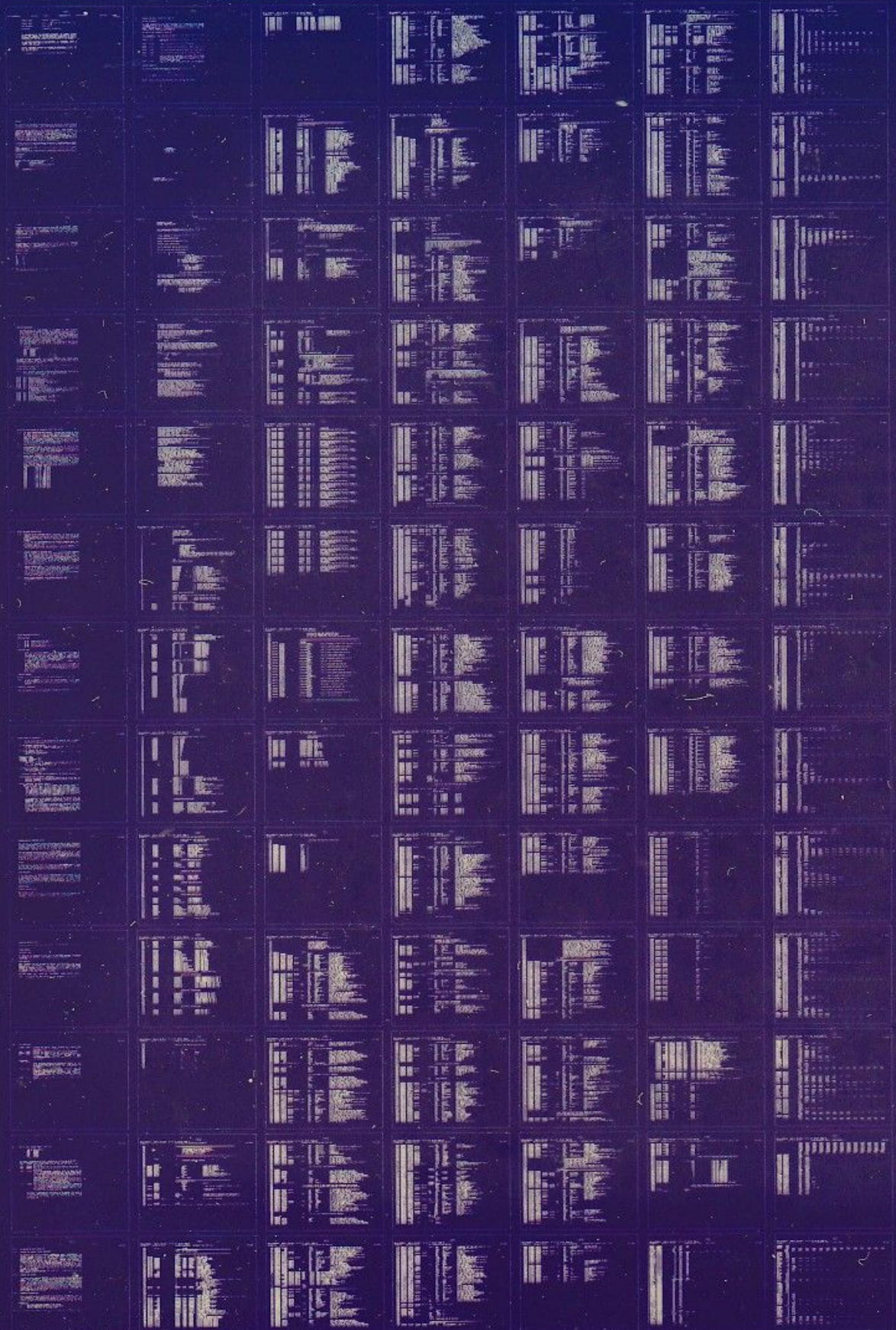


DZV-11

(4) LINE ASYNCHRONOUS MUX
MD-11-DVDZB-A
TESTS, PART 2 OF 2

EP-DVDZB-A-DL-A
COPYRIGHT 1977
FICHE 1 OF 1

OCT 1977
digital
MADE IN USA



120

801

EOF10VDZBRSB0411

00010000

770920

IDENTIFICATION

ZHDR10VDZBASE0

00010000

770920
SEQ 0001

PRODUCT CODE: MAINDEC-11-DVDZB-A-D
PRODUCT NAME: DZV11 4 LINE ASYNC MUX TESTS PART 2 OF 2
DATE RELEASED: APRIL 1977
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) 1977 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

The function of the DZV11 diagnostics is to verify the option operates according to specifications. The diagnostics also verify that the DZV11 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 DZV11 device addresses and vectors only. All remaining parameters will 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 are three standalone diagnostics (DVDZA, DVDZB, and DVDZC) one system module for DEC X/11 (DZBA), and an overlay for ITEX (DVDZD).

DVDZA together with DVDZB will test all logical functions of the DZV11 interface module.

DVDZC is designed as a non-chainable standalone diagnostic providing the operator with direct control over the testing of all DZV11 EIA cables.

2. REQUIREMENTS

2.1 EQUIPMENT

An LSI11 CPU with minimum 4K of memory.
ASR 33 (or equivalent for console)
DZV11 INTERFACE MODULE
H329 Staggered turnaround connector.
H325 Cable turnaround connector.

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

2.2 STORAGE

Program will use all 4K of memory except where ABL and BOOTSTRAP LOADER reside. Location 1500 thru 1740 are especially to be noted and to be untouched by operator after parameters have been input from console (SMO=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 Starting the processor at the Absolute Loader starting address will load the diagnostic into memory.

4. STARTING PROCEDURE

- A. Set SWR to zero for 'AUTO SIZING' or set SW00=1 for user parameter input from console terminal. NOTE: loc. 000176 is used as a software Switch Register in all of the DZV11 diagnostics. (see Sec. 4.1)
On the first startup of the diagnostic if SW07=1 and SW00=0 the program will assume that the status table has been already built from a previous DZV11 diagnostic run. NOTE: any DZV11 diagnostic will overlay the status table when loaded to preserve its contents and thus will not alter a previously built table.
- B. Start the diagnostic at Loc. 200(B). The program will type Maindec and program names (if this was the first start up of the program) and also the following: (on the first program run or if parameters were changed)

```
'MAP OF DZV11 STATUS'
1500 160100
1502 000300
1504 000017
1506 017470
1510 000000
```

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

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

4.1 CONTROL SWITCH SETTINGS

NOTE: This program utilizes a Software Switch Register which may be modified by changing Loc. 176 or by typing Control "G" (fG) on the console terminal while the program is running.

```
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 and
if SW00=0 then the program will assume that the status map
has been built from a previous DZV11 diagnostic run.

SW 06 Set: Reselect DZV11'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
```

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, mode of operation (EXTERNAL, INTERNAL, OR STAGGERED), and the number of DZV11's that are running. Using this switch alone will default the following parameters: all 4 lines are set to be tested on each DZV11, 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 LS111 with MOS memory. When running this program on a processor with a faster memory speed this delay count should be adjusted proportionately 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 DZV11'S DESIRED ACTIVE. A message is typed out on the console terminal asking the operator to type a bit map of the DZV's desired active. Using this switch allows location DZVACTV to be altered (see Sec. 8.3 for a description of this location).
EXAMPLE:
If the devices corresponding to the DZV11's numbered zero, two, and four in the DZV11 Status Map (Loc. 1500 through 1740) are to be tested, type in: 25
This will set bits zero, two, and four in location DZVACTV. All remaining devices in the status map will then not be tested.
- 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 DZV11's; the DZV11 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 DZV11 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.
(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 Go to beginning of the test(on error).
5. SW 10 Go to 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 DZV11 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 - The starting address for any DZV11 diagnostic is Loc. 200

NOTE: If address 000042 is non-zero the program assumes it is under ACT11 or XXDP control and will act accordingly. After *ALL* available DZV11s are tested the program will return to 'XXDP' or 'ACT-11'.

5. OPERATING PROCEDURE

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

5.1 NORMAL START OF DIAGNOSTIC

On the first start of the diagnostic at address 200, if SW00=1 then the following questions are asked and must be answered:

"1ST CSR ADDRESS (160000:163770): "
You must type in the first DZV11 CSR in the system you wish testing to begin at. RANGE: 160000:163770

"1ST VECTOR ADDRESS (300:770): "
You must type in the vector of the first DZV11 in the system under test. RANGE 300:770

"Maintenance Mode
[EXTERNAL <H325> (E)]
[INTERNAL <DZCSR03=1>(I)]
[STAGGERED <H329> (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 DZV11'S <IN OCTAL> (1:20): "
Type total number of DZV11's to be tested in the system. RANGE is 1 thru 20 in octal.

***** IF SW03=1 THEN THE FOLLOWING WILL BE PRINTED *****

"LINES ACTIVE BY BIT <IN OCTAL> (001:017):"
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).

"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 DZV11'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 DZV11's in the system.

If the mode of operation is different for each DZV11 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 DZV11 under test and INDICATE ONE DZV11 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 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 trouble-shooting 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.

6.1 ERROR RECOVERY

If for some reason the DZV11 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 'STSTNM' (address 1246) 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 DZV11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

See section 4.1.2
The status table should be verified regardless of how the 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 DZV11 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.

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 DZV11'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 DVDZB-A CSR: 160100 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.3 KEY LOCATIONS

SLPADR (1252) Contains the address where program will return when iteration count is reached or if loop on test is asserted.

NEXT (1362) Contains the address of the next test to be performed.

STSTNM (1246) Contains the number of the test now being performed.

RUN (1412) The bit in 'RUN' always points one past the DZV11 currently being tested. EXAMPLE: (RUN) 1412/0000000001000000 Means that DZV11 no.5 is the DZV11 now running.

STATUS MAP (1500)-(1740) These locations contain the information needed to test up to 16 (decimal) DZV11s sequentially. they contain the CSR, VECTOR and STATUS concerning the configuration of each DZV11.

DZVACTV(1406) Each bit set in this location indicates that the associated DZV11 will be tested in turn. EXAMPLE: (DZVACTV) 1406/000000000011111 means that DZV11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DZVACTV) 1406/000000000010001 Means that DZ11 no. 00,04 will be tested.

SBASE (1174) Contains the receiver CSR of the current DZV11 under test.

8.4 MORE ON THAT 'STATUS TABLE' (1500-1740)

'MAP OF DZV11 STATUS'	
1500	160100
1502	000300
1504	000017
1506	017470
1510	000000

The above information will be repeated for each of up to 16 DZV11's in the system (these will follow under this table). EXPLANATION:

1500	160100	This is the system control register for the 1st DZV11 in the system.
1502	000300	This is vector 'A' for the first DZV11 in the system.
1504	000017	This is the binary representation of what lines are to be tested.
1506	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 DZV11 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.
1510	000000	This location will contain either all zeros indicating that internal loop was selected as mode of operation or it will contain 100000 indicating that "staggered mode" was selected or it will contain 000200 indicating that "external" was the mode selected.

The above is repeated for each DZV11 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 to suit the specific configuration.

8.5 * METHOD OF AUTO SIZING *******8.5.1 FINDING THE CONTROL STATUS REGISTER.**

The program will start at address 160000 and start '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 163770 is reached. If a 'BUS REPLY' response was issued by the DZV11 (or any other device) (no nzm trap), "MASTER SCAN ENABLE" is attempted to be set and the TCR bits for all four lines are set. "TROY" is then tested to be set and "MASTER SCAN ENABLE" is tested to be still set. The diagnostic will then check that at least one TCR bit is still set. If all of the above worked, this device is assumed to be a DZV11. 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 DZV11, something is wrong and AUTO SIZING should not be done.

8.5.2 FINDING THE VECTOR

The vector area (address 300-776) is filled with the instruction IOT and '+2' (next address). Bit14 and Bit5 (TX INTERRUPT ENABLE AND MSTSCAN ENABLE) are set into the DZVCSR. All TCR bits are set, a delay occurs, and if no interrupt occurs (because of a bad DZV11) the program assumes vector address 300 and the problem should be fixed in the diagnostic. Once the problem is fixed, the program should be setup again to set the correct vector. If an interrupt occurred, the address to which the DZV11 interrupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you, there is a problem and AUTO SIZING should not be done.

8.5.3 PARAMETER ASSUMPTIONS.

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

THEREFORE:

- 1) ALL FOUR LINES ARE ASSUMED TO BE TESTED.
- 2) DEFAULT BAUD RATE IS SET TO 17 (19.2 KBAUD).
- 3) MODE OF OPERATION IS "INTERNAL MODE".

For all parameter adjustments please refer to section 8.4 for greater detail.

9.0 RUNNING THE DZV11 DIAGNOSTIC UNDER APT

9.1.1 THE APT INTERFACE

The DZV diagnostics have been designed to be compatible with the APT (Automated Product Test) system. The DZV logic test diagnostics (DVDZA, and DVDZB) can be run as standalone diagnostics or in either of the APT modes. DVDZC, however, is designed as a standalone diagnostic only and requires direct operator participation.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

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

SSWREG -(1142)	used as the software switch register while running under APT.
SVECT1 -(1170)	used to specify the first vector address
SBASE -(1174)	used to indicate bottom address of DZV11 under test
SDEV1 -(1176)	a bit map representing which DZV11's will be tested
SCDW1 -(1200)	used to indicate which lines to run on all DZV11's
SCDW2 -(1202)	used to indicate the default test mode. Set to 0 for internal testing, 200 for external loop back (H325 installed), or set to 100000 for staggered loop back testing (H329 installed).
SDDW0 -(1204)	each of the SDDW words describes the parameters (LPR) for a particular DZV11, going up to 16 DZV11's

9.1.3 RUNNING UNDER APT

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

NOTE

Be sure SBASE points to the first DZV11 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.

DVDZBA SEQ

C02

DECDOC VER 00.04 27-JUL-77 13:19 PAGE 01

SEQ 0015

DOCUMENT

DVDZBA SEQ

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

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

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

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

51 MISCELLANEOUS DEFINITIONS

63 GENERAL PURPOSE REGISTER DEFINITIONS

75 PRIORITY LEVEL DEFINITIONS

85 "SWITCH REGISTER" SWITCH DEFINITIONS

113 DATA BIT DEFINITIONS (BIT00 TO BIT15)

141 BASIC "CPU" TRAP VECTOR ADDRESSES

358 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

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

371 MEM.LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABO

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

462 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:

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

1010 INCREMENT THE PASS NUMBER (\$PASS)
 IF THERES A MONITOR GO TO IT
 IF THERE ISN'T JUMP TO CYCLE

1077 THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 AND LOAD THE TEST NUMBER(\$TSTNM) 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

1147 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

1931 ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
 IF BIT7 IN THE ENVIRONMENT MODE (\$ENVM) BYTE IS SET,
 THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.

