALL RECIEVERS TO FINISH
BNE 3$
JMP OUT
3$: INC 66$
BNE 67$
ERROR 7 ;*TRANSMITTER FAILED TO INTERRUPT
ERROR 11 ;*RECEIVER FAILED TO INTERRUPT
JMP FINI
66$: 0

;TRANS INTR SVC ROUTINE
TXSVC: TST $DZCSR ;TRANS INTR ?
BMI +4
ERROR 3 ;*TRANSMITTER FAILED
MOVB $DZCSR,R3 ;SAVE IT
;NOW TEST FOR LINE # ETC

```

3556	021460	042703	177770		BIC	#1C<7>,R3	:STRIP JUNK
3557	021464	010304			MOV	R3,R4	:SAVE
3558	021466	010337	001372		MOV	R3,SAVLIN	:ADJUST LOCATION FOR ERROR PRINTOUT
3559	021472	012702	000001		MOV	#1,R2	:SET UP POSITION POINTER
3560	021476	105303		3\$:	DECB	R3	:IS IT THIS LINE ?
3561	021500	100402			BMI	4\$:YES
3562	021502	006302			ASL	R2	:UP THE LINE #
3563	021504	000774			BR	3\$:GO 'ROUND AGAIN
3564	021506	030237	001364	4\$:	BIT	R2,LINE	:VALID LINE?
3565	021512	001001			BNE	.+4	:YES
3566	021514	104010			ERROR	10	:NO,INVALID LINE!!!!
3567	021516	006304			ASL	R4	:MAKE POWER OF 2
3568	021520	116477	001422	160340	MOVB	TDO(R4),@DZTDR	:LOAD CHARACTER
3569	021526	105264	001422		INCB	TDO(R4)	:SET UP NEXT CHARACTER
3570	021532	001002			BNE	5\$:LAST CHARACTER ?
3571	021534	040277	160316		BIC	R2,@DZTCR	:YES,CLEAR TCR BIT
3572	021540	000002		5\$:	RTI		
3573							
3574							
3575							
3576	021542	105777	160274		:REC INTR SVC ROUTINE		
3577	021546	100401			RXSVC: TSTB	@DZCSR	:REC DONE ?
3578	021550	104004			BMI	.+4	:YES
3579	021552	017704	160270		ERROR	4	:FALSE INTERRUPT
3580	021556	010403			MOV	@DZRBUF,R4	:SAVE IT
3581	021560	000303			MOV	R4,R3	
3582	021562	042703	177770		SWAB	R3	
3583	021566	010337	001372		BIC	#1C<7>,R3	:STRIP JUNK
3584	021572	032777	020000	160242	MOV	R3,SAVLIN	:SAVE LINE NUMBER
3585	021600	001401			BIT	#510AL,@DZCSR	:SILO ALARM?
3586	021602	104000			BEQ	.+4	:NO
3587	021604	005704			ERROR		:SILO ALARM SHOULD NOT =1
3588	021606	100401			TST	R4	:DATA VALID SET?
3589	021610	104023			BMI	.+4	:YES
3590	021612	032704	070000		ERROR	23	:YOU LOSE ...DATA VALID WAS'NT SET
3591	021616	001401			BIT	#OVRUN!FMERR!PARER,R4	
3592	021620	104000			BEQ	.+4	
3593	021622	012702	000001		ERROR		:RECEIVER ERROR FLAG/S WERE SET
3594	021626	105303		5\$:	MOV	#1,R2	:SET UP POSITION POINTER
3595	021630	100402			DECB	R3	
3596	021632	006302			BMI	6\$	
3597	021634	000774			ASL	R2	:RE POSITION POINTER
3598	021636	030237	001364	6\$:	BR	5\$:GO 'ROUND AGAIN
3599	021642	001001			BIT	R2,LINE	:LINE VALID ?
3600	021644	104011			BNE	.+4	:YES
3601	021646	013703	001372		ERROR	11	:INVALID LINE #
3602	021652	006303			MOV	SAVLIN,R3	:GET THE LINE NUMBER AGAIN
3603	021654	126304	001442		ASL	R3	:USE R3 AS A POINTER IN THE DATA TABLE
3604	021660	001405			CMPB	TRO(R3),R4	:DOES THE DATA CHARACTER COMPARE ?
3605	021662	016305	001442		BEQ	2\$:YES
3606	021666	042704	177400		MOV	TRO(R3),R5	:SAVE EXPECTED
3607					BIC	#1C<377>,R4	:CLEAR JUNK
3608							:R2 = LINE # BY BIT POSITION
3609							:R4 = ACTUAL DATA
3610	021672	104005			ERROR	5	:R5 = EXPECTED DATA
3611	021674	005263	001442	2\$:	INC	TRO(R3)	:*NO, DATA DOES NOT COMPARE :SET UP FOR NEXT CHARACTER

```

3612 021700 105763 001442      TSTB   TRO(R3) ;ALL CHARS DONE?
3613 021704 001002              BNE    +6
3614 021706 040237 022020      BIC    R2,RXTCR ;ZERO LINE DONE INDICATOR.
3615 021712 012716 021372      MOV    #SNAP,(SP) ;RESET THE BACKGROUND TIMING LOOP
3616 021716 000002              RTI

```

```

3617
3618
3619
3620 021720 106427 000340      OUT:   ;FINISH UP ROUTINE
      MTPS  #PR7 ;STOP ALL INTERRUPTS
      DEVICE.CLR ;CLEAR ALL INTERRUPTS AWAY

```

```

3621 021724 104413
3622 021726 005003
3623 021730 005037 001372      CLR    R3
3624 021734 012702 000001      CLR    SAVLIN
3625 021740 030237 001364      MOV    #1,R2

```

```

3626 021744 001405
3627 021746 022763 000400 001442      1S:   BIT    R2,LINE ;VALID LINE ?
      BEQ    2S ;NO
      CMP    #400,TRO(R3) ;RECEIVED A BINARY COUNT PATTERN ?
      BEQ    +4 ;YES
      ERROR  27 ;THE LINE FAILED TO RECEIVE A FULL
                ;BINARY COUNT PATTERN

```

```

3630
3631 021760 005237 001372      2S:   INC    SAVLIN
      TST   (R3)+ ;SET UP FOR NEXT LINE
      ASLB  R2 ;ADD 2
      BCC  1S ;SET UP NEXT LINE POINTER
                ;FINISHED ?

```

```

3635 021772
3636 021772 013777 002074 160072      FINI: MOV    DZRI$ ,DZRIV ;RESTORE TRAPCATCHER
3637 022000 005077 160070      CLR   DZRI$
3638 022004 013777 002100 160064      MOV   DZTI$ ,DZTIV
3639 022012 005077 160062      CLR   DZTI$

```