1963 ROUTINE USED TO "AUTO SIZE" THE DZV11
 CSR AND VECTOR.
 NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
 ADDRESS RANGE (160000:163770)
 AND THE VECTOR MAY BE ANY WHERE IN THE
 FLOATING VECTOR RANGE (300:770)

2071 ***** TEST 1 *****
 THIS TEST VERIFIES OVERRUN AND SILO ALARM
 ONE LINE AT A TIME - BASED UPON VALID LINES
 AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
 TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
 EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
 SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
 CHAR PULLED OUT OF THE SILO.
 ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
 USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
 ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
 USED TO SCOPE SILO ALARM PULSES, ETC.

- 2192 ***** TEST 2 *****
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.
ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
- 2264 ***** TEST 3 *****
THIS TEST RUNS ALL LINES FULL BORE
BASED UPON QUALIFIED LINES
..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
TRANSMITTER
- 2397 ***** TEST 4 *****
DZV11 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 THEN TESTED.
WHEN RUNNING UNDER THE APT MANUFACTURING SYSTEM
THIS TEST IS ONLY RUN THE FIRST PASS
- 2491 ***** TEST 5 *****
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.
THIS TEST FIRST CHECKS EVEN PARITY FOR ODD LINES AND
ODD PARITY FOR EVEN LINES, THEN IT CHECKS THE REVERSE.

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

000001

001120

000011
000012
000015
000200
177776

```

.TITLE MD-11-DVDZB-A
.*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
      .STARTING PROCEDURE
      .LOAD PROGRAM
      .LOAD ADDRESS 000200
      .PRESS START
      .PROGRAM WILL TYPE
      ."MAINDEC-11-DVDZBA/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 OF 2"
      .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

.REM
:SWITCH REGISTER OPTIONS
:-----

SW15=100000      ;=1, HALT ON ERROR
SW14=40000       ;=1, LOOP ON CURRENT TEST
SW13=20000       ;=1, INHIBIT ERROR TIMEOUT
SW12=10000       ;=1, DELETE TIMEOUT/BELL ON ERROR.
SW11=4000        ;=1, INHIBIT ITERATIONS
SW10=2000        ;=1, ESCAPE TO NEXT TEST ON ERROR
SW09=1000        ;=1, LOOP WITH CURRENT DATA
SW08=400         ;=1, LOOP ON ERROR
SW07=200         ;=1, DO "AUTO SIZING" ON INITIAL START UP.
SW06=100        ;=1, DESELECT SPECIFIC DEVICES
                ;NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT

SW05=40
SW04=20          ;=1, SELECT DELAY PARAMETER
SW03=10          ;=1, SELECT SPECIFIC PARAMETERS
SW02=4           ;=1, LOCK ON TEST SELECT
SW01=2           ;=1, RESTART PROGRAM AT SELECTED TEST
SW00=1          ;=1, SELECT DEVICE ADDRESS, VECTOR, ETC.

.SBTTL BASIC DEFINITIONS

.*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
HT= 11             ;;CODE FOR HORIZONTAL TAB
LF= 12             ;;CODE FOR LINE FEED
CR= 15             ;;CODE FOR CARRIAGE RETURN
CRLF= 200         ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776        ;;PROCESSOR STATUS WORD

```

```

57      .EQUIV PS,PSW
58      177774 STKLMT= 177774      ;; STACK LIMIT REGISTER
59      177772 PIRQ= 177772      ;; PROGRAM INTERRUPT REQUEST REGISTER
60      177570 DSWR= 177570      ;; HARDWARE SWITCH REGISTER
61      177570 DDISP= 177570     ;; HARDWARE DISPLAY REGISTER
62
63      .*GENERAL PURPOSE REGISTER DEFINITIONS
64      000000 R0= %0           ;; GENERAL REGISTER
65      000001 R1= %1           ;; GENERAL REGISTER
66      000002 R2= %2           ;; GENERAL REGISTER
67      000003 R3= %3           ;; GENERAL REGISTER
68      000004 R4= %4           ;; GENERAL REGISTER
69      000005 R5= %5           ;; GENERAL REGISTER
70      000006 R6= %6           ;; GENERAL REGISTER
71      000007 R7= %7           ;; GENERAL REGISTER
72      000006 SP= %6           ;; STACK POINTER
73      000007 PC= %7           ;; PROGRAM COUNTER
74
75      .*PRIORITY LEVEL DEFINITIONS
76      000000 PR0= 0           ;; PRIORITY LEVEL 0
77      000040 PR1= 40          ;; PRIORITY LEVEL 1
78      000100 PR2= 100         ;; PRIORITY LEVEL 2
79      000140 PR3= 140         ;; PRIORITY LEVEL 3
80      000200 PR4= 200         ;; PRIORITY LEVEL 4
81      000240 PR5= 240         ;; PRIORITY LEVEL 5
82      000300 PR6= 300         ;; PRIORITY LEVEL 6
83      000340 PR7= 340         ;; PRIORITY LEVEL 7
84
85      .*"SWITCH REGISTER" SWITCH DEFINITIONS
86      100000 SW15= 100000
87      040000 SW14= 40000
88      020000 SW13= 20000
89      010000 SW12= 10000
90      004000 SW11= 4000
91      002000 SW10= 2000
92      001000 SW09= 1000
93      000400 SW08= 400
94      000200 SW07= 200
95      000100 SW06= 100
96      000040 SW05= 40
97      000020 SW04= 20
98      000010 SW03= 10
99      000004 SW02= 4
100     000002 SW01= 2
101     000001 SW00= 1
102     .EQUIV SW09,SW9
103     .EQUIV SW08,SW8
104     .EQUIV SW07,SW7
105     .EQUIV SW06,SW6
106     .EQUIV SW05,SW5
107     .EQUIV SW04,SW4
108     .EQUIV SW03,SW3
109     .EQUIV SW02,SW2
110     .EQUIV SW01,SW1
111     .EQUIV SW00,SW0
112

```

```

113      .: *DATA BIT DEFINITIONS (BIT00 TO BIT15)
114      100000      BIT15= 100000
115      040000      BIT14= 40000
116      020000      BIT13= 20000
117      010000      BIT12= 10000
118      004000      BIT11= 4000
119      002000      BIT10= 2000
120      001000      BIT09= 1000
121      000400      BIT08= 400
122      000200      BIT07= 200
123      000100      BIT06= 100
124      000040      BIT05= 40
125      000020      BIT04= 20
126      000010      BIT03= 10
127      000004      BIT02= 4
128      000002      BIT01= 2
129      000001      BIT00= 1
130      .EQUIV      BIT09, BIT9
131      .EQUIV      BIT08, BIT8
132      .EQUIV      BIT07, BIT7
133      .EQUIV      BIT06, BIT6
134      .EQUIV      BIT05, BIT5
135      .EQUIV      BIT04, BIT4
136      .EQUIV      BIT03, BIT3
137      .EQUIV      BIT02, BIT2
138      .EQUIV      BIT01, BIT1
139      .EQUIV      BIT00, BIT0
140
141      .: *BASIC "CPU" TRAP VECTOR ADDRESSES
142      000004      ERRVEC= 4      ;: TIME OUT AND OTHER ERRORS
143      000010      RESVEC= 10     ;: RESERVED AND ILLEGAL INSTRUCTIONS
144      000014      TBITVEC=14     ;: "T" BIT
145      000014      TRTVEC= 14     ;: TRACE TRAP
146      000014      BPTVEC= 14     ;: BREAKPOINT TRAP (BPT)
147      000020      IOTVEC= 20     ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
148      000024      PWRVEC= 24     ;: POWER FAIL
149      000030      EMTVEC= 30     ;: EMULATOR TRAP (EMT) **ERROR**
150      000034      TRAPVEC=34     ;: "TRAP" TRAP
151      000060      TKVEC= 60      ;: TTY KEYBOARD VECTOR
152      000064      TPVEC= 64      ;: TTY PRINTER VECTOR
153      000240      PIRQVEC=240    ;: PROGRAM INTERRUPT REQUEST VECTOR
154
155
156      .: INSTRUCTION DEFINITIONS
157      -----
158
159      005746      PUSH1SP=5746    ;: DECREMENT PROCESSOR STACK 1 WORD
160      005726      POP1SP=5726    ;: INCREMENT PROCESSOR STACK 1 WORD
161      010046      PUSHRO=10046    ;: SAVE R0 ON STACK
162      012600      POPRO=12600     ;: RESTORE R0 FROM STACK
163      024646      PUSH2SP=24646  ;: DECREMENT STACK TWICE
164      022626      POP2SP=22626   ;: INCREMENT STACK TWICE
165      000200      MASK=BIT7      ;: SET INTERRUPT MASK (INHIBIT FURTHER INTERRUPTS)
166      000000      CLEAR=0        ;: ALLOW INTERRUPTS (CLEAR PROCESSOR STATUS)
167
168

```

```

169                                     ;DZV11 CONTROL AND STATUS REGISTER DEFINITIONS
170                                     ;(DZVCSR)      BIT DEFINITIONS
171                                     ;-----
172
173      000010      MAINT = BIT3      ; MAINTENANCE MODE ENABLE
174      000020      DCLR=BIT4      ; DEVICE CLEAR
175      000040      MSENAB=BIT5     ; MASTER SCAN ENABLE
176      000100      RIE=BIT6      ; RECEIVER INTERRUPT ENABLE
177      000200      RDONE=BIT7     ; RECEIVER DONE
178      010000      SILOEN= BIT12   ; SILO ALARM ENABLE
179      020000      SILOAL = BIT13  ; SILO ALARM
180      040000      TIE=BIT14     ; TRANSMITTER INTERRUPT ENABLE
181      100000      TROY=BIT15     ; TRANSMITTER READY
182
183                                     ;DZVCSR WORD DEFINITIONS
184                                     ;-----
185
186      000000      TLO=0          ; TRANSMIT LINE 0
187      000400      TL1=BIT8      ; TRANSMIT LINE 1
188      001000      TL2=BIT9      ; TRANSMIT LINE 2
189      001400      TL3=BIT9!BIT8 ; TRANSMIT LINE 3
190
191                                     ;DZVRBUF BIT DEFINITIONS
192                                     ;-----
193
194      010000      PARER=BIT12     ; PARITY ERROR
195      020000      FRMERR=BIT13   ; FRAME ERROR
196      040000      OVRUN=BIT14    ; OVERRUN ERROR
197      100000      DVALID=BIT15   ; DATA VALID
198
199                                     ;DZVRBUF WORD DEFINITIONS
200                                     ;-----
201
202      000000      RLO=0          ; RECEIVER LINE 0
203      000400      RL1=BIT8      ; RECEIVER LINE 1
204      001000      RL2=BIT9      ; RECEIVER LINE 2
205      001400      RL3=BIT9!BIT8 ; RECEIVER LINE 3
206
207                                     ;DZVLPR WORD DEFINITIONS
208                                     ;-----
209
210      000000      LPO=0          ; LINE PARAMETER 0
211      000001      LP1=BIT0      ; LINE PARAMETER 1
212      000002      LP2=BIT1      ; LINE PARAMETER 2
213      000003      LP3=BIT1!BIT0 ; LINE PARAMETER 3
214
215      000000      FIVE=0        ; FIVE BITS/CHAR, 1 STOP BIT
216      000010      SIX=BIT3      ; SIX BITS/CHAR, 1 STOP BIT
217      000020      SEVEN=BIT4     ; SEVEN BITS/CHAR, 1 STOP BIT
218      000030      EIGHT=BIT4!BIT3 ; EIGHT BITS/CHAR, 1 STOP BIT
219      000040      FIVES=BIT5     ; FIVE BITS/CHAR, 2 STOP BITS
220      000050      SIXS=BIT5!BIT3 ; SIX BITS/CHAR, 2 STOP BITS
221      000060      SEVENS=BIT5!BIT4 ; SEVEN BITS/CHAR, 2 STOP BITS
222      000070      EIGHTS=BIT5!BIT4!BIT3 ; EIGHT BITS/CHAR, 2 STOP BITS
223
224      000100      PARITY=BIT6     ; PARITY ENABLED
    
```

```

225 000200 000PAR=BIT7 ;000 PARITY ENABLED
226 000000 0NESTOP=0 ;ONE STOP BIT ENABLED
227 000040 2WOSTOP=BITS ;TWO STOP BITS ENABLED
228 000000 EVEPAR=0 ;EVEN PARITY ENABLED
229 010000 RCVON=BIT12 ;ENABLE RECEIVER (RECEIVER ON)

230
231 000000 S50=0 ;SPEED 50 BAUD
232 000400 S75=BIT8 ;SPEED 75 BAUD
233 001000 S110=BIT9 ;SPEED 110 BAUD
234 001400 S134=BIT9!BIT8 ;SPEED 134.5 BAUD
235 002000 S150=BIT10 ;SPEED 150 BAUD
236 002400 S300=BIT10!BIT8 ;SPEED 300 BAUD
237 003000 S600=BIT10!BIT9 ;SPEED 600 BAUD
238 003400 S1200=BIT10!BIT9!BIT8 ;SPEED 1200 BAUD
239 004000 S1800=BIT11 ;SPEED 1800 BAUD
240 004400 S2000=BIT11!BIT8 ;SPEED 2000 BAUD
241 005000 S2400=BIT11!BIT9 ;SPEED 2400 BAUD
242 005400 S3600=BIT11!BIT9!BIT8 ;SPEED 3600 BAUD
243 006000 S4800=BIT11!BIT10 ;SPEED 4800 BAUD
244 006400 S7200=BIT11!BIT10!BIT8 ;SPEED 7200 BAUD
245 007000 S9600=BIT11!BIT10!BIT9 ;SPEED 9600 BAUD
246 007400 S19200=BIT11!BIT10!BIT9!BIT8 ;SPEED 19200 BAUD
247
248 ;DZVTCR BIT DEFINITIONS
249 -----
250 000001 TCR0=BIT0 ;ENABLE TRANSMISSION ON LINE 0
251 000002 TCR1=BIT1 ;ENABLE TRANSMISSION ON LINE 1
252 000004 TCR2=BIT2 ;ENABLE TRANSMISSION ON LINE 2
253 000010 TCR3=BIT3 ;ENABLE TRANSMISSION ON LINE 3
254 000400 DTR0=BIT8 ;DATA TERMINAL READY FOR LINE 0
255 001000 DTR1=BIT9 ;DATA TERMINAL READY FOR LINE 1
256 002000 DTR2=BIT10 ;DATA TERMINAL READY FOR LINE 2
257 004000 DTR3=BIT11 ;DATA TERMINAL READY FOR LINE 3
258
259 ;DZVMSR BIT DEFINITIONS
260 -----
261 000001 RING0=BIT0 ;RING INDICATED ON LINE 0
262 000002 RING1=BIT1 ;RING INDICATED ON LINE 1
263 000004 RING2=BIT2 ;RING INDICATED ON LINE 2
264 000010 RING3=BIT3 ;RING INDICATED ON LINE 3
265 000400 CO0=BIT8 ;CARRIER PRESENT ON LINE 0
266 001000 CO1=BIT9 ;CARRIER PRESENT ON LINE 1
267 002000 CO2=BIT10 ;CARRIER PRESENT ON LINE 2
268 004000 CO3=BIT11 ;CARRIER PRESENT ON LINE 3
269
270 ;DZVTDR BIT DEFINITIONS
271 -----
272
273 000400 BRK0=BIT8 ;BREAK FOR LINE 0
274 001000 BRK1=BIT9 ;BREAK FOR LINE 1
275 002000 BRK2=BIT10 ;BREAK FOR LINE 2
276 004000 BRK3=BIT11 ;BREAK FOR LINE 3

```


277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294

TABLE OF LOOP AROUND FUNCTIONS (H325)

I	↑
V	↑
REC	TRANS
DATA	DATA

I	↑
V	↑
CO	RTS

I	↑
V	↑
RING	DTR

```

295 ;*****
296 -----
297 ; TRAPCATCHER FOR ILLEGAL INTERRUPTS
298 ; THE STANDARD "TRAP CATCHER" IS PLACED
299 ; BETWEEN ADDRESS 0 TO ADDRESS 776.
300 ; IT LOOKS LIKE "PC+2 HALT".
301 -----
302 ;*****
303
304 .=0
305 ; STANDARD INTERRUPT VECTORS
306 -----
307
308 .=20
309 .SCOPE .SCOPE LOOP HANDLER
310 000020 004300 ; ...BLE AT PRIORITY 7
311 000022 000200 MASK
312 000024 007236 $PWROD ; POWER FAIL HANDLER
313 000026 000340 340 ; SERVICE AT PRIORITY LEVEL 7
314 000030 006344 $ERROR ; ERROR HANDLER
315 000032 000340 340 ; SERVICE AT PRIORITY LEVEL 7
316 000034 006136 .TRPSPV ; GENERAL HANDLER DISPATCH SERVICE
317 000036 000340 340 ; SERVICE AT PRIORITY LEVEL 7
318 .SBTTL ACT11 HOOKS
319
320 ;*****
321 ; HOOKS REQUIRED BY ACT11
322 $SVPC=. ; SAVE PC
323 000046 004234 .=46 $ENDAD ;; 1) SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
324 000052 000052 .=52 ; ; 2) SET LOC.52 TO ZERO
325 000052 000000 .WORD 0 ;; RESTORE PC
326 000040 .=$SVPC
327
328 .=174
329 000174 000000 DISPREG:0 ; SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 115
330 000176 000000 SWREG: 0 ; SOFTWARE SWITCH REGISTER FOR SWITCHLESS 115
331 000200 000200 .=200
332 000200 000137 002116 JMP .START ; GO TO START OF PROGRAM
333
334
335 .=1000
336 001000 005200 040515 047111 MTITLE: .ASCIZ <200><12>/MAINDEC-11-DVDZBA/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 OF 2
(2)

```

```

337      001120      .=1120
338      ;:*****
339      .SBTTL  APT MAILBOX-ETABLE
340      ;:*****
341      .EVEN
342
343      001120      $MAIL:          ;: APT MAILBOX
344      001120      000000      $MSGTY: .WORD  AMSGTY ;: MESSAGE TYPE CODE
345      001122      000000      $FATAL: .WORD  AFATAL ;: FATAL ERROR NUMBER
346      001124      000000      $TESTN: .WORD  ATESTN ;: TEST NUMBER
347      001126      000000      $PASS:  .WORD  APASS  ;: PASS COUNT
348      001130      000000      $DEVCT: .WORD  ADEVCT ;: DEVICE COUNT
349      001132      000000      $UNIT:  .WORD  AUNIT  ;: I/O UNIT NUMBER
350      001134      000000      $MSGAD: .WORD  AMSGAD ;: MESSAGE ADDRESS
351      001136      000000      $MSGLG: .WORD  AMSGLG ;: MESSAGE LENGTH
352      001140      $ETABLE: ;: APT ENVIRONMENT TABLE
353      001140      000      $ENV:   .BYTE  AENV   ;: ENVIRONMENT BYTE
354      001141      000      $ENVH: .BYTE  AENVH  ;: ENVIRONMENT MODE BITS
355      001142      000000      $SWREG: .WORD  ASWREG ;: APT SWITCH REGISTER
356      001144      000000      $USWR:  .WORD  AUSWR  ;: USER SWITCHES
357      001146      000000      $CPUOP: .WORD  ACPUOP ;: CPU TYPE, OPTIONS
358      ;:
359      ;:      BITS 15-11=CPU TYPE
360      ;:      11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
361      ;:      11/70=06, PQQ=07, Q=10
362      ;:
363      ;:      BIT 10=REAL TIME CLOCK
364      ;:      BIT 9=FLOATING POINT PROCESSOR
365      ;:      BIT 8=MEMORY MANAGEMENT
366      001150      000      $MAMS1: .BYTE  AMAMS1 ;: HIGH ADDRESS, M.S. BYTE
367      001151      000      $MTYP1: .BYTE  AMTYP1 ;: MEM. TYPE, BLK#1
368      ;:      MEM. TYPE BYTE -- (HIGH BYTE)
369      ;:      900 NSEC CORE=001
370      ;:      300 NSEC BIPOLAR=002
371      ;:      500 NSEC MOS=003
372      001152      000000      $MADR1: .WORD  AMADR1 ;: HIGH ADDRESS, BLK#1
373      ;:      MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
374      001154      000      $MAMS2: .BYTE  AMAMS2 ;: HIGH ADDRESS, M.S. BYTE
375      001155      000      $MTYP2: .BYTE  AMTYP2 ;: MEM. TYPE, BLK#2
376      001156      000000      $MADR2: .WORD  AMADR2 ;: MEM. LAST ADDRESS, BLK#2
377      001160      000      $MAMS3: .BYTE  AMAMS3 ;: HIGH ADDRESS, M.S. BYTE
378      001161      000      $MTYP3: .BYTE  AMTYP3 ;: MEM. TYPE, BLK#3
379      001162      000000      $MADR3: .WORD  AMADR3 ;: MEM. LAST ADDRESS, BLK#3
380      001164      000      $MAMS4: .BYTE  AMAMS4 ;: HIGH ADDRESS, M.S. BYTE
381      001165      000      $MTYP4: .BYTE  AMTYP4 ;: MEM. TYPE, BLK#4
382      001166      000000      $MADR4: .WORD  AMADR4 ;: MEM. LAST ADDRESS, BLK#4
383      001170      000300      $VECT1: .WORD  AVECT1 ;: INTERRUPT VECTOR#1, BUS PRIORITY#1
384      001172      000000      $VECT2: .WORD  AVECT2 ;: INTERRUPT VECTOR#2, BUS PRIORITY#2
385      001174      160010      $BASE:  .WORD  ABASE  ;: BASE ADDRESS OF EQUIPMENT UNDER TEST
386      001176      000001      $DEVH:  .WORD  ADEVH  ;: DEVICE MAP
387      001200      000017      $CDW1:  .WORD  ACDW1  ;: CONTROLLER DESCRIPTION WORD#1
388      001202      000000      $CDW2:  .WORD  ACDW2  ;: CONTROLLER DESCRIPTION WORD#2
389      001204      017470      $DDW0:  .WORD  ADDW0  ;: DEVICE DESCRIPTOR WORD#0
390      001206      017470      $DDW1:  .WORD  ADDW1  ;: DEVICE DESCRIPTOR WORD#1
391      001210      017470      $DDW2:  .WORD  ADDW2  ;: DEVICE DESCRIPTOR WORD#2
392      001212      017470      $DDW3:  .WORD  ADDW3  ;: DEVICE DESCRIPTOR WORD#3
393      001214      017470      $DDW4:  .WORD  ADDW4  ;: DEVICE DESCRIPTOR WORD#4
394      001216      017470      $DDW5:  .WORD  ADDW5  ;: DEVICE DESCRIPTOR WORD#5

```


407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459

001244
001244 000000
001246 000
001247 000
001250 000000
001252 000000
001254 000000
001256 000000
001260 000
001261 001
001262 000000
001264 000000
001266 000000
001270 000000
001272 000000
001274 000000
001276 000000
001300 000
001301 000
001302 000000
001304 177570
001306 177570
001310 177560
001312 177562
001314 177564
001316 177566
001320 000
001321 002
001322 012
001323 000
001324 000000
001326 000000
001330 000000
001332 000000
001334 000000
001336 000000
001340 000000
001342 000000
001344 000000
001346 000000
001350 000000
001352 000000
001354 000000
001356 077
001357 015
001360 000012

.SBTTL COMMON TAGS

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

SCMTAG: ; START OF COMMON TAGS
STSTNM: .WORD 0 ; CONTAINS THE TEST NUMBER
SERFLG: .BYTE 0 ; CONTAINS ERROR FLAG
SICNT: .WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
SLPADR: .WORD 0 ; CONTAINS SCOPE LOOP ADDRESS
SLPERR: .WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
SERTTL: .WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 0 ; CONTAINS ITEM CONTROL BYTE
SERMAX: .BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
SERRPC: .WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 0 ; CONTAINS ADDRESS OF 'GOOD' DATA
\$BDAOR: .WORD 0 ; CONTAINS ADDRESS OF 'BAD' DATA
\$GDOAT: .WORD 0 ; CONTAINS 'GOOD' DATA
\$BDOAT: .WORD 0 ; CONTAINS 'BAD' DATA
; RESERVED--NOT TO BE USED
SAUTOB: .BYTE 0 ; AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0 ; INTERRUPT MODE INDICATOR
SWR: .WORD DSWR ; ADDRESS OF SWITCH REGISTER
DISPLAY: .WORD DDISP ; ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 ; TTY KBD STATUS
\$TKB: 177562 ; TTY KBD BUFFER
\$TPS: 177564 ; TTY PRINTER STATUS REG. ADDRESS
\$TPB: 177566 ; TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12 ; INSERT FILL CHARS. AFTER A "LINE FEED"
\$TPFLG: .BYTE 0 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$REGAD: .WORD 0 ; CONTAINS THE ADDRESS FROM WHICH (\$REG0) WAS OBTAINED
\$REG0: .WORD 0 ; CONTAINS ((SREGAD)+0)
\$REG1: .WORD 0 ; CONTAINS ((SREGAD)+2)
\$REG2: .WORD 0 ; CONTAINS ((SREGAD)+4)
\$REG3: .WORD 0 ; CONTAINS ((SREGAD)+6)
\$REG4: .WORD 0 ; CONTAINS ((SREGAD)+10)
\$REG5: .WORD 0 ; CONTAINS ((SREGAD)+12)
\$TMP0: .WORD 0 ; USER DEFINED
\$TMP1: .WORD 0 ; USER DEFINED
\$TMP2: .WORD 0 ; USER DEFINED
\$TMP3: .WORD 0 ; USER DEFINED
\$TMP4: .WORD 0 ; USER DEFINED
\$TIMES: 0 ; MAX. NUMBER OF ITERATIONS
\$QUES: .ASCII /?/ ; QUESTION MARK
\$CRLF: .ASCII <15> ; CARRIAGE RETURN
\$LF: .ASCII <12> ; LINE FEED

460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

001362

001362 000000
001364 000000

001366 000017
001370 017470
001372 000000
001374 000000
001376 000000
001400 000000
001402 000000
001404 000000
001406 000001
001410 000001
001412 000001
001414 000001
001415 001
001416 000001
001420
001420 001500

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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

\$ERRTB:

;PROGRAM CONTROL PARAMETERS
;-----

NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP

;PROGRAM VARIABLES
;-----

LINE: 17 ;DEFAULT ALL FOUR 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
DZVACTV: .BLKW 1 ;*DZV11'S SELECTED ACTIVE.
SAVACTV: .BLKW 1 ;*A BIT MAP OF DZV11'S IN THE SYSTEM
RUN: 1 ;*POINTER ONE PAST RUNNING DEVICE.
DZVNUM: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM.
SAVNUM: .BYTE 1 ;*WORKABLE NUMBER.
SAVNO: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S BEING TESTED
.EVEN
ACTIVE: DZV.MAP ;TABLE POINTER.