```

3640 022016 104400
3641 022020 000000      RXTCR: ADVANCE ;GO TO THE NEXT TEST
      D ;RX IMAGE OF TCR BITS

```

```

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
3667

```

```

***** TEST 34 *****
;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 34
;*****
†ST34: SCOPE
      MOV   #34,$TSTNM ;LOAD THE NUMBER OF THIS TEST
      MOV   #2,$TIMES
      MOV   #TST35,NEXT ;POINT TO THE START OF THE NEXT TEST
      MOV   #3,$LOCK ;SET FOR LOOP
      CLR   OFFSET ;RESET THIS VARIABLE

```

3668	022060	005037	001372			CLR	SAVLIN	:RESET LINE NUMBER INDICATOR
3669	022064	005037	001374			CLR	XMTLIN	:USE THIS WORD TO TELL WHAT LINE TRANSMITTED
3670	022070	012737	000001	001216		MOV	#1,STMP0	:USE STMP0 AS A BIT POINTER
3671	022076	012737	010070	022530		MOV	#RCVON!SSO!EIGHT	:TWOSTOP,7\$:BUILD TEMPORARY PARAMETERS
3672	022104	033737	001216	001364	1\$:	BIT	STMP0,LINE	:IS THIS LINE ACTIVE?
3673	022112	001027				BNE	3\$:IF SO, GO GET STARTED
3674	022114	012737	010070	022530	2\$:	MOV	#RCVON!SSO!EIGHT	:TWOSTOP,7\$:LOAD PARAMETERS TEMPORARILY
3675	022122	012700	001422			MOV	#TDO,RO	:POINT TO THE DATA AREA
3676	022126	005020				CLR	(RO)+	:CLEAR A DATA WORD
3677	022130	022700	001462			CMP	#STOP,RO	:FINISHED ?
3678	022134	001374				BNE	.-6	:NO
3679	022136	005237	001374			INC	XMTLIN	:POINT TO THE NEXT LINE TO TRANSMIT
3680	022142	042737	000007	022530		BIC	#7,7\$:MAKE SURE TEMPORARY PARAMETERS POINT TO 0
3681	022150	053737	001374	022530		BIS	XMTLIN,7\$:ADD DESIRED LINE NUMBER
3682	022156	005037	023766			CLR	OFFSET	
3683	022162	106337	001216			ASLB	STMP0	:POINT TO THE NEXT LINE
3684	022166	103346				BCC	1\$:PROCESS THE NEXT LINE
3685	022170	104400				ADVANCE		:TEST TO SEE IF THIS TEST GETS REPEATED
3686	022172				3\$:			
3687	022172	104417				DCLASM		:CLEAR DEVICE AND SET MAINT BIT IF I MODE
3688	022174	042737	010000	022530		BIC	#RCVON,7\$:ZERO PARAMTERS FOR TX LINE
3689	022202	013777	022530	157642		MOV	7\$,ADZLPR	:LOAD PARAMTERS FOR TX
3690	022210	005737	001370			TST	MODE	:STAGGERED?
3691	022214	100011				BPL	100\$:BR IF NO
3692	022216	000241				CLC		:SET UP LINE
3693	022220	006037	022530			ROR	7\$	
3694	022224	103002				BCC	98\$:BR IF LINE WAS EVEN
3695	022226	000241				CLC		:PREPARE TO MKE LINE EVEN
3696	022230	000401				BR	99\$:CONTINUE
3697	022232	000261			98\$:	SEC		:PREPARE TO MAKE LINE ODD
3698	022234	006137	022530		99\$:	ROL	7\$:SET ALTERED LINE
3699	022240	052737	010000	022530	100\$:	BIS	#RCVON,7\$:SET RX ON
3700	022246	013777	022530	157576		MOV	7\$,ADZLPR	:LOAD RX PARAMETERS
3701	022254	013737	022530	001372		MOV	7\$,SAVLIN	:ADJUST LOCATION FOR ERROR PRINTOUT
3702	022262	042737	177770	001372		BIC	#1C<7>,SAVLIN	:STRIP JUNK
3703	022270	042737	000007	022530		BIC	#7,7\$:CLEAR OLD LINE #
3704	022276	053737	001374	022530		BIS	XMTLIN,7\$:SET LINE UP AGAIN
3705	022284	013737	022530	001400		MOV	7\$,REGIST	:SAVE PARAMETERS FOR PRINTOUT
3706	022292	012700	001422			MOV	#TDO,RO	:POINT TO THE DATA AREA
3707	022296	005020				CLR	(RO)+	:CLEAR A DATA WORD
3708	022300	022700	001462			CMP	#STOP,RO	:FINISHED ?
3709	022324	001374				BNE	.-6	:NO
3710	022326	005002				CLR	R2	:USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
3711	022330	005003				CLR	R3	:USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
3712	022332	005037	001220			CLR	STMP1	:INITIALIZE THE TIMER
3713	022336	005037	001224			CLR	STMP3	:INITIALIZE THESE BITS ALSO
3714	022342	012737	000020	001376		MOV	#20,XMTCNT	:SET HOW MANY CHARACTERS TO TRANSMIT
3715	022350	012777	023410	157520		MOV	#XMTSRV,ADZTIV	
3716	022356	012777	023554	157506		MOV	#RXISR1,ADZRIV	
3717	022364	013777	027060	157502		MOV	DZPRT,ADZRIS	
3718	022372	013777	027060	157500		MOV	DZPRT,ADZTIS	
3719	022400	113777	001216	157450		MOVB	STMP0,ADZTCR	:START THE VALID LINE
3720	022406	052777	040140	157426		BIS	#TIE!RIE!MSENAB,ADZCSR	
3721	022414	106427	000000			MTPS	#0	:LOWER THE PRIORITY TO ALLOW INTERRUPTS
3722	022420	032777	000100	157414	4\$:	BIT	#RIE,ADZCSR	:IS ROUTINE DONE?
3723	022426	001407				BEQ	5\$:WHEN ALL IS DONE RX IE IS CLEARED IN ISR.

```

3724 022430 005237 001220      INC      STMP1      ;COUNT TIME
3725 022434 001371              BNE      4$        ;CONTINUE TEST
3726 022436 105237 001224      INCB     STMP3      ;DOUBLE COUNT
3727 022442 001366              BNE      4$        ;CONTINUE TEST
3728 022444 104011              ERROR    11        ;INTERRUPTS NOT FINISHED
3729 022446 004737 007360      JSR      PC,SERV.G ;<TG>?
3730 022452 104401              SCOP1                    ;LOOP?
3731 022454 062737 000002 023766      ADD      #2,OFFSET
3732 022462 013700 022530      MOV      7$,R0
3733 022466 042700 170377      BIC      #17*400,R0
3734 022472 022700 007400      CMP      #17*400,R0
3735 022476 001010              BNE      6$
3736 022500 032737 000030 022530      BIT      #BIT4+BIT3,7$
3737 022506 001602              BEQ      2$
3738 022510 162737 000010 022530      SUB      #BIT3,7$
3739 022516 000625              BR       3$
3740 022520 062737 000400 022530 6$:      ADD      #400,7$
3741 022526 000621              BR       3$
3742 022530 000000              7$:      0
3743                                     ;***** TEST 35 *****
3744                                     ;* THIS TEST VERIFIES THAT EVEN PARITY WORKS
3745                                     ;* FOR ALL ODD LINES SELECTED AND THAT ODD PARITY WORKS FOR ALL
3746                                     ;* EVEN LINES SELECTED.
3747                                     ;*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
3748                                     ;*THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
3749                                     ;*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
3750                                     ;*YOU ARE IN "STAGGERED" MODE.
3751                                     ;*40(8) CHARS ARE USED FOR THIS TEST.
3752                                     ;*ALL SELECTED LINES WILL BE ENABLED
3753                                     ;*AT THE SAME TIME!
3754                                     ;:* TEST 35
3755                                     ;*****
3756 022532 000004      TST35:  SCOPE
3757 022534 012737 000035 001122      MOV      #35,STSTNM ;LOAD THE NUMBER OF THIS TEST
3758 022542 012737 022772 001360      MOV      #TST36,NEXT ;POINT TO THE START OF THE NEXT TEST
3759 022550 005737 001370      TST      MODE        ;IS THIS STAGGERED MODE?
3760 022554 100105      BPL      6$          ;IF NOT, DON'T DO THIS TEST
3761 022556 104417      DCLASM                    ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
3762 022560 013701 001366      MOV      PAR,R1      ;USE R1 TO BUILD PARAMETERS TO BE LOADED
3763 022564 042701 000200      BIC      #ODDPAR,R1 ;MAKE SURE ODD PARITY ISN'T SET
3764 022570 052701 000100      BIS      #PARITY,R1 ;MAKE SURE PARITY IS TURNED ON
3765 022574 012702 000001      MOV      #1,R2       ;USE R2 AS A LINE POINTER
3766 022600 030237 001364      1$:      BIT      R2,LINE     ;IS THIS A VALID LINE?
3767 022604 001411      BEQ      3$          ;IF NOT, SKIP TO THE NEXT LINE
3768 022606 032701 000001      BIT      #BIT0,R1   ;IS THIS LINE AN ODD LINE?
3769 022612 001002      BNE      2$          ;IF IT'S ODD, USE EVEN PARITY
3770 022614 052701 000200      BIS      #ODDPAR,R1 ;IF IT'S EVEN, USE ODD PARITY
3771 022620 010177 157226      2$:      MOV      R1,ADZLPR  ;LOAD THE LINE PARAMETER REGISTER
3772 022624 042701 000200      BIC      #ODDPAR,R1 ;SET UP THE NEXT PARITY TO EVEN
3773 022630 005201      3$:      INC      R1         ;POINT TO THE NEXT LINE
3774 022632 106302      ASLB    R2          ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
3775 022634 103361      BCC     1$          ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
3776 022636 005037 001372      CLR     SAVLIN     ;CLEAR THE LINE NUMBER INDICATOR
3777 022642 005002      CLR     R2         ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
3778 022644 005003      CLR     R3         ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
3779 022646 012737 000040 001376      MOV     #40,XMTCNT ;TRANSMIT A BINARY COUNT PATTERN(00-40)

```

```

3780 022654 012700 001422      MOV      #TDO,RO      ;POINT TO THE DATA AREA
3781 022660 005020              CLR      (RO)+        ;CLEAR A DATA WORD
3782 022662 022700 001462      CMP      #STOP,RO    ;FINISHED ?
3783 022666 001374              BNE     -6           ;NO
3784 022670 005000              CLR      RO          ;CLEAR OFFSET
3785 022672 012777 023410 157176  MOV      #XMTSRV,ADZTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3786 022700 012777 023232 157164  MOV      #PARESE,ADZRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
3787 022706 013777 027060 157160  MOV      DZPRT,ADZRIS  ;SET THE INTERRUPT VECTOR STATUS
3788 022714 013777 027060 157156  MOV      DZPRT,ADZTIS  ;SET TRANSMITTER INTERRUPT PRIORITY
3789 022722 052777 040140 157112  BIS      #RIE!TIE!MSENAB,ADZCSR ;ENABLE THE DEVICE
3790 022730 113777 001364 157120  MOV     B LINE,ADZTCR ;ENABLE ALL SELECTED LINES
3791 022736 106427 000000              MTPS     #0          ;ALLOW INTERRUPTS
3792 022742 032777 000100 157072 5$:  BIT      #RIE,ADZCSR  ;WHEN RX DONE; RIE WILL =0
3793 022750 001407              BEQ     6$          ;BR IF ALL DONE
3794 022752 005237 023404              INC     COUNTO
3795 022756 102771              BVS     5$
3796 022760 105237 023406              INCB   COUNT1
3797 022764 100366              BPL     5$
3798 022766 104011              ERROR  11          ;*RX FAILED TO FINISH (INTERRUPT)
3799 022770 104400              ADVANCE ;ADVANCE LOOP
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
022772 000004
022774 012737 000036 001122
023002 012737 004562 001360
023010 005737 001370
023014 100105
023016 104417
023020 013701 001366
023024 042701 000200
023030 052701 000100
023034 012702 000001
023040 030237 001364 1$:
023044 001411
023046 032701 000001
023052 001402
023054 052701 000200
023060 010177 156766 2$:
023064 042701 000200
023070 005201 3$:
023072 106302
023074 103361
023076 005037 001372
023102 005002
023104 005003
023106 012737 000040 001376
    