501					
502					
503					
504					
505	001422	000	INIFLG: .BYTE	0	; PROGRAM INITIALIZATION FLAG
506	001423	000	HDRFLG: .BYTE	0	; PROGRAM INITIALIZATION FLAG FOR HEADER MAP
507	001424	000	MNTFLG: .BYTE	0	; MAINTENANCE BIT SET FLAG
508	001425	000	DONFLG: .BYTE	0	; TRANSMISSION COMPLETION FLAG
509			.EVEN		
510			.DATA VARIABLES		
511	001426	000000	T00: .WORD	0	
512	001430	000000	T01: .WORD	0	
513	001432	000000	T02: .WORD	0	
514	001434	000000	T03: .WORD	0	
515	001436	000000	TR0: .WORD	0	
516	001440	000000	TR1: .WORD	0	
517	001442	000000	TR2: .WORD	0	
518	001444	000000	TR3: .WORD	0	
519	001446		STOP:		
520			.SBTTL		APT PARAMETER BLOCK
521					
522					
523					
524					
525		001446			
526		000024	.SX=.		; SAVE CURRENT LOCATION
527	000024	000200	.=24		; SET POWER FAIL TO POINT TO START OF PROGRAM
528		000044	200		; FOR APT START UP
529	000044	001446	.=44		; POINT TO APT INDIRECT ADDRESS PNTR.
530		001446	SAPTHDR		; POINT TO APT HEADER BLOCK
531			.=.SX		; RESET LOCATION COUNTER
532					
533					
534					
535	001446		SAPTHD:		
536	001446	000000	SHIBTS: .WORD	0	; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
537	001450	001120	SMBADR: .WORD	SMAIL	; ADDRESS OF APT MAILBOX (BITS 0-15)
538	001452	000132	SPSTM: .WORD	90.	; RUN TIM OF LONGEST TEST
539	001454	000137	SASTM: .WORD	95.	; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
540	001456	000000	SUNITM: .WORD	0.	; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
541	001460	000052	.WORD	SETEND-SMAIL/2	; LENGTH MAILBOX-ETABLE(WORDS)
542					; DZV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
543					
544					
545		001500	.=1500		
546	001500		DZV.MAP:		
547					
548	001500	000001	DZCR0: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 0
549	001502	000001	DZVC0: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 0
550	001504	000001	LINE0: .BLKW	1	; ALL LINES SELECTED
551	001506	000001	PAR0: .BLKW	1	; PARAMETERS
552	001510	000001	MANT0: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
553					
554	001512	000001	DZCR1: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 1
555	001514	000001	DZVC1: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 1
556	001516	000001	LINE1: .BLKW	1	; ALL LINES SELECTED

557	001520	000001	PAR1:	.BLKW	1	:PARAMETERS
558	001522	000001	MANT1:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
559						
560	001524	000001	DZCR2:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 2
561	001526	000001	DZVC2:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 2
562	001530	000001	LINE2:	.BLKW	1	:ALL LINES SELECTED
563	001532	000001	PAR2:	.BLKW	1	:PARAMETERS
564	001534	000001	MANT2:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
565						
566	001536	000001	DZCR3:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 3
567	001540	000001	DZVC3:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 3
568	001542	000001	LINE3:	.BLKW	1	:ALL LINES SELECTED
569	001544	000001	PAR3:	.BLKW	1	:PARAMETERS
570	001546	000001	MANT3:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
571						
572	001550	000001	DZCR4:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 4
573	001552	000001	DZVC4:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 4
574	001554	000001	LINE4:	.BLKW	1	:ALL LINES SELECTED
575	001556	000001	PAR4:	.BLKW	1	:PARAMETERS
576	001560	000001	MANT4:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
577						
578	001562	000001	DZCR5:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 5
579	001564	000001	DZVC5:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 5
580	001566	000001	LINE5:	.BLKW	1	:ALL LINES SELECTED
581	001570	000001	PAR5:	.BLKW	1	:PARAMETERS
582	001572	000001	MANT5:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
583						
584	001574	000001	DZCR6:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 6
585	001576	000001	DZVC6:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 6
586	001600	000001	LINE6:	.BLKW	1	:ALL LINES SELECTED
587	001602	000001	PAR6:	.BLKW	1	:PARAMETERS
588	001604	000001	MANT6:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
589						
590	001606	000001	DZCR7:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 7
591	001610	000001	DZVC7:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 7
592	001612	000001	LINE7:	.BLKW	1	:ALL LINES SELECTED
593	001614	000001	PAR7:	.BLKW	1	:PARAMETERS
594	001616	000001	MANT7:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
595						
596	001620	000001	DZCR10:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 10
597	001622	000001	DZVC10:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 10
598	001624	000001	LINE10:	.BLKW	1	:ALL LINES SELECTED
599	001626	000001	PAR10:	.BLKW	1	:PARAMETERS
600	001630	000001	MANT10:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
601						
602	001632	000001	DZCR11:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 11
603	001634	000001	DZVC11:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 11
604	001636	000001	LINE11:	.BLKW	1	:ALL LINES SELECTED
605	001640	000001	PAR11:	.BLKW	1	:PARAMETERS
606	001642	000001	MANT11:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
607						
608	001644	000001	DZCR12:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 12
609	001646	000001	DZVC12:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 12
610	001650	000001	LINE12:	.BLKW	1	:ALL LINES SELECTED
611	001652	000001	PAR12:	.BLKW	1	:PARAMETERS
612	001654	000001	MANT12:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE

G03

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 14
DVDZBA.P11 28-JUL-77 07:37 APT PARAMETER BLOCK

SEQ 0032

613					
614	001656	000001	DZCR13: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 13
615	001660	000001	DZVC13: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 13
616	001662	000001	LINE13: .BLKW	1	; ALL LINES SELECTED
617	001664	000001	PAR13: .BLKW	1	; PARAMETERS
618	001666	000001	MANT13: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
619					
620	001670	000001	DZCR14: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 14
621	001672	000001	DZVC14: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 14
622	001674	000001	LINE14: .BLKW	1	; ALL LINES SELECTED
623	001676	000001	PAR14: .BLKW	1	; PARAMETERS
624	001700	000001	MANT14: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
625					
626	001702	000001	DZCR15: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 15
627	001704	000001	DZVC15: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 15
628	001706	000001	LINE15: .BLKW	1	; ALL LINES SELECTED
629	001710	000001	PAR15: .BLKW	1	; PARAMETERS
630	001712	000001	MANT15: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
631					
632	001714	000001	DZCR16: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 16
633	001716	000001	DZVC16: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 16
634	001720	000001	LINE16: .BLKW	1	; ALL LINES SELECTED
635	001722	000001	PAR16: .BLKW	1	; PARAMETERS
636	001724	000001	MANT16: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
637					
638	001726	000001	DZCR17: .BLKW	1	; CONTROL STATUS REGISTER FOR DZV11 NUMBER 17
639	001730	000001	DZVC17: .BLKW	1	; RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 17
640	001732	000001	LINE17: .BLKW	1	; ALL LINES SELECTED
641	001734	000001	PAR17: .BLKW	1	; PARAMETERS
642	001736	000001	MANT17: .BLKW	1	; MAINTENANCE MODE FOR THIS DEVICE
643					
644	001740	177777	DZV.END:	177777	

645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692

;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	
SHIFT=TRAP+20	;CALL TO ROTATE LINE POINTER
.SHIFT	
LPRSET=TRAP+21	;CALL TO SET UP LPR DEVICE REGISTER
.LPRSET	
BUFSET=TRAP+22	;CALL TO ZERO BUFFER AREA
.BUFSET	

;:*****

```

693                                     ;DZV11 VECTOR AND REGISTER INDIRECT POINTERS
694                                     ;WORKING AREA
695
696 002010 160040 DZVCSR: 160040 ;R/W
697 002012 160041 HDZVCSR: 160041 ;R/W
698 002014 160042 DZVRBUF: 160042 ;READ ONLY
699 002016 160043 HDZVRBUF: 160043 ;READ ONLY
700 002020 160042 DZVLPR: 160042 ;WRITE ONLY
701 002022 160043 HDZVLPR: 160043 ;WRITE ONLY
702 002024 160044 DZVTCR: 160044 ;R/W
703 002026 160045 HDZVTCR: 160045 ;R/W
704 002030 160046 DZVMSR: 160046 ;READ ONLY
705 002032 160047 HDZVMSR: 160047 ;READ ONLY
706 002034 160046 DZVTDR: 160046 ;WRITE ONLY
707 002036 160047 HDZVTDR: 160047 ;WRITE ONLY
708
709                                     ;DEFAULT DZV VECTORS
710
711 002040 000300 DZVRIV: 300 ;REC INTR VECTOR
712 002042 000302 DZVRIS: 302 ;REC INTR STATUS
713 002044 000304 DZVTIV: 304 ;XMIT INTR VECTOR
714 002046 000306 DZVTIS: 306 ;XMIT INTR STATUS
715
716
    
```

717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740

002050
002050 000000
002052 000000
002054 000000
00 156 000000
00 160 000000
00 162 000000
00 164 000000
00 166 000000
00 170 000000
00 172 000000
00 174 000000
00 176 000000
002100 000000
002102 000000
002104 000000
002106 000000
002110 000000
002112 000000
002114 000000

; TIME TABLE FOR RELATIVE TIMING TESTS
;-----

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

K03

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 18
 DVDZBA.P11 28-JUL-77 07:37 PROGRAM INITIALIZATION AND START UP.

SEQ 0036

```

741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796

```

```

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

.START:
RESET
MOV #STACK, SP
MTPS #MASK
MOV #SPWRDN, @#24
CLR $PASS
CLRB $ERRFLG
MOV #DZV.MAP, ACTIVE
MOV #1, RUN
CLR $ERTTL
CLR $ERRPC
CLR $TSTNM
MOV #.START, $LPADR

;CLEAR THE WORLD. START NEW ENVIRONMENT
;SET UP STACK
;LOCK OUT INTERRUPTS
;SET UP POWER FAIL VECTOR
;CLEAR PASS COUNT
;CLEAR ERROR FLAG
;GET MAP POINTER.
;POINT POINTER TO FIRST DEVICE.
;CLEAR ERROR COUNT
;CLEAR LAST ERROR POINTER
;SET UP FOR TEST 1
;SET UP FOR POWER FAIL BEFORE
;TESTING STARTS

;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
MOV #SWREG, SWR
MOV #DISPREG, DISPLAY
TSTB INIFLG
BNE 10$
CMP @#42, #SENDAD
BEQ 1$
TYPE ,MTITLE
1$: DECB INIFLG
10$: TSTB $ENVH
BPL 15$
JSR PC, SETAPT
JMP 105$
15$: BIT #SW00, @SWR
BNE 20$
JMP 55$
20$: MOV #DZV.MAP, RO
CLRB $ORFLG
25$: CLR (RO)+
CMP RO, #DZV.END
BNE 25$
DECB INIFLG

;POINT TO SOFTWARE SWR
;POINT TO SOFTWARE DISPLAY REGISTER
;HAVE WE ALREADY BEEN HERE TODAY?
;IF SO, SKIP PRINTING THE TITLE
;IF RUNNING UNDER ACT
;DON'T PRINT TITLE
;PRINT THE DIAGNOSTIC'S TITLE
;SET THE ONCE ONLY FLAG
;DETERMINE WHETHER APT SIZING SHOULD BE DONE
;IF NOT, GO CHECK FOR AUTO-SIZING
;OTHERWISE, GO DO APT SIZING FROM ETABLE
;GO PRINT DZV STATUS TABLE
;RESELECT ?
;IF YES, GO SET UP THE INFORMATION
;IF NO, SKIP THE INTERROGATION
;POINT TO THE BEGINNING OF THE MAP TABLE
;MAKE SURE A MAP GETS PRINTED
;CLEAR A TABLE LOCATION
;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
;IF NOT, CLEAR THE NEXT LOCATION IN THE TABLE
;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!

;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
;TABLE AND SET UP THE DIAGNOSTIC.

;GET THE BASE ADDRESS OF THE DZV11'S

INSTR
91$
PARAM
160000
163770
DZVRO

```

L03

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 19
 DVDZBA.P11 28-JUL-77 07:37 PROGRAM INITIALIZATION AND START UP.

SEQ 0037

797	002340	007				.BYTE	7		;MASK OF INVALID BITS FOR THIS PARAMETER
798	002341	001				.BYTE	1		;NUMBER OF PARAMETERS TO STORE
799	002342	013737	001500	001174		MOV	DZCRO,\$BASE		;COPY BASE ADDRESS TO ETABLE
800									
801									;GET THE BASE VECTOR ADDRESS
802									
803	002350	104403				INSTR			;CALL THE STRING INPUT ROUTINE
804	002352	003062				92\$;POINTER TO MESSAGE TO BE PRINTED
805	002354	104405				PARAM			;CALL THE OCTAL TO ASCII CONVERT ROUTINE
806	002356	000300				300			;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
807	002360	000776				776			;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
808	002362	001502				DZVCO			;POINTER TO MAP LOCATION TO BE FILLED
809	002364	003				.BYTE	3		;MASK OF INVALID BITS FOR THIS PARAMETER
810	002365	001				.BYTE	1		;NUMBER OF PARAMETERS TO STORE
811	002366	013737	001502	001170		MOV	DZVCO,\$VECT1		;COPY VECTOR TO ETABLE
812									;GET THE MODE OF OPERATION (E,I,S)
813									
814	002374	104403				INSTR			;CALL THE STRING INPUT ROUTINE
815	002376	003311				96\$;POINTER TO THE MESSAGE TO BE PRINTED
816	002400	104406				SETFLG			;CALL THE MAINTENANCE FLAG SETUP ROUTINE
817	002402	001510				MANTO			;THIS IS THE FLAG BEING SETUP
818									
819									;GET THE NUMBER OF DZV11'S RUNNING
820									
821	002404	104403				INSTR			;CALL THE STRING INPUT ROUTINE
822	002406	003246				95\$;POINTER TO MESSAGE TO BE PRINTED
823	002410	104405				PARAM			;CALL THE OCTAL TO ASCII CONVERT ROUTINE
824	002412	000001				1			;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
825	002414	000020				16			;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
826	002416	001344				\$TMP1			;POINTER TO MAP LOCATION TO BE FILLED
827	002420	000				.BYTE	0		;MASK OF INVALID BITS FOR THIS PARAMETER
828	002421	001				.BYTE	1		;NUMBER OF PARAMETERS TO STORE
829									
830	002422	012737	000017	001504		MOV	#17,LINED		;SET UP DEFAULT LINES
831	002430	012737	017470	001506		MOV	#17470,PARO		;SET UP DEFAULT LPR PARAMETER
832									;RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
833	002436	032777	000010	176640		BIT	#SW03,\$SWR		;DO YOU WANT PARAMETERS?
834	00 444	001402				BEQ	30\$;IF NO, SKIP THE PARAMETER CALL
835	00 446	004737	002626			JSR	PC,65\$;GET PARAMETERS
836	00 452	012737	000001	001410	30\$:	MOV	#1,\$AVACTV		;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
837	00 0	113737	001344	001414		MOVB	\$TMP1,DZVNUM		;COPY THE NUMBER OF DEVICES
838	00 166	005337	001344		35\$:	DEC	\$TMP1		; \$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
839	002472	001404				BEQ	40\$;SELECTED DEVICES
840	002474	000261				SEC			;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
841	002476	006137	001410			ROL	\$AVACTV		;POINT TO THE NEXT DEVICE
842	002502	000771				BR	35\$;GO DO THIS PROCEDURE AGAIN
843	002504	013737	001410	001346	40\$:	MOV	\$AVACTV,\$TMP2		;# OF TIMES
844	002512	012700	001500			MOV	#DZCRO,R0		;SET A POINTER TO THE SPECIFIED INFORMATION
845	002516	012701	001512			MOV	#DZCR1,R1		;POINT R1 TO THE REST OF THE MAP TABLE
846	002522	012702	001204			MOV	#SDDW0,R2		;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
847	002526	000241				CLC			;INITIALIZE THE "C" BIT FOR A ROTATION
848	002530	006037	001346			ROF	\$TMP2		;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
849	002534	006237	001346		45\$:	ASK	\$TMP2		;ISOLATE A SELECTION FLAG IN THE "C" BIT
850	002540	103404				BCS	50\$;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
851	002542	012711	177777			MOV	#-1,(R1)		;TERMINATE THE LIST
852	002546	000137	003514			JMP	100\$;GO TO THE NEXT BLOCK

M03

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 20
 DVDZBA.P11 28-JUL-77 07:37

PROGRAM INITIALIZATION AND START UP.