```

```

***** TEST 36 *****
*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!
::* TEST 36
*****
TST36: SCOPE
MOV      #36,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV      #SEOP,NEXT ;POINT TO THE END-OF-PASS HANDLER
TST     MODE        ;IS THIS STAGGERED MODE?
BNE     6$          ;IF NOT, DON'T DO THIS TEST
DCLASM ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
MOV     PAR,R1      ;USE R1 TO BUILD PARAMETERS TO BE LOADED
BIC     #ODDPAR,R1 ;MAKE SURE ODD PARITY ISN'T SET
BIS     #PARITY,R1 ;MAKE SURE PARITY IS TURNED ON
MOV     #1,R2       ;USE R2 AS A LINE POINTER
BIT     R2,LINE     ;IS THIS A VALID LINE?
BEQ     3$          ;IF NOT, SKIP TO THE NEXT LINE
BIT     #BITO,R1    ;IS THIS LINE AN ODD LINE?
BEQ     2$          ;IF IT'S EVEN, USE EVEN PARITY
BIS     #ODDPAR,R1 ;IF IT'S ODD, USE ODD PARITY
MOV     R1,ADZLPR   ;LOAD THE LINE PARAMETER REGISTER
BIC     #ODDPAR,R1 ;SET UP THE NEXT PARITY TO EVEN
INC     R1          ;POINT TO THE NEXT LINE
ASLB   R2          ;MOVE THE BIT POINTER IN R2 TO THE NEXT LINE
BCC     1$          ;IF WE'RE NOT DONE, GO CHECK THE NEXT LINE
CLR     SAVLIN      ;CLEAR THE LINE NUMBER INDICATOR
CLR     R2          ;USE R2 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
CLR     R3          ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
MOV     #40,XMTCNT ;TRANSMIT A BINARY COUNT PATTERN(00-40)
    
```


3836	023114	012700	001422			MOV	#TDO,RO	:POINT TO THE DATA AREA
3837	023120	005020				CLR	(RO)+	:CLEAR A DATA WORD
3838	023122	022700	001462			CMP	#STOP,RO	:FINISHED ?
3839	023126	001374				BNE	-6	:NO
3840	023130	005000				CLR	RO	:CLEAR OFFSET
3841	023132	012777	023410	156736		MOV	#XMTSRV,@DZTIV	:SET UP THE TRANSMITTER INTERRUPT VECTOR
3842	023140	012777	023232	156724		MOV	#PARESE,@DZRIV	:SET UP THE RECEIVER INTERRUPT VECTOR
3843	023146	013777	027060	156720		MOV	DZPRT,@DZ RIS	:SET THE INTERRUPT VECTOR STATUS
3844	023154	013777	027060	156716		MOV	DZPRT,@DZTIS	:SET TRANSMITTER INTERRUPT PRIORITY
3845	023162	052777	040140	156652		BIS	#RIE!TIE!MSENAB,@DZCSR	:ENABLE THE DEVICE
3846	023170	113777	001364	156660		MOVB	LINE,@DZTCR	:ENABLE ALL SELECTED LINES
3847	023176	106427	000000			MTPS	#0	:ALLOW INTERRUPTS
3848	023202	032777	000100	156632	5\$:	BIT	#RIE,@DZCSR	:WHEN RX DONE; RIE WILL =0
3849	023210	001407				BEQ	6\$:BR IF ALL DONE
3850	023212	005237	023404			INC	COUNT0	
3851	023216	102771				BVS	5\$	
3852	023220	105237	023406			INCB	COUNT1	
3853	023224	100366				BPL	5\$	
3854	023226	104011				ERROR	11	:*RX FAILED TO FINISH (INTERRUPT)
3855	023230	104400			6\$:	ADVANCE		:ADVANCE LOOP

```

3856
3857 ;RECEIVER SERVICE ROUTINE(PARITY TEST ONLY)
3858
3859 023232 017704 156610 PARESE: MOV @DZRBUF,R4 ;GET THE CHARACTER
3860 023236 010401 MOV R4,R1 ;COPY THE RECEIVED INFORMATION
3861 023240 000301 SWAB R1 ;GET THE LINE NUMBER IN THE LOWER BYTE
3862 023242 042701 177770 BIC #C<7>,R1 ;ISOLATE THE LINE NUMBER
3863 023246 010137 001372 MOV R1,SAVLIN ;FILL LOC. FOR ERROR PRINTOUT
3864 023252 005704 TST R4 ;WAS DATA VALID?
3865 023254 100401 BMI 10$ ;BRANCH IF YES
3866 023256 104023 ERROR 23 ;ERROR - DATA VALID NOT SET!
3867 023260 006301 10$ ASL R1 ;ALIGN IT ON A WORD BOUNDARY
3868 023262 032704 010000 BIT #PARER,R4 ;PARITY ERROR SHOULD BE SET. IS IT?
3869 023266 001013 BNE 11$ ;IF SO, GO CHECK CHARACTER
3870 023270 013737 002046 001400 MOV DZRBUF,REGIST ;SET UP FOR THE ERROR MESSAGE
3871 023276 010405 MOV R4,R5
3872 023300 042705 000377 BIC #377,R5
3873 023304 156105 001442 BISB TRO(R1),R5 ;GET THE CORRECT CHARACTER
3874 023310 052705 110000 BIS #DVALID!PARER,R5 ;BUILD WHAT WAS EXPECTED
3875 023314 104006 ERROR 6 ;ERROR- DID NOT GET CORRECT INFORMATION
3876 023316 126104 001442 11$ CMPB TRO(R1),R4 ;CHECK THE CHARACTER. IS IT CORRECT?
3877 023322 001407 BEQ 12$ ;IF SO, GO SET UP NEXT CHARACTER
3878 023324 116105 001442 MOVB TRO(R1),R5 ;LOAD THE CHARACTER FOR ERROR REPORTING
3879 023330 042705 177400 BIC #C<377>,R5 ;CLEAR SIGN EXTEND
3880 023334 042704 177400 BIC #C<377>,R4 ;REMOVE THE JUNK FROM R4, THE ACTUAL CHARACTER
3881 023340 104005 ERROR 5 ;DATA ERROR
3882 023342 005261 001442 12$ INC TRO(R1) ;SET UP THE NEXT CHARACTER
3883 023346 005203 INC R3 ;ADD TO THE TOTAL RECEIVED COUNT
3884 023350 005037 023404 CLR COUNT0 ;RESET COUNTERS TO NEXT
3885 023354 005037 023406 CLR COUNT1 ;RECIEVER INTERRUPT
3886 023360 032777 040000 156454 BIT #TIE,@DZCSR ;ARE TRANSMISSIONS DONE?
3887 023366 001005 BNE 13$ ;IF NO, GO RECEIVE SOME MORE
3888 023370 020203 CMP R2,R3 ;ARE ALL CHARACTERS RECEIVED?
3889 023372 001003 BNE 13$ ;IF NO, GO RECEIVE SOME MORE
3890 023374 042777 000100 156440 BIC #RIE,@DZCSR ;DISABLE RECEIVER INTERRUPTS
3891 023402 000002 13$ RTI ;GO BACK TO RECEIVER WAIT LOOP
3892 023404 000000 COUNT0: 0
3893 023406 000000 COUNT1: 0
3894
3895
3896 ;TRANSMITTER INTERRUPT SERVICE
3897 -----
3898
3899 023410 117701 156430 XMTRSrv: MOV @HDZCSR,R1 ;GET THE LINE NUMBER. IS THE TRANSMITTER
3900 023414 100411 BMI 1$ ;REALLY READY? IF SO, GO LOAD THE CHARACTER
3901 023416 013700 001372 MOV SAVLIN,R0 ;ADJUST LOCATION SAVLIN
3902 023422 042701 177770 BIC #C<7>,R1 ;ISOLATE THE LINE NUMBER
3903 023426 010137 001372 MOV R1,SAVLIN ;FOR ERROR PRINTOUT
3904 023432 104003 ERROR 3 ;*TRANSMITTER NOT READY- FALSE INTERRUPT
3905 023434 010037 001372 MOV R0,SAVLIN ;RESET SAVLIN TO PREVIOUS VALUE
3906 023440 042701 177770 1$ BIC #C<7>,R1 ;ISOLATE THE LINE NUMBER
3907 023444 006301 ASL R1 ;MAKE SURE IT REFERENCES A WORD BOUNDARY
3908 023446 116177 001422 156412 MOVB TDO(R1),@DZTDR ;LOAD THE CURRENT CHARACTER FOR THIS LINE
3909 023454 005261 001422 INC TDO(R1) ;SET UP NEXT CHARACTER FOR THIS LINE
3910 023460 005202 INC R2 ;UP THE NUMBER OF TRANSMISSIONS
3911 023462 023761 001376 001422 CMP XMTCNT,TDO(R1) ;HAVE WE DONE ALL PATTERNS ON THIS LINE?

```

```

3912 023470 001015          BNE      4$          ; IF NOT, KEEP ON TRANSMITTING
3913 023472 012700 000001  MOV      #1,R0      ; SET UP A DESELECTION POINTER
3914 023476 006201          ASR      R1          ; GET THE LINE NUMBER AGAIN
3915 023500 005301          2$: DEC     R1          ; REDUCE THE COUNT. WAS THIS THE LINE?
3916 023502 100402          BMI      3$          ; IF SO, GO DISABLE THE ENABLE BIT FOR IT
3917 023504 006300          ASL      R0          ; MOVE THE POINTER TO THE NEXT LINE
3918 023506 000774          BR       2$          ; GO CHECK THE NEXT LINE
3919 023510 140077 156342  3$: BICB   R0, @DZTCR ; DISABLE THE LINE POINTED TO BY R0
3920 023514 001003          BNE      4$          ; IF MORE LINES ARE ACTIVE, GO CONTINUE TRANSMIT
3921 023516 042777 040000 156316 4$: BIC     #TIE, @DZCSR ; IF NOT, DISABLE TRANSMITTER INTERRUPTS
3922 023524 000002          RTI          ; RETURN TO THE TIMING LOOP
3923
3924          ; RELATIVE TIME BUILDING ROUTINE
3925          ; -----
3926
3927 023526 012737 000004 001222 BUILD: MOV     #4, STMP2 ; ROTATE 4 BITS BACK INTO STMP1
3928 023534 006037 001224 1$: ROR     STMP3 ; GET THE BITS FROM STMP3, THE HIGH BYTE
3929 023540 006037 001220          ROR     STMP1 ; OF THE RELATIVE TIME COUNTER. PUT THEM BACK
3930 023544 005337 001222          DEC     STMP2 ; INTO STMP1 USING THE CARRY BIT WITH
3931          ; ROTATE INSTRUCTIONS
3932 023550 001371          BNE      1$          ; REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
3933 023552 000207          RTS      PC         ; RETURN TO CALLING TEST
3934

```