SEQ 0038

```

853 002552 012011          50$:  MOV      (R0)+,(R1)      ;ADDRESS
854 002554 062721 000010  ADD      #10,(R1)+      ;POINT TO THE NEXT DZV11 ADDRESS VALUE
855 002560 012011          MOV      (R0)+,(R1)      ;VECTOR
856 002562 062721 000010  ADD      #10,(R1)+      ;POINT TO THE NEXT VECTOR VALUE
857 002566 012021          MOV      (R0)+,(R1)+     ;LINES
858 002570 012021          MOV      (R0)+,(R1)+     ;PARAMETERS
859 002572 012021          MOV      (R0)+,(R1)+     ;MAINTENANCE MODE
860 002574 000757          BR        45$
861 002576 032777 000010 176500 55$:  BIT      #SM03,#SMR      ;ASK PARAMETERS ?
862 002580 001002          BNE      60$            ;IF NO, GO DO AUTO SIZING
863 002586 000137 003514  JMP      100$           ;GO SET UP FOR AUTO SIZING
864 002592 004737 002626 60$:  JSR      PC,65$         ;GO ASK PARAMETERS
865 002616 105337 001422  DECB     INIFLG         ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
866 002622 000137 003540  JMP      105$           ;GO TO THE NEXT BLOCK
867
868 ;GET THE ACTIVE LINES PARAMETER
869
870 002626          65$:
871 002626 104403          INSTR     ;CALL THE STRING INPUT ROUTINE
872 002630 003123          93$      ;POINTER TO MESSAGE TO BE PRINTED
873 002632 104405          PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
874 002634 000001          1        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
875 002636 000017          17       ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
876 002640 001504          LINE0    ;POINTER TO MAP LOCATION TO BE FILLED
877 002642          360    ;MASK OF INVALID BITS FOR THIS PARAMETER
878 002643          001    ;NUMBER OF PARAMETERS TO STORE
879 002644 105037 001423  CLR0     HDRFLG        ;MAKE SURE THE CHANGES ARE PRINTED
880
881 ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
882 ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
883
884 002650 005737 001510          TST      MANTO         ;IS STAGGERED THE MODE OF OPERATION?
885 002654 100021          BPL      85$          ;IF NOT, SKIP THIS SEGMENT
886 002656 013703 001504          MOV      LINE0,R3     ;GET A SCRATCH COPY OF THE ACTIVE LINES
887 002662 006003          70$:  ROR      R3          ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
888 002664 103410          BCS      80$          ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
889 002666 001414          BEQ      85$          ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
890 002670 006203          ASR      R3          ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
891 002672 103373          BCC      70$         ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
892 002674 104402 001356          TYPE     'SQUES      ;THIS IS AN INCORRECT PARAMETER
893 002700 104402 010020          TYPE     'MBA0LN     ;LET THE USER KNOW ABOUT IT
894 002704 000750          BR        65$         ;GO GET THE CORRECT PARAMETER
895 002706 001772          80$:  BEQ      75$          ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
896 002710 006203          ASR      R3          ;GET THE NEXT FLAG
897 002712 103370          BCC      75$          ;IF IT ISN'T SET, THERE'S AN ERROR
898 002714 000241          CLC      ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
899 002716 000761          BR        70$         ;GO TEST THE NEXT PAIR OF FLAGS
900
901 ;GET THE LINE PARAMETER REGISTER ARGUMENT
902
903 002720          85$:
904 002720 104403          INSTR     ;CALL THE STRING INPUT ROUTINE
905 002722 003176          94$      ;POINTER TO MESSAGE TO BE PRINTED
906 002724 104405          PARAM    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
907 002726 000000          0        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
908 002730 000017          17       ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE

```

N03

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 21
 DVDZBA.P11 28-JUL-77 07:37

PROGRAM INITIALIZATION AND START UP.

SEQ 0039

909	002732	001506				PARO		: POINTER TO MAP LOCATION TO BE FILLED
910	002734	000				.BYTE	0	: MASK OF INVALID BITS FOR THIS PARAMETER
911	002735	001				.BYTE	1	: NUMBER OF PARAMETERS TO STORE
912	002736	012702	001504			MOV	#LINE0,R2	: POINT TO THE LINE SELECTION PARAMETER
913	002742	012703	001506			MOV	#PAR0,R3	: POINT TO THE CHOSEN PARAMETERS
914	00746	011304				MOV	(R3),R4	: USE BAUD RATE AS AN INDEX IN DELAY TABLE
915	00750	006304				ASL	R4	: ALIGN INDEX ON WORD BOUNDARY
916	00752	016437	017166	006230		MOV	DLYTBL(R4),DLYCNT	: SET THE DELAY COUNT FOR THIS BAUD RATE
917	00750	003313				SWAB	(R3)	: PLACE IN HIGH BYTE
918	00752	052713	010070			BIS	#10070,(R3)	: PLACE EXTRA PARAMETERS INTO LOC
919	00755	011262	000012		90\$:	MOV	(R2),12(R2)	: LOAD THE LINES
920	00772	011363	000012			MOV	(R3),12(R3)	: LOAD THE PARAMETERS
921	00776	062702	000012			ADD	#12,R2	: POINT TO THE NEXT SET
922	00772	062703	000012			ADD	#12,R3	: OF BOTH PARAMETERS
923	00776	003327	001734			CMP	R3,#PAR17	: HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
924	003012	001365				BNE	90\$: IF NOT, GO LOAD SOME MORE PARAMETERS
925	003014	000207				RTS	PC	: RETURN TO CALLING BLOCK
926	003016	0370	052123	041440	91\$:	.ASCIZ	<200>/1ST CSR ADDRESS (16000:163770): /	
(1)	003062	030600	052123	053040	92\$:	.ASCIZ	<200>/1ST VECTOR ADDRESS (300:770): /	
(1)	003123	200	044514	042516	93\$:	.ASCIZ	<200>/LINES ACTIVE BY BIT (IN OCTAL)(001:17): /	
(1)	003176	042200	043105	052501	94\$:	.ASCIZ	<200>/DEFAULT BAUD RATE (IN OCTAL)(00:17): /	
(1)	003246	021600	047440	020106	95\$:	.ASCIZ	<200>/# OF DZV11'S (IN OCTAL) (1:20): /	
(1)	003311	200	040515	047111	96\$:	.ASCII	<200>/MAINTENANCE MODE/	
(1)	003332	020200	042533	052130		.ASCII	<200>/ [EXTERNAL (H325) (E)]/	
(1)	003366	020200	044533	052116		.ASCII	<200>/ [INTERNAL (DZVCSR03=1)(I)]/	
(1)	003423	200	055440	052123		.ASCIZ	<200>/ [STAGGERED (H329) (S)]: /	
(1)	003462	042600	052116	051105	97\$:	.ASCIZ	<200>/ENTER DELAY PARAMETER: /	
(1)	003514	003514			100\$:	.EVEN		
927	003514	122737	000377	001422		CMPB	#377,INIFLG	: ONLY DO AUTO SIZE ON 1ST START
928	003522	001006				BNE	105\$	
929	003524	032777	000200	175552		BIT	#BIT7,2SWR	: BIT7=1??
930	003532	001002				BNE	105\$: BR IF NO AUTO SIZE
931	003534	004737	011464			JSR	PC,AUTO.SIZE	: GO DO THE AUTO SIZE
932	003540	105737	001423		105\$:	TSTB	HDRFLG	: HAS THE TABLE BEEN TYPED YET?
933	003544	001021				BNE	120\$: IF SO, DON'T TYPE IT AGAIN
934	003546	105337	001423			DECB	HDRFLG	: INDICATE THAT THE TABLE WILL BE TYPED
935	003552	104402	007772			TYPE	XHEAD	: TYPE MAP HEADER
936	003554	012700	001500			MOV	#DZV.MAP,R0	: SET POINTER
937	003556	010037	001344		110\$:	MOV	R0,\$TMP1	: POINT TO THE MAP LOCATION
938	003557	012037	001346			MOV	(R0)+,\$TMP2	: SET DATA
939	003576	022737	177777	001346		CMP	#-1,\$TMP2	: END OF LIST?
940	003600	001403				BEQ	120\$: BR IF YES
941	003602	104411			115\$:	CONVRT		: CALL THE OCTAL TO ASCII CONVERSION ROUTINE
942	003604	010062				XSTATQ		: CONVERT THE DATA AT THIS ADDRESS
943	003606	000765				BR	110\$: GO PRINT THE NEXT PARAMETER
944	003610	013737	001410	001406	120\$:	MOV	SAVACTV,DZVACTV	: COPY BIT MAP OF ACTIVE DEVICES
945	003616	113737	001414	001416		MOVB	DZVNUM,SAVNO	: COPY NO. OF DEVICES IN THE SYSTEM
946	003624	032777	000100	175452		BIT	#SW06,2SWR	: DESELECT SPECIFIC DEVICES??
947	003632	001431				BEQ	135\$: BR IF NO.
948	003634				121\$:			
949	003634	104403				INSTR		: CALL THE STRING INPUT ROUTINE
950	003636	007710				MNEW		: POINTER TO MESSAGE TO BE PRINTED
951	003640	104405				PARAM		: CALL THE OCTAL TO ASCII CONVERT ROUTINE
952	003642	000001				1		: LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
953	003644	177777				177777		: HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE

954	003646	001406			DZVACTV				: POINTER TO MAP LOCATION TO BE FILLED
955	003650	000			.BYTE	0			: MASK OF INVALID BITS FOR THIS PARAMETER
956	003651	001			.BYTE	1			: NUMBER OF PARAMETERS TO STORE
957	003652	023737	001406	001410	CMP	DZVACTV, SAVACTV			: IS VALUE VALID?
958	003660	101403			BLOS	127\$: IF YES BRANCH
959	003662	104402	007562		TYPE	, MERR3			: IF NOT TYPE ERROR
960	003666	000762			BR	121\$: THEN REASK QUESTION
961	003670	105037	001416	122\$:	CLRB	SAVNO			: INITIALIZE NO. OF ACTIVE DEVICES
962	003674	013737	001406	001344	MOV	DZVACTV, STMP1			: COPY BIT MAP OF ACTIVE DEVICES
963	003702	006237	001344	126\$:	ASR	STMP1			: ROTATE OUT AN ACTIVE BIT
964	003706	103002			BCC	127\$: IF NOT ACTIVE SKIP RECORDING IT
965	003710	105237	001416		INCB	SAVNO			: INCREMENT NO. OF ACTIVE DEVICES
966	003714	001372		127\$:	BNE	126\$: IF NOT DONE GO CONTINUE
967	003716	032777	000020	175360	135\$:	BIT	#SWC4, 2SWR		: CHECK TO SEE IF DELAY COUNT CHANGES
968	003724	001407			BEQ	140\$: IF NOT, GO CLEAR VECTOR AREA
969	003726	1044J3			INSTR				: CALL THE STRING INPUT ROUTINE
970	003730	003462			97\$: POINTER TO MESSAGE TO BE PRINTED
971	003732	104405			PARAM				: CALL THE OCTAL TO ASCII CONVERT ROUTINE
972	003734	000001			1				: LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
973	003736	177777			177777				: HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
974	003740	006230			DLYCNT				: POINTER TO MAP LOCATION TO BE FILLED
975	003742	000			.BYTE	0			: MASK OF INVALID BITS FOR THIS PARAMETER
976	003743	001			.BYTE	1			: NUMBER OF PARAMETERS TO STORE
977	003744	012700	000300	140\$:	MOV	#300, R0			: PREPARE TO CLEAR THE FLOATING
978	003750	012701	000302		MOV	#302, R1			: VECTOR AREA. 300-776
979	003754	010120		145\$:	MOV	R1 (R0)+			: START PUTTING "PC+2 - HALT"
980	003756	005021			CLR	(R1)+			: IN VECTOR AREA.
981	003760	022021			CMP	(R0)+, (R1)+			: POP POINTERS
982	003762	022700	001000		CMP	#1000, R0			: ALL DONE??
983	003766	001372			BNE	145\$: BR IF NO.
984									
985									
986									
987									
988	003770	012706	001120		.BEGIN:	MOV	#STACK, SP		: SET UP STACK
989	003774	106427	000200		MTPS	#MASK			: LOCK OUT INTERRUPTS
990	004000	005737	000042		TST	2#42			: IS PROGRAM UNDER MONITOR CONTROL
991	004004	001015			BNE	2\$: BR IF YES
992	004006	032777	000004	175270	BIT	#BIT2, 2SWR			: CHECK FOR LOCK ON TEST
993	004014	001406			BEQ	1\$: BR IF NO LOCK DESIRED.
994	004016	104402	007606		TYPE	, MLOCK			: TYPE LOCK SELECTED.
995	004022	012737	000240	004312	MOV	#NOP, TTST			: ADJUST SCOPE ROUTINE.
996	004030	000403			BR	2\$: CONTINUE ALONG.
997	004032	013737	004540	004312	1\$:	MOV	BRW, TTST		: PREPARE NORMAL SCOPE ROUTINE
998	004040	012737	010436	001252	2\$:	MOV	#CYCLE, \$LPADR		: START AT "CYCLE" FIND WHICH DEVICE TO TEST
999	004046	113737	001416	001415	MOV	SAVNO, SAVNUM			: COPY NO. OF ACTIVE DEVICES
1000	004054	104402	007477		TYPE	, MR			: TYPE "RUNNING"
1001	004060	000177	175166		JMP	\$SLPADR			: START TESTING

; TEST START AND RESTART

1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057

004064
004064 000004
004066 005037 001262
004072 105037 001247
004076 104402 007453
004102 104402 007635
004106 104412 004250
004112 104402 007643
004116 104412 004256
004122 005237 001126
004126 104402 007651
004132 104412 004264
004136 005337 001126
004142 104402 007662
004146 104412 004272
004152 005237 001130
004156 105337 001415
004162 001030
004164 113737 001416 001415
004172 005037 001354
004176 005237 001126
004202 042737 100000 001126
004210 005327
004212 000001
004214 003013
004216 012737
004220 000001
004222 004212
004224 013700 000042
004230 001405
004232 000005
004234 004710
004236 000240
004240 000240
004242 000240
004244
004244 000137
004246 010436
004250 000001
004252 006 002
004254 002010
004256 000001
004260 003 002

```

:END OF PASS
:TYPE NAME OF TEST
:UPDATE PASS COUNT
:CHECK FOR EXIT TO ACT-11
:RESTART TEST
.SBTTL END OF PASS ROUTINE

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

SEOP:
SCOPE
CLR $ERRPC ;CLEAR LAST ERROR PC
CLR $ERFLG ;CLEAR ERROR FLAG
TYPE ,MEPASS ;TYPE END PASS
TYPE ,MCSRX ;TYPE CSR
CNVRT ,XCSR ;SHOW IT
TYPE ,MVECX ;TYPE VECTOR
CNVRT ,XVEC ;SHOW IT
INC $PASS ;RAISE PASS COUNT
TYPE ,MPASSX ;TYPE PASSES
CNVRT ,XPASS ;SHOW IT
DEC $PASS ;RESTORE PASS COUNT
TYPE ,MERRX ;TYPE ERRORS
CNVRT ,XERR ;SHOW IT
INC $DEVCT ;INC DEVCT FOR APT
DECB $AVNUM ;ARE ALL DEVICES TESTED?
BNE $DOAGN ;BR IF NO.
MOV $AVNO,$AVNUM ;RESTORE THE COUNT
CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
INC $PASS ;INCREMENT THE PASS NUMBER
BIC #10000,$PASS ;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;LOOP?

SEOPCT: .WORD 1 ;YES
BGT $DOAGN ;RESTORE COUNTER
MOV (PC)+,(PC)+

SEDOCT: .WORD 1

$GET42: MOV #42,R0 ;GET MONITOR ADDRESS
BEQ $DOAGN ;BRANCH IF NO MONITOR
RESET ;CLEAR THE WORLD
SENDAD: JSR PC,(R0) ;GO TO MONITOR
NOP ;SAVE ROOM
NOP ;FOR
NOP ;ACT11

$DOAGN: ;RETURN
JMP #2(PC)+

$RTNAD: .WORD CYCLE

XCSR: 1
.BYTE 6,2
DZVCSR

XVEC: 1
.BYTE 3,2
```

```

1058 004262 002040
1059 004264 000001
1060 004266 006 002
1061 004270 001126
1062 004272 000001
1063 004274 006 002
1064 004276 001256
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081 004300
1082 004300 005037 001262
1083 004304 022716 012172
1084 004310 001413
1085 004312 000406
1086 004314 105777 174770
1087 004320 100067
1088 004322 017766 174764 177776
1089 004330 032777 040000 174746
1090 004336 001060
1091
1092 004340 000416
1093
1094 004342 013746 000004
1095 004346 012737 004366 000004
1096 004354 005737 177060
1097 004360 012637 000004
1098 004364 000436
1099 004366 022626
1100 004370 012637 000004
1101 004374 000441
1102 004376
1103 004376 105737 001247
1104 004402 001404
1105 004404 105037 001247
1106 004410 005037 001354
1107 004414 032777 004000 174662
1108 004422 001011
1109 004424 005737 001126
1110 004430 001406
1111 004432 005237 001250
1112 004436 023737 001354 001250
1113 004444 002015

```

```

DZVRIV
XPASS: 1
       .BYTE 6,2
       $PASS
XERR:  1
       .BYTE 6,2
       $ERTTL
;SCOPE LOOP AND ITERATION HANDLER
-----
.SBTTL SCOPE HANDLER ROUTINE
;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER($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

$SCOPE:
.SCOPE: CLR $ERRPC ;CLEAR LAST ERROR PC.
        CMP #TST1+2,(SP) ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
        BEQ $XTSTR ;IF SO, DON'T LOOP ON IT
TTST:   BR 1$ ;GOTO 1$ (IF LOCK SW02=1; THIS LOC =240)
        TSTB #STKS ;KEYBOARD DONE?
        BPL $OVER ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
        MOV #STKB,-2(SP) ;CLEAR DONE BIT
1$:     BIT #BIT14,$SWR ;LOOP ON PRESENT TEST?
        BNE $OVER ;YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$
        MOV @#ERRVEC,-(SP) ;IF RUNNING ON THE "XOR" TESTER CHANGE
        MOV #55,@#ERRVEC ;THIS INSTRUCTION TO A "NOP" (NOP=240)
        TST @#177060 ;SAVE THE CONTENTS OF THE ERROR VECTOR
        MOV (SP)+,@#ERRVEC ;SET FOR TIMEOUT
        BR $SVLAD ;TIME OUT ON XOR?
        CMP (SP)+,(SP)+ ;RESTORE THE ERROR VECTOR
        MOV (SP)+,@#ERRVEC ;GO TO THE NEXT TEST
        BR $OVER ;CLEAR THE STACK AFTER A TIME OUT
5$:     ;RESTORE THE ERROR VECTOR
        ;LOOP ON THE PRESENT TEST
6$: ;*****END OF CODE FOR THE XOR TESTER*****
2$:     TSTB $ERFLG ;HAS AN ERROR OCCURRED?
        BEQ 3$ ;BR IF NO
4$:     CLRB $ERFLG ;ZERO THE ERROR FLAG
3$:     CLR $TIMES ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
        BIT #BIT11,$SWR ;INHIBIT ITERATIONS?
        BNE 1$ ;BR IF YES
        TST $PASS ;IF FIRST PASS OF PROGRAM
        BEQ 1$ ;INHIBIT ITERATIONS
        INC $ICNT ;INCREMENT ITERATION COUNT
        CMP $TIMES,$ICNT ;CHECK THE NUMBER OF ITERATIONS MADE
        BGE $OVER ;BR IF MORE ITERATION REQUIRED

```

```

1114 004446 012737 000001 001250 1$: MOV #1, $ICNT ;; REINITIALIZE THE ITERATION COUNTER
1115 004454 013737 004542 001354 MOV $MXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
1116 004462 105237 001246 $SVLAD: INCB $STNM ;; COUNT TEST NUMBERS
1117 004466 113737 001246 001124 MOV $STNM, $TESTN ;; SET TEST NUMBER IN APT MAILBOX
1118 004474 011637 001252 MOV (SP), $LPAOR ;; SAVE SCOPE LOOP ADDRESS
1119 004500 013777 001246 174600 $OVER: MOV $STNM, $DISPLAY ;; DISPLAY TEST NUMBER
1120 004506 013716 001252 MOV $LPAOR, (SP) ;; FUDGE RETURN ADDRESS
1121 004512 004737 006772 JSR PC, SERV.G ;; FIND OUT IF TG WAS TYPED
1122 004516 105037 001424 CLR $MNTFLG ;; CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
1123 004522 005737 001372 TST $MODE ;; HAS THE MODE BEEN CHANGED?
1124 004526 001003 BNE $S ;; IF NOT INTERNAL, GO DO A TEST
1125 004530 112737 000010 001424 MOV $MNTFLG, $MNTFLG ;; IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
1126 004536 000002 $S: RTI ;; GO DO THE TEST
1127 004540 000406 BRW: 406
1128 004542 000005 $MXCNT: 5 ;; MAX. NUMBER OF ITERATIONS
1129
1130 ;; CHECK FOR FREEZE ON CLURRENT DATA
1131 -----
1132
1133 004544 032777 001000 174532 .SCOPI: BIT #SW09, $SWR ;; IS SW09=1 (SET)?
1134 004552 001405 BEQ $S ;; BR IF NOT SET.
1135 004554 005737 001364 TST $LOCK ;; IS THERE A TIGHT LOOP SPECIFIED?
1136 004560 001402 BEQ $S ;; IF NO, RETURN
1137 004562 013716 001364 MOV $LOCK, (SP) ;; IF YES, GOTO THE ADDRESS IN $LOCK.
1138 004566 000002 $S: RTI ;; GO BACK.
1139
1140 004570 032777 010000 174506 .TYPE: BIT #SW12, $SWR ;; INHIBIT ALL PRINTOUT??
1141 004576 001403 BEQ $TYPE ;; IF NOT, GO TYPE
1142 004600 062716 000002 ADD #2, (SP) ;; SKIP OVER MESSAGE POINTER
1143 004604 000002 RTI ;; RETURN TO WHERE PROCEDURE WAS INVOKED
1144
1145 .SBTTL TYPE ROUTINE
1146
1147 ;; *****
1148 ;; ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1149 ;; THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1150 ;; NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1151 ;; NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1152 ;; NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1153 ;;
1154 ;; CALL:
1155 ;; 1) USING A TRAP INSTRUCTION
1156 ;; TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1157 ;; OR
1158 ;; TYPE
1159 ;; MESADR
1160 ;;
1161 004606 105737 001323 $TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
1162 004612 100002 $S BPL $S ;; BR IF YES
1163 004614 000000 HALT ;; HALT HERE IF NO TERMINAL
1164 004616 000430 BR $S ;; LEAVE
1165 004620 010046 $S: MOV $RO, -(SP) ;; SAVE RO
1166 004622 017600 000002 MOV $R2, $RO ;; GET ADDRESS OF ASCIZ STRING
1167 004626 122737 000001 001140 CMPB #APTENV, $ENV ;; RUNNING IN APT MODE
1168 004634 001011 BNE $S2 ;; NO, GO CHECK FOR APT CONSOLE
1169 004636 132737 000100 001141 BITB #APTPOOL, $ENVM ;; SPOOL MESSAGE TO APT

```

```

1170 004644 001405      BEQ      62$      ; NO GO CHECK FOR CONSOLE
1171 004646 010037 004656  MOV      RD,61$  ; SETUP MESSAGE ADDRESS FOR APT
1172 004652 004737 005076  JSR      PC,$ATY3 ; SPOOL MESSAGE TO APT
1173 004656 000000      .WORD    0       ; MESSAGE ADDRESS
1174 004660 132737 000040 001141 61$:     BITB     #APTCSUP,SEVM ; APT CONSOLE SUPPRESSED
1175 004666 001003      BNE      60$      ; YES, SKIP TYPE OUT
1176 004670 112046      2$:     MOVB     (PC)+,-(SP) ; PUSH CHARACTER TO BE TYPED ONTO STACK
1177 004672 001005      BNE      4$       ; BR IF IT ISN'T THE TERMINATOR
1178 004674 005726      TST      (SP)+    ; IF TERMINATOR POP IT OFF THE STACK
1179 004676 012600      60$:    MOV      (SP)+,RD ; RESTORE RD
1180 004700 062716 000002 3$:     ADD      #2,(SP) ; ADJUST RETURN PC
1181 004704 000002      RTI                      ; RETURN
1182 004706 122716 000011 4$:     CMPB     #HT,(SP) ; BRANCH IF <HT>
1183 004712 001430      BEQ      8$       ;
1184 004714 122716 000200 5$:     CMPB     #CRLF,(SP) ; BRANCH IF NOT <CRLF>
1185 004720 001006      BNE      5$       ;
1186 004722 005726      TST      (SP)+    ; POP <CR><LF> EQUIV
1187 004724 104402      TYPE                    ; TYPE A CR AND LF
1188 004726 001357      $CRLF
1189 004730 105037 005064 6$:     CLRB     $CHARCNT ; CLEAR CHARACTER COUNT
1190 004734 000755      BR      2$        ; GET NEXT CHARACTER
1191 004736 004737 005020 5$:     JSR      PC,$TYPEC ; GO TYPE THIS CHARACTER
1192 004742 123726 001322 6$:     CMPB     $FILLC,(SP)+ ; IS IT TIME FOR FILLER CHARS.?
1193 004746 001350      BNE      2$        ; IF NO GO GET NEXT CHAR.
1194 004750 013746 001320 7$:     MOV      $NULL,-(SP) ; GET # OF FILLER CHARS. NEEDED
1195                                     ; AND THE NULL CHAR.
1196 004754 105366 000001 7$:     DECB     1(SP) ; DOES A NULL NEED TO BE TYPED?
1197 004760 002770      BLT      6$        ; BR IF NO--GO POP THE NULL OFF OF STACK
1198 004762 004737 005020      JSR      PC,$TYPEC ; GO TYPE A NULL
1199 004766 105337 005064      DECB     $CHARCNT ; DO NOT COUNT AS A COUNT
1200 004772 000770      BR      7$        ; LOOP
1201
1202 ; HORIZONTAL TAB PROCESSOR
1203
1204 004774 112716 000040 8$:     MOVB     #' ,(SP) ; REPLACE TAB WITH SPACE
1205 005000 004737 005020 9$:     JSR      PC,$TYPEC ; TYPE A SPACE
1206 005004 132737 000007 005064 9$:     BITB     #7,$CHARCNT ; BRANCH IF NOT AT
1207 005012 001372      BNE      9$        ; TAB STOP
1208 005014 005726      TST      (SP)+    ; POP SPACE OFF STACK
1209 005016 000724      BR      2$        ; GET NEXT CHARACTER
1210 005020 105777 174270 $TYPEC: TSTB     $STPS ; WAIT UNTIL PRINTER IS READY
1211 005024 100375      BPL      $TYPEC
1212 005026 116677 000002 174262 1$:     MOVB     2(SP),$STPB ; LOAD CHAR TO BE TYPED INTO DATA REG.
1213 005034 122766 000015 000002 1$:     CMPB     #CR,2(SP) ; IS CHARACTER A CARRIAGE RETURN?
1214 005042 001003      BNE      1$        ; BRANCH IF NO
1215 005044 105037 005064 1$:     CLRB     $CHARCNT ; YES--CLEAR CHARACTER COUNT
1216 005050 000406      BR      $TYPEX ; EXIT
1217 005052 122766 000012 000002 1$:     CMPB     #LF,2(SP) ; IS CHARACTER A LINE FEED?
1218 005060 001402      BEQ      $TYPEX ; BRANCH IF YES
1219 005062 105227      INCB     (PC)+    ; COUNT THE CHARACTER
1220 005064 000000 $CHARCNT: .WORD    0 ; CHARACTER COUNT STORAGE
1221 005066 000207 $TYPEX:  RTS      PC
1222
1223 .SBTTL  APT COMMUNICATIONS ROUTINE
1224
1225 ; *****