```

;RECEIVER SERVICE ROUTINE
3935
3936
3937 023554 105777 156262 RXISR1: TSTB @DZCSR ; IS THE RECEIVER REALLY READY?
3938 023560 100401 BMI 1$ ; IF SO, GO SERVICE IT
3939 023562 104004 ERROR 4 ; *ERROR- RECEIVER DONE FLAG ISN'T SET
3940 023564 017704 156256 1$: MOV @DZRBUF,R4 ; SAVE THE RECEIVER INFORMATION
3941 023570 100401 BMI 2$ ; IF IT WAS VALID, GO PROCESS IT
3942 023572 104023 ERROR 23 ; ERROR- DATA VALID WASN'T SET
3943 023574 032704 070000 2$: BIT #OVRUN!FRMERR!PARER,R4 ; ARE ANY ERROR FLAGS SET?
3944 023600 001403 BEQ 3$ ; IF NOT, GO CONTINUE PROCESSING
3945 023602 013700 002046 MOV DZRBUF,R0 ; SET UP FOR ERROR REPORTING
3946 023606 104002 ERROR 2 ; ERROR- RECEIVER ERROR FLAG SET
3947 023610 010401 3$: MOV R4,R1 ; COPY THE RECEIVER INFORMATION
3948 023612 000301 SWAB R1 ; GET THE LINE NUMBER IN THE LOWER BYTE
3949 023614 042701 177770 BIC #1C<7>,R1 ; ISOLATE THE LINE NUMBER
3950 023620 006301 ASL R1 ; ALIGN IT ON A WORD BOUNDARY
3951 023622 120461 001442 CMPB R4,TR0(R1) ; IS THE CHARACTER WHAT IT SHOULD BE?
3952 023626 001413 BEQ 4$ ; IF SO, GO CONTINUE PROCESSING
3953 023630 116105 001442 MOVB TR0(R1),R5 ; GET WHAT WAS EXPECTED FOR ERROR REPORTING
3954 023634 042705 177400 BIC #1C<377>,R5 ; ELIMINATE PROPAGATED SIGN
3955 023640 042704 177400 BIC #1C<377>,R4 ; ISOLATE THE ACTUAL CHARACTER
3956 023644 010137 001372 MOV R1,SAVLIN ; GET THE LINE NUMBER OF THE RECEIVER ERROR
3957 023650 006237 001372 ASR SAVLIN ; ALIGN IT CORRECTLY FOR REPORTING
3958 023654 104005 ERROR 5 ; *DATA ERROR
3959 023656 005261 001442 4$: INC TR0(R1) ; SET UP THE NEXT EXPECTED CHARACTER
3960 023662 005203 INC R3 ; INCREMENT THE COUNT OF RECEIVED CHARACTERS
3961 023664 032761 000020 001442 BIT #20,TR0(R1) ; HAVE ALL CHARACTERS BEEN RECEIVED?
3962 023672 001402 BEQ 5$ ; IF NOT, GO RECEIVE SOME MORE
3963 023674 020203 CMP R2,R3 ; HAVE WE RECEIVED ALL CHARACTERS?
3964 023676 001401 BEQ 6$ ; IF SO, GO DETERMINE THE TIMING
3965 023700 000002 5$: RTI ; GO CONTINUE TIMING AND ALLOW INTERRUPTS
3966 023702 004737 023526 6$: JSR PC,BUILD ; GET THE RELATIVE TIME (SIGNIFICANT BITS)
3967
3968 023706 013700 023766 MOV OFFSET,R0 ; GET POINTER
3969 023712 013760 001220 002102 MOV $TMP1,TMTBL(R0) ; SAVE THIS TEST'S TIME
3970 023720 005737 023766 TST OFFSET ; FIRST TEST?
3971 023724 001414 BEQ 7$ ; IF NOT, GO CHECK THE TIME
3972 023726 005740 TST -(R0) ; POINT TO THE PREVIOUS TIME TAKEN
3973 023730 026037 002102 001220 CMP TMTBL(R0),$TMP1 ; IS THIS TIME WHAT IT SHOULD BE?
3974 023736 101007 BHI 7$ ; IF SO, GO TO THE NEXT TEST
3975 023740 016005 002102 MOV TMTBL(R0),R5 ; PLACE WHAT WAS EXPECTED IN R5
3976 023744 010137 001372 MOV R1,SAVLIN ; GET THE LINE NUMBER OF THE RECEIVER
3977 023750 006237 001372 ASR SAVLIN ; MAKE SURE IT'S THE LINE NUMBER
3978 023754 104021 ERROR 21 ; TIMING ERROR
3979 023756 042777 000140 156056 7$: BIC #RIE!MSENAB,@DZCSR ; DISABLE THE DEVICE
3980 023764 000002 RTI ; RETURN TO THE PROGRAM
3981 023766 000000 OFFSET: 0

```

;DZ11 ECHO/CABLE TEST
;COPYRIGHT 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037

; *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 STARTE

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

023770	012706	001120		XSTART: MOV	#STACK, SP	; SET UP PROCESSOR STACK
023774	106427	000340		MTPS	#PR7	; LOCK OUT INTERRUPTS
024000	012737	023770	001126	MOV	#XSTART, \$LPADR	; SET UP IN CASE OF POWER FAIL
024006	005037	026164		CLR	STFLG	; CLEAR TEST START FLAG
024012	005037	001242		CLR	\$PASS	; CLEAR PASS COUNT
024016	005037	001132		CLR	\$ERTTL	; CLEAR ERROR COUNT
024022	105037	001123		CLRB	\$ERFLG	; CLEAR ERROR FLAG

4038	024026	005037	026170			CLR	LAST		; CLEAR LAST ERROR PC
4039	024032	032777	000001	155120	VEC1:	BIT	#SW00, @SWR		; IF SW00=1, GET NEW VECTOR
4040	024040	001465				BEQ	OTHER		; AND CSR
4041	024042	012701	000300		VEC2:	MOV	#300, R1		
4042	024046	012702	000302			MOV	#302, R2		
4043	024052	010221			IS:	MOV	R2, (R1)+		; RESTORE TRAPCATCHER
4044	024054	005022				CLR	(R2)+		; IN FLOATING VECTOR AREA
4045	024056	022122				CMP	(R1)+, (R2)+		; UPDATE THE POINTERS
4046	024060	020127	001000			CMP	R1, #1000		
4047	024064	001372				BNE	IS		
4048	024066	104403				INSTR			; INPUT ADDRESS OF DEVICE VECTOR
4049	024070	026216				MVECTOR			; MESSAGE "VECTOR ADDRESS--"
4050	024072	104405				PARAM			; CONVERT STRING TO OCTAL
4051	024074	000300				300			; LOW LIMIT
4052	024076	000770				770			; HIGH LIMIT
4053	024100	002072				DZRIV			; LOCATIONS TO BE FILLED
4054	024102	003			.BYTE	3			; LSB MASK
4055	024103	004			.BYTE	4			; NUMBER OF LOCATIONS
4056	024104	104403				INSTR			; INPUT ADDRESS OF DEVICE CSR
4057	024106	026240				MREGAD			; MESSAGE "CONTROL REGISTER ADDRESS--"
4058	024110	104405				PARAM			; CONVERT STRING TO OCTAL
4059	024112	160000				160000			; LOW LIMIT
4060	024114	163700				163700			; HIGH LIMIT
4061	024116	002042				DZCSR			; LOCATIONS TO BE FILLED
4062	024120	007			.BYTE	7			; LSB MASK
4063	024121	001			.BYTE	1			; NUMBER OF LOCATIONS
4064	024122	013737	002042	002046		MOV	DZCSR, DZRBUF		; BEGIN BUILDING DEVICE ADDRESSES
4065	024130	062737	000002	002046		ADD	#2, DZRBUF		; FORM THE READ BUFFER ADDRESS
4066	024136	013737	002046	002052		MOV	DZRBUF, DZLPR		; REMEMBER THAT THIS IS ALSO LINE PARAMETER REG.
4067	024144	013737	002046	002056		MOV	DZRBUF, DZTCR		; BEGIN BUILDING TRANSMITTER CONTROL REGISTER
4068	024152	062737	000002	002056		ADD	#2, DZTCR		; FORM THE TRANSMITTER CONTROL REGISTER POINTER
4069	024160	013737	002056	002060		MOV	DZTCR, HDZTCR		
4070	024166	005237	002060			INC	HDZTCR		
4071	024172	013737	002056	002066		MOV	DZTCR, DZTDR		; BEGIN FORMING TRANSMITTER DATA REGISTER
4072	024200	062737	000002	002066		ADD	#2, DZTDR		; FORM THE TRANSMITTER DATA REGISTER
4073	024206	013737	002066	002062		MOV	DZTDR, DZMSR		
4074	024214	032777	000002	154736	OTHER:	BIT	#SW01, @SWR		; RESELECT OF TEST?
4075	024222	001427				BEQ	XBEGIN		; IF NOT, SKIP ASKING WHICH ONE
4076	024224	104403				INSTR			; INPUT WHICH TEST YOU ARE RUNNING
4077	024226	026424				MWHICH			; ECHO OR CABLE
4078	024230	104416				PAWCH			; SET FLAG
4079	024232	026162				WCHFLG			; THIS FLAG
4080	024234	104403			BAUD:	INSTR			; INPUT BAUD RATE
4081	024236	026346				MSPEED			; MESSAGE "BAUD RATE--"
4082	024240	104415				PARMD			; CONVERT DECIMAL STRING TO OCTAL
4083	024242	000062				50.			; LOW LIMIT
4084	024244	022600				9600.			; HIGH LIMIT
4085	024246	026200				LINESP			; LOCATION TO BE FILLED
4086	024250	000			.BYTE	0			; LSB MASK
4087	024251	001			.BYTE	1			; NUMBER OF LOCATIONS
4088	024252	104413			LINEX:	DEVICE.CLR			; CLEAR DEVICE
4089	024254	005037	026164			CLR	STFLG		; CLEAR PROGRAM START FLAG
4090	024260	104403				INSTR			; INPUT LINE NUMBER
4091	024262	026336				MLINE			; MESSAGE "LINE NUMBER--"
4092	024264	104405				PARAM			; CONVERT STRING TO OCTAL
4093	024266	000000				0			; LOW LIMIT

```

4094 024270 000007          7          ;HIGH LIMIT
4095 024272 001372          SAVLIN      ;LOCATION TO BE FILLED
4096 024274          000          .BYTE      ;LSB MASK
4097 024275          001          .BYTE      ;NUMBER OF LOCATIONS
4098 024276 004537 025766    JSR        R5,SET
4099
4100 024302 106427 000340    XBEGIN: MTPS  #PR7          ;LOCK OUT INTERRUPTS
4101 024306 012706 001120    MOV        #STACK,SP      ;SET UP PROCESSOR STACK
4102 024312 005037 026166    CLR        LOCKUP         ;CLEAR TIMEOUT
4103 024316 005737 026162    TST        WCHFLG         ;ECHO OR CABLE TEST ?
4104 024322 001413          BEQ        2$              ;ECHO
4105 024324 012737 025040 001126  MOV        #TEST2,$LPADR   ;CABLE TEST
4106 024332 005737 026164    TST        STFLG          ;ARE YOU LOOPING ?
4107 024336 001017          BNE        1$              ;YES
4108 024340 005137 026164    COM        STFLG          ;NO
4109 024344 104402 026517    TYPE      MCABLE         ;TYPE CABLE TEST
4110 024350 000412          BR         1$
4111 024352 012737 024402 001126 2$: MOV        #TEST1,$LPADR   ;SET UP ECHO TEST
4112 024360 005737 026164    TST        STFLG          ;ARE YOU LOOPING ?
4113 024364 001004          BNE        1$              ;YES
4114 024366 005137 026164    COM        STFLG          ;NO
4115 024372 104402 026472    TYPE      MTERM          ;TYPE ECHO TEST
4116 024376 000177 154524    1$: JMP        $SLPADR      ;START TESTING
4117          ;THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
4118          ;:(IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
4119          ;ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
4120
4121 024402 104413          TEST1: DEVICE.CLR        ;CLEAR DZ11
4122 024404 012737 000001 001122  MOV        #1,$STSTNM
4123 024412 013777 026206 155436  MOV        NUMTCR,$DZTCR  ;SET TCR BIT
4124 024420 013737 026204 001366  MOV        NUMLIN,PAR     ;SET PARAMETERS
4125 024426 053737 026202 001366  BIS        SPEED,PAR     ;SET BAUD RATE
4126 024434 013777 001366 155410  MOV        PAR,$DZLPR    ;LOAD PARAM.
4127 024442 012777 000040 155372  MOV        #MSENAB,$DZCSR ;SET SCANN ENABLE
4128 024450 005004          CLR        R4
4129 024452 012705 026534          MOV        #MQUICK,R5    ;SET MESSAGE BUFFER
4130 024456 005777 155360 3$: TST        $DZCSR      ;TRDY?
4131 024462 100404          BMI        2$            ;BR IF YES
4132 024464 104414          DELAY
4133 024466 005304          DEC        R4
4134 024470 001372          BNE        3$
4135 024472 104003          ERROR     3              ;NO TRDY SET! WHY?
4136 024474 005004          CLR        R4            ;RESET COUNTER TO 0
4137 024476 112577 155364 2$: MOVB      (R5)+,$DZTDR  ;LOAD CHAR
4138 024502 001365          BNE        3$
4139 024504 004737 007360          JSR        PC,SERV.G     ;<↑G>?
4140 024510 122777 000377 154442  CMPB      #377,$SWR     ;RE-DO QUICK BROWN?
4141 024516 001731          BEQ        TEST1        ;BR IF REPEAT PATTERN
4142 024520 104413          DEVICE.CLR
4143 024522 106427 000340          MTPS  #PR7          ;LOCK OUT INTERRUPTS
4144 024526 012737 025476 001360  MOV        #XEOP,NEXT
4145 024534 104413          DEVICE.CLR
4146 024536 013737 026204 001366  MOV        NUMLIN,PAR     ;SELECT LINE # & SET INTERRUPT ENABLE
4147 024544 053737 026202 001366  BIS        SPEED,PAR     ;SET LINE SPEED AND
4148          ;CHARACTER LENGTH (TRANS. & REC.)
4149 024552 052737 010000 001366  BIS        #RCVON,PAR    ;MAKE SURE RECEIVER IS TURNED ON

```

```

4150 024560 013777 001366 155264      MOV     PAR,ADZLPR      ;LOAD THE LINE PARAMETER REGISTER
4151 024566 012777 024642 155276      MOV     #INTSVC,ADZRIV ;SET UP INTERRUPT SERVICE
4152 024574 013777 026210 155272      MOV     PRIO,ADZRRIS  ;AND LEVEL
4153 024602 106437 027062      MTPS   @#LESS1       ;ALLOW INTERRUPTS
4154 024606 012777 000140 155226      MOV     #RIE!MSENAB,ADZCSR ;SET RECEIVER INTERRUPT ENABLE
4155 024614 104402 026364      TYPE   MCHAR         ;TYPE "ANY CHARACTER"
4156 024620 105777 154340      1$:    TSTB   @STKS     ;IF SOMEBODY HITS A KEY- GET NEW LINE #
4157 024624 100375      BPL    1$            ;LOOP HERE
4158 024626 005777 154334      TST    @STKB        ;CLEAR CHAR
4159 024632 004737 007360      JSR    PC,SERV.G    ;MAKE SURE IT WASN'T <↑G>
4160 024636 000137 024252      JMP    LINEX        ;
    
```

```

4161
4162
4163
4164 024642 105777 155174      INTSVC: TSTB   @DZCSR     ;THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
4165 024646 100401      BMI    .+4          ;TEST REC. FLAG
4166 024650 104004      ERROR  4           ;ERROR - INTERRUPT NOT CAUSED BY FLAG
4167 024652 017737 155170 026212      MOV     @DZRBUF,RECDAT
4168 024660 100401      BMI    .+4
4169 024662 104023      ERROR  23          ;NON- VALID CHARACTER
4170 024664 032737 020000 026212      BIT     #BIT13,RECDAT ;CHECK FOR FRAMING ERROR
4171 024672 001401      BEQ    .+4          ;BR IF NO ERROR
4172 024674 104025      ERROR  25          ;EITHER SOMEBODY HIT THE
4173                                     ;"BREAK KEY" OR YOU HAVE AN ERROR!
4174 024676 113737 026212 026214      MOVSB  RECDAT,TBUF   ;MOVE CHARACTER TO OUTPUT AREA
4175 024704 113737 026212 010620      MOVSB  RECDAT,INBUF  ;MOVE CHARACTER TO CHECK FOR ↑C
4176 024712 042737 177600 010620      BIC    #↑C<177>,INBUF ;STRIP JUNK PLUS PARITY
4177 024720 042737 174377 026212      BIC    #174377,RECDAT ;SAVE ONLY LINE NUMBER
4178 024726 000337 026212      SWAB   RECDAT
4179 024732 023737 001372 026212      CMP    SAVLIN,RECDAT ;DOES THE LINE # COMPARE?
4180 024740 001401      BEQ    .+4
4181 024742 104015      ERROR  15          ;*WRONG LINE NUMBER
4182 024744 012777 000040 155070      MOV     #MSENAB,ADZCSR ;START THE TRANSMITTERS SCANNER
4183 024752 123727 010620 000003      CMPB   INBUF,#3     ;IS IT A ↑C ?
4184 024760 001004      BNE    1$          ;NO
4185 024762 104413      DEVICE.CLR
4186 024764 012716 025476      MOV     #XEOP,(SP)  ;CRUNCH STACK
4187 024770 000002      RTI
4188 024772 005003      1$:    CLR     R3        ;INITIALIZE DELAY
4189 024774 013777 026206 155054      MOV     NUMTCR,@DZTCR ;ENABLE THE LINE
4190 025002 005777 155034      10$:   TST    @DZCSR     ;TRANSMITTER READY?
4191 025006 100403      BMI    2$          ;IF YES BRANCH
4192 025010 005203      INC    R3          ;INCREMENT DELAY
4193 025012 001373      BNE    10$        ;DELAY DONE?
4194 025014 104003      ERROR  3           ;TRANSMIT READY NOT SET!
4195 025016 113777 026214 155042      MOVSB  TBUF,@DZTCR  ;TRANSMIT THE CHARACTER
4196 025024 012777 000140 155010      MOV     #RIE!MSENAB,ADZCSR ;RESTART THE RECEIVER
4197 025032 005077 155020      CLR    @DZTCR     ;CLEAR TCR BIT
4198 025036 000002      RTI
4199
4200
4201
4202
4203
4204 025040 106427 000340      ;THIS TEST TRANSMITS A BINARY COUNT PATTERN
4205 025044 012737 000002 001122      ;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
    