```

```

1226 005070 112737 000001 005334 SATY1: MOVB #1, SFFLG ;; TO REPORT FATAL ERROR
1227 005076 112737 000001 005332 SATY3: MOVB #1, SMFLG ;; TO TYPE A MESSAGE
1228 005104 000403 BR SATYC
1229 005106 112737 000001 005334 SATY4: MOVB #1, SFFLG ;; TO ONLY REPORT FATAL ERROR
1230 005114 SATYC:
1231 005114 010046 MOV RO, -(SP) ;; PUSH RO ON STACK
1232 005116 010146 MOV R1, -(SP) ;; PUSH R1 ON STACK
1233 005120 105737 005332 TSTB SMFLG ;; SHOULD TYPE A MESSAGE?
1234 005124 001450 BEQ SS ;; IF NOT: BR
1235 005126 122737 000001 001140 CMPB #APTENV, SENV ;; OPERATING UNDER APT?
1236 005134 001031 BNE SS ;; IF NOT: BR
1237 005136 132737 000100 001141 BITB #APTPOOL, SENVM ;; SHOULD SPOOL MESSAGES?
1238 005144 001425 BEQ SS ;; IF NOT: BR
1239 005146 017600 000004 MOV #24(SP), RO ;; GET MESSAGE ADDR.
1240 005152 062766 000002 000004 ADD #2, 4(SP) ;; BUMP RETURN ADDR.
1241 005160 005737 001120 1S: TST SMSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
1242 005164 001375 BNE 1S ;; IF NOT: WAIT
1243 005166 010037 001134 MOV RO, MSGAD ;; PUT ADDR IN MAILBOX
1244 005172 105720 2S: TSTB (RO)+ ;; FIND END OF MESSAGE
1245 005174 001376 BNE 2S
1246 005176 163700 001134 SUB MSGAD, RO ;; SUB START OF MESSAGE
1247 005202 006200 ASR RO ;; GET MESSAGE LNTH IN WORDS
1248 005204 010037 001136 MOV RO, SMSGLT ;; PUT LENGTH IN MAILBOX
1249 005210 012737 000004 001120 MOV #4, SMSGTYPE ;; TELL APT TO TAKE MSG.
1250 005216 000413 BR SS
1251 005220 017637 000004 005244 3S: MOV #24(SP), 4S ;; PUT MSG ADDR IN JSR LINKAGE
1252 005226 062766 000002 000004 ADD #2, 4(SP) ;; BUMP RETURN ADDRESS
1253 005234 013746 177776 MOV 177776, -(SP) ;; PUSH 177776 ON STACK
1254 005240 004737 004606 JSR PC, $TYPE ;; CALL TYPE MACRO
1255 005244 000000 4S: .WORD 0
1256 005246 5S:
1257 005246 105737 005334 10S: TSTB SFFLG ;; SHOULD REPORT FATAL ERROR?
1258 005252 001416 BEQ 12S ;; IF NOT: BR
1259 005254 005737 001140 TST SENV ;; RUNNING UNDER APT?
1260 005260 001413 BEQ 12S ;; IF NOT: BR
1261 005262 005737 001120 11S: TST SMSGTYPE ;; FINISHED LAST MESSAGE?
1262 005266 001375 BNE 11S ;; IF NOT: WAIT
1263 005270 017637 000004 001122 MOV #24(SP), $FATAL ;; GET ERROR #
1264 005276 062766 000002 000004 ADD #2, 4(SP) ;; BUMP RETURN ADDR.
1265 005304 005237 001120 INC SMSGTYPE ;; TELL APT TO TAKE ERROR
1266 005310 105037 005334 12S: CLRB SFFLG ;; CLEAR FATAL FLAG
1267 005314 105037 005333 CLRB SLFLG ;; CLEAR LOG FLAG
1268 005320 105037 005332 CLRB SMFLG ;; CLEAR MESSAGE FLAG
1269 005324 012601 MOV (SP)+, R1 ;; POP STACK INTO R1
1270 005326 012600 MOV (SP)+, RO ;; POP STACK INTO RO
1271 005330 000207 RTS PC ;; RETURN
1272 005332 000 SMFLG: .BYTE 0 ;; MESSG. FLAG
1273 005333 000 SLFLG: .BYTE 0 ;; LOG FLAG
1274 005334 000 SFFLG: .BYTE 0 ;; FATAL FLAG
1275 005336 .EVEN
1276 000200 APTSIZE=200
1277 000001 APTENV=001
1278 000100 APTPOOL=100
1279 000040 APTCSUP=040
1280
1281 ;STRING INPUT ROUTINE

```

```

1282 ;-----
1283 ;
1284 005336 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON STACK
1285 005340 010446 MOV R4,-(SP) ;SAVE R4 ON STACK
1286 005342 017637 000004 005360 MOV 24(SP),.MSG ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
1287 005350 062766 000002 000004 ADD #2,4(SP) ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
1288 0 536 104402 .INST1: TYPE ;PRINT THE MESSAGE
1289 0 5360 000000 .MSG: 0 ;MESSAGE IS POINTED TO FROM HERE
1290 0 5372 012704 010270 MOV #INBUF,R4 ;POINT R4 TO THE INPUT BUFFER
1291 0 5376 012703 000007 MOV #7,R3 ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
1292 0 5377 105777 173712 1$: TSTB @STKS ;HAS A CHARACTER BEEN RECEIVED?
1293 0 5376 100375 BPL 1$ ;IF NO, KEEP WAITING FOR IT
1294 0 5370 117714 173706 MOVB @STKB,(R4) ;IF YES, SAVE IT IN THE INPUT BUFFER
1295 0 5404 142714 000200 BICB #200,(R4) ;KEEP ONLY THE 7-BIT ASCII INFORMATION
1296 0 5410 122427 000015 CMPB (R4),#15 ;IS THIS CHARACTER A LINE FEED?
1297 0 5414 001417 BEQ INSTR2 ;IF SO, TERMINATE THE INPUT SEQUENCE
1298 0 5416 105777 173672 2$: TSTB @STPS ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
1299 0 5422 100375 BPL 2$ ;IF WE CAN'T, WAIT UNTIL WE CAN
1300 0 5424 017777 173662 173664 MOV @STKB,@STPB ;ECHO THE CHARACTER BACK
1301 0 5432 005303 DEC R3 ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
1302 0 5434 001356 BNE 1$ ;IF WE DON'T HAVE 7, GO GET SOME MORE
1303 0 5436 012604 MOV (SP)+,R4 ;IF WE HAVE 7, RESTORE R4
1304 005440 012603 MOV (SP)+,R3 ;RESTORE R3
1305 05442 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON THE STACK
1306 0 5444 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1307 005446 104402 001356 TYPE ,QUES ;PRINT A QUESTION MARK... WHAT'S GOING ON?
1308 005452 000741 BR .INST1 ;GO PRINT THE MESSAGE AGAIN
1309 005454 012604 INSTR2: MOV (SP)+,R4 ;RESTORE R4
1310 005456 012603 MOV (SP)+,R3 ;RESTORE R3
1311 005460 000002 RTI ;RETURN TO THE MAIN PROCEDURE
1312 ;
1313 ;CONVERT ASCII STRING TO OCTAL
1314 ;-----
1315 ;
1316 005462 010546 .PARAM: MOV R5,-(SP) ;SAVE R5 ON THE STACK
1317 005464 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
1318 005466 016605 000004 MOV 4(SP),R5 ;GET THE SETUP INFORMATION POINTER
1319 005472 012537 005652 MOV (R5)+,LOLIM ;SET THE LOW LIMIT FOR THE INPUT
1320 005474 012537 005654 MOV (R5)+,HILIM ;SET THE HIGH LIMIT FOR THE INPUT
1321 005502 012537 005656 MOV (R5)+,DEVAOR ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
1322 005506 112537 005660 MOVB (R5)+,LOBITS ;GET THE MASK OF THE INCORRECT BITS
1323 005512 112537 005661 MOVB (R5)+,ADRCNT ;GET THE COUNT OF ITEMS TO BE STORED
1324 005516 010566 000004 MOV R5,4(SP) ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
1325 005522 005005 PARAM1: CLR R5 ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
1326 005524 012704 010270 MOV #INBUF,R4 ;POINT TO THE INPUT BUFFER
1327 005530 122714 000015 CMPB #15,(R4) ;IS THIS CHARACTER A CARRIAGE RETURN?
1328 005534 001420 BEQ PARERR ;IF SO, PRINT THE MESSAGE AGAIN
1329 005536 121427 000060 1$: CMPB (R4),#60 ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
1330 005542 002415 BLT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1331 005544 121427 000067 CMPB (R4),#67 ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
1332 005550 003012 BGT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
1333 005552 142714 000060 BICB #60,(R4) ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
1334 005556 152405 BISB (R4)+,R5 ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
1335 005560 122714 000015 CMPB #15,(R4) ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
1336 005564 001406 BEQ LIMITS ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
1337 005566 006305 ASL R5 ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
    
```

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 29
 DVDZBA.P11 28-JUL-77 07:37 APT COMMUNICATIONS ROUTINE

SEQ 0047

```

1338 005570 006305          ASL    R5          ; CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
1339 005572 006305          ASL    R5          ; MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
1340                                ; NEXT THREE BITS
1341 005574 000760          BR     1$          ; GO GET THE NEXT CHARACTER
1342 005576 104404          PARERR: INSTER    ; THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
1343 005600 000750          BR     PARAM1     ; TRY GETTING THE PARAMETERS AGAIN
1344
1345                                ; TEST TO SEE IF NUMBER IS WITHIN LIMITS
1346                                ;-----
1347
1348 005602 020537 005654    LIMITS: CMP    R5,HILIM ; DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
1349 005606 101373          BHI   PARERR      ; IF YES, GO PRINT THE MESSAGE AGAIN
1350 005610 020537 005652    CMP    R5,LOLIM   ; IS THE RESULT LOWER THAN ALLOWED?
1351 005614 103770          BLO   PARERR      ; IF YES, GO PRINT THE MESSAGE AGAIN
1352 005616 133705 005660    BITB  LOBITS,R5  ; ARE ANY INCORRECT BITS SET IN THE RESULT?
1353 005622 001365          BNE   PARERR      ; IF SO, GO PRINT THE MESSAGE AGAIN
1354
1355                                ; STORE NUMBER AT SPECIFIED ADDRESS
1356
1357 005624 013704 005656    1$:   MOV    DEVAOR,R4 ; POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
1358 005630 010524          MOV    R5,(R4)+  ; STORE THE RESULT
1359 005632 062705 000002    ADD    #2,R5     ; CALCULATE THE NEXT DATUM
1360 005636 105337 005661    DECB  ADRCNT     ; REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
1361 005642 001372          BNE   1$         ; IF NOT, GO STORE THE NEXT DATUM
1362 005644 012604          MOV    (SP)+,R4  ; RESTORE R4
1363 005646 012605          MOV    (SP)+,R5  ; RESTORE R5
1364 005650 000002          RTI           ; RETURN TO THE MAIN PROGRAM
1365
1366 005652 000000          LOLIM: 0         ; LOWEST ACCEPTABLE VALUE
1367 005654 000000          HILIM: 0         ; HIGHEST ACCEPTABLE
1368 005656 000000          DEVAOR: 0        ; LOCATION WHERE RESULT WILL BE STORED
1369 005660 000          LOBITS: .BYTE 0  ; INCORRECT BITS MASK
1370 005661 000          ADRCNT: .BYTE 0 ; COUNT OF ITEMS TO BE STORED
1371
1372                                ; SAVE PC OF TEST THAT FAILED AND R0-R5
1373                                ;-----
1374
1375 005662 016637 000004 001404 .SAV05: MOV    4(SP),SAVPC ; SAVE R7 (PC)
1376
1377                                ; SAVE R0-R5
1378
1379 005670 010537 001340    SV05: MOV    R5,$REG5 ; SAVE R5
1380 005674 010437 001336    MOV    R4,$REG4 ; SAVE R4
1381 005700 010337 001334    MOV    R3,$REG3 ; SAVE R3
1382 005704 010237 001332    MOV    R2,$REG2 ; SAVE R2
1383 005710 010137 001330    MOV    R1,$REG1 ; SAVE R1
1384 005714 010037 001326    MOV    R0,$REG0 ; SAVE R0
1385 005720 000002          RTI           ; LEAVE.
1386
1387                                ; RESTORE R0-R5
1388
1389 005722 013700 001326    .RES05: MOV    $REG0,R0 ; RESTORE R0
1390 005726 013701 001330    MOV    $REG1,R1 ; RESTORE R1
1391 005732 013702 001332    MOV    $REG2,R2 ; RESTORE R2
1392 005736 013703 001334    MOV    $REG3,R3 ; RESTORE R3
1393 005742 013704 001336    MOV    $REG4,R4 ; RESTORE R4