```



```

4206 025052 012737 025476 001360 MOV #XEOP,NEXT
4207 025060 104413 DEVICE.CLR
4208 ;*TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
4209 ;*WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
4210 ;*THE DIST PNL MUST HAVE JUMPER FROM DTR TO RQST TO SEND
4211 ;*IN ORDER FOR THIS TEST TO WORK!
4212 025062 012737 025070 001362 MOV #15,LOCK ;LOOP
4213 025070 113777 026206 154762 1$: MOVB NUMTCR,ADZTCR ;SET DTR
4214 025076 005005 CLR R5
4215 025100 153705 026206 BISB NUMTCR,R5 ;BUILD EXPECTED
4216 025104 000305 SWAB R5 ;PUT IN HIGH BYTE
4217 025106 153705 026206 BISB NUMTCR,R5
4218 025112 104414 DELAY ;WAIT FOR CABLE DELAY
4219 025114 017704 154742 MOV ADZMSR,R4 ;READY MODEM BITS
4220 025120 020504 CMP R5,R4 ;ARE THEY OK?
4221 025122 001401 BEQ 25 ;BR IF YES
4222 025124 104022 ERROR 22 ;IS THE TEST CONNECTOR ON?
4223 ;HAS RIGHT LINE BEEN SELECTED?
4224 ;IF SO- YOU HAVE A PROBLEM!
4225 ;MODEM BITS NOT RIGHT
4226 025126 104401 2$: SCOP1 ;LOOP
4227 025130 104413 3$: DEVICE.CLR ;INIT DZ11
4228 025132 013737 026202 001366 MOV SPEED,PAR ;SET LINE SPEED
4229 025140 053737 026204 001366 BIS NUMLIN,PAR ;SELECT LINE # & REC. INTERRUPT ENABLE
4230 025146 052737 010000 001366 BIS #RCVON,PAR ;ENABLE THE RECEIVER FOR THIS LINE
4231 025154 052777 040140 154660 BIS #TIE!RIE!MSENAB,ADZCSR ;SET TRANSMITTER INTERRUPT ENABLE
4232 025162 012777 025276 154702 MOV #INTREC,ADZRIV ;SET UP INTR SERVICE
4233 025170 013777 026210 154676 MOV #PRIO,ADZRIS ;SET UP LEVEL
4234 025176 012777 025456 154672 MOV #INTRAN,ADZTIV ;SET UP INTR SERVICE
4235 025204 013777 026210 154666 MOV #PRIO,ADZTIS ;SET UP LEVEL
4236 025212 005001 CLR R1 ;RX DATA POINTER- SET TO 0
4237 025214 005002 CLR R2 ;TX DATA POINTER- SET TO 0
4238 025216 013777 001366 154626 MOV PAR,ADZLPR ;SET THE PARAMETERS AND TURN ON RECEIVER
4239 025224 106437 027062 MTPS #LESS1 ;ALLOW INTERRUPTS
4240 025230 013777 026206 154620 MOV NUMTCR,ADZTCR ;SET UP TCR BIT
4241
4242 ;YOU RETURN HERE AFTER EVERY RECEIVER INTERRUPT
4243 025236 105777 153722 SPIN: TSTB #STKS ;IF SOMEBODY HITS A KEY- GET A NEW LINE #
4244 025242 100006 BPL 15 ;BR IF NO KEY HIT
4245 025244 005777 153716 TST #STKB ;CLEAR CHAR
4246 025250 004737 007360 JSR PC,SERV.G ;MAKE SURE IT WASN'T <↑G>
4247 025254 000137 024252 JMP LINEX ;SW02=1
4248 025260 005237 026166 1$: INC LOCKUP ;INC TIMEOUT FLAG
4249 025264 001364 BNE SPIN ;IF NOT 0 RETURN SPINNING
4250 025266 104011 ERROR 11 ;*RECEIVER FAILED TO INTERRUPT CHECK CABLE/TERMINATOR
4251 025270 104413 QUIT: DEVICE.CLR
4252 025272 000137 025476 JMP XEOP ;CALL FOR END OF PASS
4253 025276 005037 026166 INTREC: CLR LOCKUP ;CLEAR TIMEOUT FLAG
4254 025302 105777 154534 TSTB ADZCSR ;TEST REC DONE
4255 025306 100401 BMI .+4 ;YES
4256 025310 104004 ERROR 4 ;*FALSE INTERRUPT
4257 025312 017737 154530 026212 MOV ADZRBUF,RECDAT ;SAVE WORD
4258 025320 100401 BMI .+4
4259 025322 104023 ERROR 23 ;*NON VALID CHARACTER
4260 025324 032737 040000 026212 BIT #BIT14,RECDAT ;DATA OVERRUN ?
4261 025332 001401 BEQ .+4 ;NO
    
```

4262	025334	104024			ERROR	24		; *YES
4263	025336	032737	020000	026212	BIT	#BIT13, RECDAT		; FRAMING ERROR ?
4264	025344	001401			BEQ	.+4		; NO
4265	025346	104025			ERROR	25		; *YES
4266	025350	032737	010000	026212	BIT	#BIT12, RECDAT		; PARITY ERROR ?
4267	025356	001401			BEQ	.+4		; NO
4268	025360	104026			ERROR	26		; *YES
4269	025362	110105			MOVB	R1, R5		; SET EXPECTED
4270	025364	042705	177400		BIC	#1C<377>, R5		; CLEAR HIGH BYTE
4271	025370	113704	026212		MOVB	RECDAT, R4		; GET FOUND
4272	025374	042704	177400		BIC	#1C<377>, R4		; CLEAR HIGH BYTE
4273	025400	020504			CMP	R5, R4 ; OK?		
4274	025402	001401			BEQ	.+4		
4275	025404	104005			ERROR	5		; DATA ERROR
4276	025406	042737	174377	026212	BIC	#174377, RECDAT		; SAVE ONLY LINE NUMBER
4277	025414	000337	026212		SWAB	RECDAT		
4278	025420	023737	001372	026212	CMP	SAVLIN, RECDAT		; DOES THE LINE # COMPARE ?
4279	025426	001401			BEQ	.+4		; YES
4280	025430	104015			ERROR	15		; *WRONG LINE #
4281	025432	120127	000377		CMPB	R1, #377		; LAST CHARACTER ?
4282	025436	001003			BNE	1\$; NO
4283	025440	012716	025270		MOV	#QUITS, (SP)		; CRUNCH STACK
4284	025444	000403			BR	2\$		
4285	025446	105201			1\$: INCB	R1		; UPDATE EXPECTED DATA
4286	025450	012716	025236		MOV	#SPIN, (SP)		; CRUNCH STACK
4287	025454	000002			2\$: RTI			
4288								
4289	025456	005777	154360		INTRAN: TST	@DZCSR ; TEST TRANSMIT FLAG		
4290	025462	100401			BMI	.+4		
4291	025464	104003			ERROR	3		; *FALSE INTERRUPT
4292	025466	110277	154374		MOVB	R2, @DZTDR		; TRANSMIT A CHARACTER
4293	025472	105202			INCB	R2		; UPDATE TX DATA
4294	025474	000002			RTI	; RETURN		

```

4295
4296
4297
4298
4299 025476 104402
4300 025500 026274
4301 025502 005037 026170
4302 025506 105037 001123
4303 025512 000137 024302
4304
4305
4306 025516 011605
4307 025520 012537 025702
4308 025524 012537 025704
4309 025530 012537 025706
4310 025534 112537 025710
4311 025540 112537 025711
4312 025544 010516
4313 025546 005005
4314 025550 012704 010620
4315 025554 122714 000015
4316 025560 001424
4317 025562 121427 000060
4318 025564 002421
4319 025570 121427 000071
4320 025574 003016
4321 025576 142714 000060
4322 025602 005002
4323 025604 152402
4324 025606 060205
4325 025610 122714 000015
4326 025614 001410
4327 025616 006305
4328 025620 010502
4329 025622 006305
4330 025624 006305
4331 025626 060205
4332 025630 000754
4333 025632 104404
4334 025634 000744
4335
4336
4337
4338 025636 020537 025704
4339 025642 101373
4340 025644 020537 025702
4341 025650 103770
4342 025652 133705 025710
4343 025656 001365
4344
4345
4346
4347 025660 013704 025706
4348 025664 010524
4349 025666 062705 000002
4350 025672 105337 025711

```

```

;END OF PASS
;RESTART TEST
XEOP: TYPE ;TYPE NAME OF TEST
MPASS
CLR LAST ;CLEAR LAST ERROR PC
CLRB SERFLG ;CLEAR ERROR FLAG
RSTRT: JMP XBEGIN

.PARMD: ;CONVERT DECIMAL ASCII STRING TO OCTAL
MOV (SP),R5
MOV (R5)+,6$
MOV (R5)+,7$
MOV (R5)+,8$
MOVB (R5)+,9$
MOVB (R5)+,10$
MOV R5,(SP)
2$: CLR R5
MOV #INBUF,R4
CMPB #15,(R4)
BEQ 3$
1$: CMPB (R4),#'0
BLT 3$
CMPB (R4),#'9
BGT 3$
BICB #'0,(R4)
CLR R2
BISB (R4)+,R2
ADD R2,R5
CMPB #15,(R4)
BEQ 4$
ASL R5 ;X2
MOV R5,R2 ;SAVE X2
ASL R5 ;X4
ASL R5 ;X8
ADD R2,R5 ;TIMES 10
BR 1$
3$: INSTER
BR 2$

;TEST TO SEE IF NUMBER IS WITHIN LIMITS
4$: CMP R5,7$
BHI 3$
CMP R5,6$
BLO 3$
BITB 9$,R5
BNE 3$

;STORE NUMBER AT SPECIFIED ADDRESS
5$: MOV 8$,R4
MOV R5,(R4)+
ADD #2,R5
DECB 10$

```

```

4351 025676 001372          BNE      5$
4352 025700 000002          RTI
4353 025702 000000          6$:      0
4354 025704 000000          7$:      0
4355 025706 000000          8$:      0
4356 025710          000          9$:      .BYTE 0
4357 025711          000          10$:     .BYTE 0
4358
4359
4360
4361          ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
4362          ;BUFFER TO THE CHARACTERS "E" AND "C"
4363          ;IF THE CHARACTER IS "E" CLEAR THE FLAG
4364          ;IF THE CHARACTER IS "C" SET THE FLAG
4365 025712 017605 000000          .PAWCH:MOV    2(SP),R5
4366 025716 142737 000040 010620          BICB      #40,INBUF      ;SET FOR LOWER CASE INPUT
4367 025724 122737 000105 010620          CMPB      #'E,INBUF      ;IS IT "E" ?
4368 025732 001002          BNE      1$
4369 025734 105015          CLR      (R5)           ;000
4370 025736 000406          BR       2$
4371 025740 122737 000103 010620          1$:      CMPB      #'C,INBUF      ;IS IT "C" ?
4372 025746 001005          BNE      3$
4373 025750 112715 177777          MOV      #-1,(R5)       ;3177
4374 025754 062716 000002          2$:      ADD      #2,(SP)
4375 025760 000002          RTI
4376 025762 104404          3$:      INSTER          ;RETRY
4377 025764 000752          BR       .PAWCH
4378
4379
4380
4381          ;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
4382          ;LINE NUMBER (SAVLIN) FOR DZLPR, DZTCR AND DZCSR
4383          ;REGISTER USAGE.
4384
4385 025766 013737 001372 026204          SET:      MOV      SAVLIN,NUMLIN      ;SAVE SAVLIN
4386 025774 013700 001372          XTCRO:    MOV      SAVLIN,R0          ;COPY THE LINE NUMBER FOR LOOP CONTROL
4387 026000 005037 026206          CLR      NUMTCR          ;SET A DEFAULT OF LINE 0 OR NO LINES
4388 026004 012702 000001          MOV      #1,R2          ;SET A BIT POINTER TO THE FIRST LINE
4389 026010 005300          XTCR1:    DEC      R0          ;REDUCE THE INDICATOR, IS IT MINUS YET?
4390 026012 100402          BMI      SET1          ;IF SO, R2 POINTS TO THE RIGHT LINE
4391 026014 006302          ASL      R2          ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
4392 026016 000774          BR       XTCR1          ;GO SEE IF THIS LINE IS THE ONE
4393 026020 012701 026062          SET1:    MOV      #TABLE2,R1
4394 026024 010237 026206          MOV      R2,NUMTCR      ;COPY THE CORRECT BIT POINTER
4395 026030 022137 026200          1$:      CMP      (R1)+,LINESP
4396 026034 001407          BEQ      2$
4397 026036 005721          TST      (R1)+          ;IS IT THE END OF TABLE?
4398 026040 001373          BNE      1$            ;NO
4399 026042 104402 026310          TYPE     #MINVAL          ;INVALID BAUD RATE,BEGIN AGAIN
4400 026046 012705 024234          MOV      #BAUD,R5       ;JUMP TO BAUD THRU R5
4401 026052 000402          BR       3$
4402 026054 011137 026202          2$:      MOV      (R1),SPEED      ;SET UP BAUD RATE
4403 026060 000205          3$:      RTS      R5
4404
4405
4406
    
```

4407
4408 026062 000062
4409 026064 010070
4410 026066 000113
4411 026070 010470
4412 026072 000156
4413 026074 011070
4414 026076 000207
4415 026100 011470
4416 026102 000226
4417 026104 012070
4418 026106 000454
4419 026110 012430
4420 026112 001130
4421 026114 013030
4422 026116 002260
4423 026120 013430
4424 026122 003410
4425 026124 014030
4426 026126 003720
4427 026130 014430
4428 026132 004540
4429 026134 015030
4430 026136 007020
4431 026140 015430
4432 026142 011300
4433 026144 016030
4434 026146 016040
4435 026150 016430
4436 026152 022600
4437 026154 017070
4438 026156 177777 000000
4439
4440
4441 026162 000000
4442 026164 000000
4443 026166 000000
4444 026170 000000
4445 026172 000000
4446 026174 000000
4447 026176 000000
4448 026200 000156
4449 026202 006307
4450
4451 026204 000100
4452
4453 026206 000001
4454 026210 000240
4455 026212 000000
4456 026214 000000
4457 026216 053200 041505 047524
026240 041600 047117 051124
026274 050200 051501 020123
026310 044600 053116 046101
026336 046200 047111 035105
026346 041200 052501 020104

TABLE2: ;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR)
:WORD 50. :50 BAUD
:WORD 10070 :75 BAUD
:WORD 75. :110 BAUD
:WORD 10470 :TWO STOP BITS
:WORD 110. :134.5 BAUD
:WORD 11070 :TWO STOP BITS
:WORD 135. :150 BAUD
:WORD 11470 :TWO STOP BITS
:WORD 150. :300 BAUD
:WORD 12070 :ONE STOP BIT
:WORD 300. :600 BAUD
:WORD 12430 :ONE STOP BIT
:WORD 600. :1200 BAUD
:WORD 13030 :ONE STOP BIT
:WORD 1200. :1800 BAUD
:WORD 13430 :ONE STOP BIT
:WORD 1800. :2000 BAUD
:WORD 14030 :ONE STOP BIT
:WORD 2000. :2400 BAUD
:WORD 14430 :ONE STOP BIT
:WORD 2400. :3600 BAUD
:WORD 15030 :ONE STOP BIT
:WORD 3600. :4800 BAUD
:WORD 15430 :ONE STOP BIT
:WORD 4800. :7200 BAUD
:WORD 16030 :ONE STOP BIT
:WORD 7200. :9600 BAUD
:WORD 16430 :ONE STOP BIT
:WORD 9600. :9600 BAUD
:WORD 17070 :TABLE TERMINATOR
:WORD -1,0

WCHFLG: 0 ;ECHO OR CABLE FLAG
STFLG: 0 ;PROGRAM START FLAG
LOCKUP: 0 ;TIMEOUT FLAG
LAST: 0 ;LAST ERROR PC
TDATA: 0
RDATA: 0
BYTCNT: 0
LINESP: 110 ;DEFAULT BAUD RATE
SPEED: 6307 ;DEFAULT 110 BAUD, 8 BITS/CHAR,
;FDX 2 STOP BITS
;DEFAULT VALUE, REC. INTERRUPT ENABLED
NUMLIN: 100
NUMTCR: 1 ;DEFAULT VALUE, TCR BIT 0
PRIO: 240 ;DEFAULT DEVICE PRIORITY 5
RECDAT: 0
TBUF: 0
MVECTO: .ASCIZ <200>/VECTOR ADDRESS- /
MREGAD: .ASCIZ <200>/CONTROL REGISTER ADDRESS- /
MPASS: .ASCIZ <200>/PASS DONE. /
MINVAL: .ASCIZ <200>/INVALID BAUD RATE - /
MLINE: .ASCIZ <200>/LINE: /
MSPEED: .ASCIZ <200>/BAUD RATE - /

026364	052200	050131	020105
026424	053600	044510	044103
026472	052200	051105	044515
026517	200	040503	046102
026534	006777	177777	177412
026543	124	042510	050440
026640	006777	177777	177412
	026652		

```

MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZ11 TERMINAL /
MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /
MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /
MCABLE: .ASCIZ <200>/CABLE TEST /
MQUICK: .ASCII <377><15><377><377><12><377><377>
        .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/
        .ASCII <377><15><377><377><12><377><377><377><0>

```

```

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

```

4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503

026652	006337	027060	
026656	006337	027060	
026662	006337	027060	
026666	006337	027060	
026672	006337	027060	
026676	013737	027060	027062
026704	162737	000001	027062
026712	042737	000037	027062
026720	013700	002072	
026724	062700	000002	
026730	010037	002074	
026734	062700	000002	
026740	010037	002076	
026744	062700	000002	
026750	010037	002100	
026754	013700	001310	
026760	010037	002042	
026764	005200		
026766	010037	002044	
026772	005200		
026774	010037	002046	
027000	010037	002052	
027004	005200		
027006	010037	002050	
027012	010037	002054	
027016	005200		
027020	010037	002056	
027024	005200		
027026	010037	002060	
027032	005200		
027034	010037	002062	
027040	010037	002066	
027044	005200		
027046	010037	002064	
027052	010037	002070	
027056	000207		
027060	000240		
027062	000200		

```

;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, DZ RIS ;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, HDZTCR ;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
LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS

```

			;ERROR ERROR TABLE	
	.ERRTAB:			
4504				
4505	027064	000000	0	;ERROR 0
4506	027066	000000	0	
4507	027070	000000	0	
4508				
4509	027072	027304	EM1	;ERROR
4510	027074	030554	DH1	
4511	027076	030752	DT1	
4512				
4513	027100	027357	EM2	;ERROR 2
4514	027102	030577	DH2	
4515	027104	030764	DT2	
4516				
4517	027106	027405	EM3	;ERROR 3
4518	027110	030632	DH3	
4519	027112	031002	DT3	
4520				
4521	027114	027444	EM4	;ERROR 4
4522	027116	030632	DH3	
4523	027120	031002	DT3	
4524				
4525	027122	027473	EM5	;ERROR 5
4526	027124	030644	DH4	
4527	027126	031010	DT4	
4528				
4529	027130	027522	EM6	;ERROR 6
4530	027132	030644	DH4	
4531	027134	031010	DT4	
4532				
4533	027136	027560	EM7	;ERROR 7
4534	027140	030632	DH3	
4535	027142	031002	DT3	
4536				
4537	027144	027621	EM8	;ERROR 10
4538	027146	030632	DH3	
4539	027150	031002	DT3	
4540				
4541	027152	027663	EM9	;ERROR 11
4542	027154	030632	DH3	
4543	027156	031002	DT3	
4544				
4545	027160	027721	EM10	;ERROR 12
4546	027162	030632	DH3	
4547	027164	031002	DT3	
4548				
4549	027166	027760	EM13	;ERROR 13
4550	027170	030632	DH3	
4551	027172	031002	DT3	
4552				
4553	027174	030011	EM14	;ERROR 14
4554	027176	030632	DH3	
4555	027200	031002	DT3	
4556				
4557	027202	030043	EM15	;ERROR 15
4558	027204	000000	0	
4559	027206	000000	0	

4560				
4561	027210	030105	EM16	
4562	027212	030632	DH3	
4563	027214	031002	DT3	
4564				
4565	027216	030156	EM17	;ERROR 17
4566	027220	030632	DH3	
4567	027222	031002	DT3	
4568				
4569	027224	030214	EM20	
4570	027226	030632	DH3	
4571	027230	031002	DT3	
4572				
4573	027232	030255	EM21	;ERROR 21
4574	027234	030673	DH5	
4575	027236	031026	DT5	
4576				
4577	027240	030305	EM22	;ERROR 22
4578	027242	030644	DH4	
4579	027244	031010	DT4	
4580				
4581	027246	030347	EM23	;ERROR 23
4582	027250	030632	DH3	
4583	027252	031002	DT3	
4584				
4585	027254	030377	EM24	
4586	027256	030632	DH3	
4587	027260	031002	DT3	
4588				
4589	027262	030425	EM25	
4590	027264	030632	DH3	
4591	027266	031002	DT3	
4592				
4593	027270	030455	EM26	
4594	027272	030632	DH3	
4595	027274	031002	DT3	
4596				
4597	027276	030504	EM27	
4598	027300	030632	DH3	
4599	027302	031002	DT3	

4600
4601

```

027304 047200 020117 046123 EM1: .ASCIZ <200>/NO SLAVE SYNC RESPONSE FROM DZ11 REGISTER/
027357 200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE?
027405 200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
027444 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
027473 200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
027522 042200 030532 020061 EM6: .ASCIZ <200>/DZ11 *RECEIVER BUFFER* ERROR/
027560 052200 040522 051516 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
027621 200 047125 054105 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
027663 200 042522 042503 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
027721 200 047125 054105 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
027760 051600 046111 020117 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
030011 200 044523 047514 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
030043 200 041501 044524 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
030105 200 042522 042101 EM16: .ASCIZ <200>/READING DZRBUF DID NOT CLEAR SILO ALARM/
030156 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
030214 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
030255 200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
030305 200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
030347 200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
030377 200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
030425 200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
030455 200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
030504 043200 046125 020114 EM27: .ASCIZ <200>/FULL BINARY COUNT PATTERN NOT RECEIVED/

030554 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZ11 REG/
030577 200 054105 042520 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
030632 046200 047111 020105 DH3: .ASCIZ <200>/LINE NO./
030644 042600 050130 041505 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
030673 200 054124 046040 DH5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/

```

.EVEN

```

030752 000002 DT1: 2 .DATA TABLES FOR ERROR MESSAGES
030754 006 003 .BYTE 6,3
030756 001204 $REG1
030760 006 001 .BYTE 6,1
030762 001202 $REG0

030764 000003 DT2: 3
030766 006 004 .BYTE 6,4
030770 001214 $REG5
030772 006 001 .BYTE 6,1
030774 001212 $REG4
030776 006 001 .BYTE 6,1
031000 001202 $REG0

031002 000001 DT3: 1
031004 003 001 .BYTE 3,1
031006 001372 $AVLIN

031010 000003 DT4: 3
031012 006 004 .BYTE 6,4
031014 001214 $REG5
031016 006 001 .BYTE 6,1
031020 001212 $REG4

```

```

031022      003      001      .BYTE      3,1
031024      001372      SAVLIN

031026      000004      DTS:      4
031030      003      005      .BYTE      3,5
031032      001372      SAVLIN
031034      006      011      .BYTE      6,9.
031036      001214      $REGS
031040      006      007      .BYTE      6,7
031042      001220      $TMP1
031044      006      001      .BYTE      6,1
031046      001400      REGIST

```

;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

```

031050      002450      DLYTBL: 2450      ;TIME FOR 50 BAUD
031052      001560      1560      ;TIME FOR 75 BAUD
031054      001120      1120      ;TIME FOR 110 BAUD
031056      000750      750      ;TIME FOR 134 BAUD
031060      000660      660      ;TIME FOR 150 BAUD
031062      000330      330      ;TIME FOR 300 BAUD
031064      000150      150      ;TIME FOR 600 BAUD
031066      000060      60      ;TIME FOR 1200 BAUD
031070      000040      40      ;TIME FOR 1800 BAUD
031072      000030      30      ;TIME FOR 2000 BAUD
031074      000020      20      ;TIME FOR 2400 BAUD
031076      000010      10      ;TIME FOR 3600 BAUD
031100      000001      1      ;TIME FOR 4800 BUAD
031102      000001      1      ;TIME FOR 7200 BAUD
031104      000001      1      ;TIME FOR 9600 BAUD
031106      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.

```

031110      002362      CORMAX:  =2362
002362      000240      NOP
002364      000240      NOP
001512      001512      =1512
001512      100000      100000
011572      011572      =11572
011572      105720      TSTB      (RD)+
000001      000001      .END

```


.DCLAS	006672	720	1553#	
.DELAY	006704	714	1557#	
.DEVIC	006652	712	1545#	
.ERRTA	027064	1597	4505#	
.INSTE	006134	698	1385#	
.INSTR	006030	696	1364#	
.INST1	006050	1368#	1388	
.MSG	006052	1366*	1369#	
.PARAM	006154	700	1396#	
.PARMD	025516	716	4306#	
.PAWCH	025712	718	4365#	4377
.RESOS	006414	706	1469#	
.SAVOS	006354	704	1455#	
.SCOPE	004772	322	1161#	2050
.SCOPI	005236	692	1212#	
.SETFL	010500	702	1764#	1783
.START	002150	345	778#	791
.TRPSR	006630	328	1534#	
.TRPTA	002002	688#	1539	
.TYPE	005262	694	1219#	
.\$ASTA=	***** U	1307	1310	
.\$X =	001462	546#	551	

MD-11-DZDZA-E DZDZAE.P11	MACY11 30(1046) 03-OCT-77 09:39	03-OCT-77 09:43 CROSS REFERENCE	PAGE 114 TABLE	MACRO NAMES											SEQ 0140
\$MRESE	1# 2673	2760	2931	2992	3049	3061	3121	3137	3245	3297	3427	3510	3686	3761	
\$MRR	1# 2382														
\$MRAW	1# 2127	2159	2191	2223	2255										
\$MRWD	1# 2463	2489													
\$MSG	1# 1753														
\$PARTS	1# 3743	3800													
\$SCOPE	1# 1144														
\$SETFL	1# 1759														
\$STAG	1# 2700	2793	2951	3204	3358										
\$STAGF	1#														
\$STCR	1# 2287	2329													
\$STLINE	1# 2622														
\$STRPDE	1# 689	691	693	695	697	699	701	703	705	707	709	711	713	715	
\$TSTN	717 1# 2058	2101	2133	2165	2197	2229	2261	2292	2335	2389	2427	2467	2493	2527	
	2585 1# 2627	2667	2754	2836	2923	2987	3056	3132	3240	3291	3421	3505	3660	3754	
	3810 1#														
\$UNIBU	1# 2054														
\$VARIA	1# 349														
\$XZ	1# 2054	2058	2097	2101	2127	2133	2159	2165	2191	2197	2223	2229	2255	2261	
	2287 1# 2292	2329	2335	2382	2389	2419	2427	2463	2467	2489	2493	2516	2526	2575	
	2585 1# 2622	2627	2658	2667	2749	2754	2830	2836	2915	2923	2982	2987	3051	3056	
	3124 1# 3132	3234	3240	3280	3291	3415	3421	3500	3505	3644	3660	3743	3754	3800	
	3810 1#														
\$SCMRE	352# 391	392	393	394	395	396									
\$SCMTM	352# 397	398	399	400											
\$SESCA	132#														
\$SNEWT	132# 2059	2102	2134	2166	2198	2230	2262	2293	2336	2390	2428	2468	2494	2528	
	2586 1# 2628	2668	2755	2837	2924	2988	3057	3133	3241	3292	3422	3506	3661	3755	
	3811 1#														
\$SSKIP	132#														
.EQUAT	1# 22														
.HEADE	1# 3														
.SETUP	1#														
.SACT1	1# 330														
.SAPT8	1# 406#														
.SAPTH	1# 541														
.SAPTY	1# 1303														
.SCATC	1#														
.SCMTA	1# 352														
.SEOP	1# 1087														
.SERRO	1#														
.SPOWE	1# 1711														
.SSCOP	1# 1148														
.STRAP	1#														
.STYPE	1# 1223														

. ABS. 031110 000

ERRORS DETECTED: 0

DZDZAE, DZDZAE/SOL/CRF/NL:TOC=DZDZAE.P11
RUN-TIME: 29 20 2 SECONDS

RUN-TIME RATIO: 246/52=4.6
CORE USED: 36K (71 PAGES)

EOF1DZDZAESEQ

00010000

780223

M11
PDP10 411

1