```



```

1394 005746 013705 001340      MOV      $REGS,R5      ;RESTORE R5
1395 005752 000002      RTI                    ;LEAVE
1396
1397      ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1398      -----
1399
1400 005754 104402 001357      .CONVR: TYPE          $CRLF          ;PRINT A CARRIAGE RETURN
1401 005760 010046      .CNVRT: MOV           R0,-(SP)      ;SAVE R0
1402 005762 010146      MOV           R1,-(SP)      ;SAVE R1
1403 005764 010346      MOV           R3,-(SP)      ;SAVE R3
1404 005766 010446      MOV           R4,-(SP)      ;SAVE R4
1405 005770 010546      MOV           R5,-(SP)      ;SAVE R5
1406 005772 017601 000012      MOV           @12(SP),R1     ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
1407 005776 062766 000002 000012      ADD           #2,12(SP)     ;POINT TO WHERE MAIN PROGRAM WILL RESUME
1408 006004 012137 006130      MOV           (R1)+,WRDCNT  ;GET NUMBER OF WORDS TO BE PRINTED
1409 006010 112105      1$: MOV        (R1)+,R5     ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
1410 006012 112100      MOV        (R1)+,R0     ;GET THE NUMBER OF SPACES TO PRINT
1411 006014 013104      MOV           @1(R1)+,R4    ;COPY THE WORD TO BE CONVERTED
1412 006016 110537 006132      MOV        R5,CHRCNT   ;COPY THE CHARACTER COUNT
1413 006022 010403      3$: MOV        R4,R3    ;COPY THE PREVIOUS WORD AGAIN
1414 006024 042703 177770      BIC         #1<7>,R3     ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
1415 006030 062703 000060      ADD         #060,R3     ;MAKE AN ASCII CHARACTER OUT OF THEM
1416 006034 110346      MOV        R3,-(SP)    ;SAVE THAT CHARACTER
1417 006036 006004      ROR        R4          ;MOVE THE NEXT THREE BITS INTO PLACE
1418 006040 006204      ASR        R4          ;MOVE THEM AGAIN
1419 006042 006204      ASR        R4          ;AND FINALLY A THIRD TIME
1420 006044 005305      DEC        R5          ;REDUCE CHARACTER COUNT. ARE ALL CHARACTERS
1421                          ;BUILT?
1422 006046 001365      BNE        3$         ;IF NO, GO BUILD THE NEXT ONE.
1423 006050 012703 010374      MOV        #MDATA,R3   ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
1424 006054 112623      4$: MOV        (SP)+,(R3)+ ;STORE THE CHARACTER, STARTING WITH THE MOST
1425 006056 105337 006132      DECB      CHRCNT     ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
1426 006062 001374      BNE        4$         ;IF NO, GO TRANSFER ANOTHER
1427 006064 105700      TSTB      R0         ;ARE ANY SPACES TO BE PRINTED?
1428 006066 001404      BEQ        6$         ;IF NO, DON'T SET UP ANY
1429 006070 112723 000040      5$: MOV        #040,(R3)+ ;ADD A SPACE TO THE OUTPUT BUFFER
1430 006074 105300      DECB      R0         ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
1431 006076 001374      BNE        5$         ;IF YES, GO ADD ANOTHER SPACE
1432 006100 105013      6$: CLRB      (R3)    ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
1433 006102 104402 010374      TYPE      ,MDATA     ;PRINT THE STRING WE JUST BUILT
1434 006106 005337 006130      DEC        WRDCNT    ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
1435 006112 001336      BNE        1$         ;IF YES, GO CONVERT THEM
1436 006114 012605      MOV        (SP)+,R5   ;RESTORE R5
1437 006116 012604      MOV        (SP)+,R4   ;RESTORE R4
1438 006120 012603      MOV        (SP)+,R3   ;RESTORE R3
1439 006122 012601      MOV        (SP)+,R1   ;RESTORE R1
1440 006124 012600      MOV        (SP)+,R0   ;RESTORE R0
1441 006126 000002      RTI                    ;RETURN TO THE MAIN PROGRAM
1442 006130 000000      WRDCNT: 0
1443 006132 000      CHRCNT: .BYTE
1444 006133 000      SPACNT: .BYTE 0
1445
1446 006134 000000      BINWRD: 0
1447
1448
1449      ;TRAP DISPATCH SERVICE
    
```

```

1450                                     ; ARGUMENT OF TRAP IS EXTRACTED
1451                                     ; AND USED AS OFFSET TO OBTAIN POINTER
1452                                     ; TO SELECTED SUBROUTINE
1453
1454 006136 010046 .TRPSR: MOV      RO, -(SP)      ; SAVE RO. USE RO TO FIND TRAP ROUTINE
1455 006140 016600 000002 MOV      2(SP), RO      ; GET TRAP ADDRESS
1456 006144 005740 TST      -(RO)         ; GET TRAP
1457 006146 111000 MOVNB   (RO), RO       ; GET RIGHT BYTE OF TRAP (TRAP OFFSET)
1458 006150 006300 ASL     RO             ; POSITION OFFSET FOR TABLE INDEXING
1459 006152 016000 001742 MOV      .TRPTAB(RO), RO ; PLACE INDEXED ADDRESS OF TABLE IN RO
1460 006156 000200 RTS      RO             ; TRANSFER TO THAT ADDRESS AND RESTORE OLD RO
1461
1462                                     ; DEVICE CLEAR ROUTINE
1463                                     ; ISSUE A DEVICE CLEAR
1464 -----
1465 006160 .DEVICE.CLR:
1466 006160 052777 000020 173622 BIS      #DCLR, @DZVCSR ; SET DCLR
1467 006166 032777 000020 173614 IS: BIT      #DCLR, @DZVCSR ; DID IT CLEAR?
1468 006174 001374 BNE     IS             ; BR IF NO
1469 006176 000002 RTI     ; EXIT ROUTINE
1470
1471                                     ; ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
1472 -----
1473 006200 .DCLASM: DEVICE.CLR ; ISSUE A DEVICE CLEAR
1474 006202 153777 001424 173600 BISB    MNTFLG, @DZVCSR ; LOAD THE MAINTENANCE BIT IF IT IS I MODE
1475 006210 000002 RTI     ; RETURN TO CALLING ROUTINE
1476
1477 .DELAY:
1478 006212 010046 MOV      RO, -(SP)      ; SAVE RO
1479 006214 013700 006230 MOV      DLYCNT, RO     ; SET COUNT
1480 006220 005300 IS: DEC     RO             ; DELAY
1481 006222 001376 BNE     IS             ;
1482 006224 012600 MOV      (SP)+, RO     ; RESTORE RO
1483 006226 000002 RTI     ; LEAVE ROUTINE
1484 006230 000001 DLYCNT: .WORD 1 ; PATCHABLE LOC FOR MORE TIME
1485
1486                                     ; ADVANCE TO NEXT TEST HANDLER
1487 -----
1488
1489 006232 013716 001362 .ADVANCE: MOV     NEXT, (SP) ; CRUNCH STACK WITH ADDRESS OF SCOPE CALL
1490 006236 005037 001364 CLR     LOCK           ; RESET TIGHT LOOP ADDRESS
1491 006242 000002 RTI     ; CHECK TO SEE IF OLD TEST GETS REPEATED
1492
1493                                     ; ROUTINE TO SHIFT LINE POINTER
1494                                     ; AND SWITCH TESTS IF NECESSARY
1495 -----
1496 006244 106302 .SHIFT: ASLB    R2             ; POINT TO THE NEXT LINE
1497 006246 032702 000020 BIT      #BIT4, R2      ; HAVE WE PASSED ALL LINE POINTERS?
1498 006252 001402 BEQ     IS             ; IF NOT, RETURN TO THE TEST
1499 006254 022626 POP2SP ; REMOVE THE TRAP CALL FROM THE STACK
1500 006256 104400 ADVANCE ; GO TO THE NEXT TEST
1501 006260 000002 IS: RTI     ; RETURN TO THE PRESENT TEST
1502

```

```

1503                                     ;LINE PARAMETER REGISTER SETUP ROUTINE
1504
1505 006262 010146 .LPRSET:MOV R1,-(SP) ;SAVE CONTENTS OF R1
1506 006264 010246 MOV R2,-(SP) ;SAVE CONTENTS OF R2
1507 006266 013701 001370 MOV PAR,R1 ;MOVE DEFAULT PARAM. INTO R1
1508 006272 012702 000001 MOV #1,R2 ;INIT. FOR LINE 1
1509 006276 010177 173516 1$: MOV R1,00ZVLP ;LOAD PARAM. REGISTER
1510 006302 005201 INC R1 ;SET R1 FOR NEXT LINE
1511 006304 106302 ASLB R2 ;SET R2 FOR NEXT LINE
1512 006306 032702 000020 BIT #BIT4,R2 ;ALL LINES DONE?
1513 006312 001771 BEQ 1$ ;IF NO LOAD NEXT LINE
1514 006314 012602 MOV (SP)+,R2 ;RELOAD R2
1515 006316 012601 MOV (SP)+,R1 ;RELOAD R1
1516 006320 000002 RTI ;RETURN
1517
1518                                     ;ROUTINE TO ZERO DATA BUFFER
1519
1520 006322 010046 .BUFSET:MOV R0,-(SP) ;SAVE CONTENTS OF R0
1521 006324 012700 001426 MOV #TDO,R0 ;SET R0 TO TOP OF BUFFER
1522 006330 005020 1$: CLR (R0)+ ;CLEAR BUFFER LOCATION
1523 006332 022700 001446 CMP #STOP,R0 ;IS BUFFER ALL CLEARED
1524 006336 001374 BNE 1$ ;IF NOT CLEAR NEXT LOCATION
1525 006340 012600 MOV (SP)+,R0 ;RELOAD R0
1526 006342 000002 RTI ;RETURN
1527
1528                                     ;ERROR HANDLER
1529 -----
1530
1531 006344 004737 006772 $ERROR: JSR PC,SERV.G ;FIND OUT IF <IG> WAS HIT
1532 006350 032777 010000 172726 BIT #SW12,2SWR ;BELL ON ERROR?
1533 006356 001406 BEQ XBX ;BR IF NO BELL
1534 006360 105777 172730 TSTB 2STPS ;TTY READY.
1535 006364 100003 BPL XBX ;DON'T WAIT IF TTY NOT READY.
1536 006366 112777 000207 172722 MOVB #207,2STPB ;PUSH A BELL AT THE TTY.
1537 006374 032777 020000 172702 XBX: BIT #SW13,2SWR ;DELETE ERROR PRINT OUT?
1538 006402 001113 BNE HALTS ;BR IF NO PRINT OUT WANTED.
1539 006404 021637 001262 CMP (SP),$ERRPC ;WAS THIS ERROR FOUND LAST TIME?
1540 006410 001404 BEQ 1$ ;BR IF YES
1541 006412 011637 001262 MOV (SP),$ERRPC ;RECORD BEING HERE
1542 006416 105037 001247 CLRB $ERFLG ;PREPARE HEADER
1543 006422 104407 1$: SAVOS ;SAVE ALL PROC REGISTERS
1544 006424 011605 MOV (SP),R5 ;GET THE PC OF ERROR
1545 006426 162705 000002 SUB #2,R5 ;GET ADDRESS OF TRAP CALL
1546 006432 011504 MOV (R5),R4 ;GET ERROR INSTRUCTION
1547 006434 110437 001260 MOVB R4,$ITEMB ;COPY TEST NUMBER FOR APT HANDLING
1548 006440 006304 ASL R4 ;MULT BY TWO
1549 006442 061504 ADD (R5),R4 ;DOUBLE IT
1550 006444 006304 ASL R4 ;MULT AGAIN
1551 006446 042704 177001 BIC #177001,R4 ;CLEAR JUNK
1552 006452 062704 015316 ADD #.ERRTAB,R4 ;GET POINTER
1553 006456 012437 006602 MOV (R4)+,ERRMSG ;GET ERROR MESSAGE
1554 006462 012437 006614 MOV (R4)+,DATAHD ;GET DATA HEADRER
1555 006466 011437 006626 MOV (R4),DATABP ;GET DATA TABLE
1556 006472 105737 001247 TSTB $ERFLG ;TYPE HEADER
1557 006476 001403 BEQ TYPMSG ;BR IF YES
1558 006500 005737 006626 TST DATABP ;DOES DATA TABLE EXIST?
    
```

```

1559 006504 001044
1560 006506 104402 001357 TYPMSG: TYPE ,SCRLF ;BR IF YES.
1561 006512 104402 001357 TYPE ,SCRLF ;TYPE A CARRIAGE RETURN
1562 0 516 005737 001364 TST LOCK ;AND TYPE ANOTHER
1563 0 22 001402 BEQ IS
1564 006524 104402 007705 TYPE ,MASTEK
1565 0 30 104402 007673 15: TYPE ,MTSTN
1566 0 34 104412 006764 CNVRT ,XTSTN ;SHOW IT
1567 006540 104402 007765 TYPE ,MERRPC ;TYPE PC.
1568 0 344 104412 007756 CNVRT ,ERTAB0 ;SHOW IT
1569 0 0 104402 007635 TYPE ,MCSRX
1570 0 34 104412 004250 CNVRT ,XCSR
1571 006560 104402 001357 TYPE ,SCRLF ;GIVE A CR/LF
1572 0 54 112737 177777 001247 MOVB #-1, SERFLG ;NO MORE HEADER UNLESS NO DATA TABLE.
1573 006572 005737 006602 TST ERRMSG ;IS THERE AN ERROR MESSAGE?
1574 006576 001402 BEQ WTBS.FM ;BR IF NO.
1575 0 600 104402 TYPE ;TYPE
1576 0 602 000000 ERRMSG: 0 ;ERROR MESSAGE
1577 0 604 WTBS.FM:
1578 0 604 005737 006614 TST DATA0 ;DATA HEADER?
1579 0 610 001402 BEQ TYPDAT ;BR IF NO
1580 0 612 104402 TYPE ;TYPE
1581 0 614 000000 DATA0: 0 ;DATA HEADER
1582 0 616 005737 006626 TYPDAT: TST DATABP ;DATA TABLE?
1583 0 622 001402 BEQ RESREG ;BR IF NO.
1584 0 624 104411 CNVRT ;SHOW
1585 006626 000000 DATABP: 0 ;DATA TABLE
1586 006630 104410 RESREG: RESOS ;RESTORE PROC REGISTERS
1587 0 632 122737 000701 001140 HALTS: CMPB #APTENV, SEMV ;IS APT RUNNING?
1588 0 640 001007 BNE IS ;SKIP APT CALL IF NOT
1589 0 642 113737 001260 006654 MOVB $ITEMB, 55 ;COPY ERROR NUMBER
1590 0 650 004737 005106 JSR PC, SATY4 ;CALL APT SERVICE
1591 0 54 000000 55: .WORD 0 ;ERROR NUMBER STUCK HERE
1592 0 656 000777 105: BR 105 ;LOCK UP HERE
1593 0 660 022737 004234 000042 155: CMP #SENDAD, @#42 ;CHECK TO SEE IF IN ACT-11 MODE
1594 0 56 001403 BEQ 205 ;IF SO, HANDLE ACCORDINGLY
1595 0 6670 005777 172410 TST @SWR ;HALT ON ERROR?
1596 006674 100004 BPL EXITER ;BR IF NO HALT ON ERROR
1597 0 576 016677 000002 172402 205: MOV 2(SP), @DISPLAY ;SHOW ERROR PC IN DATA DISPLAY
1598 0 704 000000 HALT ;HALT
1599 006706 005237 001256 EXITER: INC $ERTTL ;UPDATE ERROR COUNT
1600 006712 004737 006772 JSR PC, SERV.G ;FIND OUT IF IG WAS TYPED
1601 0 6716 032777 000400 172360 BIT #SW08, @SWR ;GOTO TOP OF TEST?
1602 006724 001007 BNE IS ;BR IF YES
1603 006726 032777 002000 172350 BIT #SW10, @SWR ;GOTO NEXT TEST?
1604 006734 001407 BEQ 25 ;BR IF NO
1605 006736 013737 001362 001252 15: MOV NEXT, $LPA0R ;SET FOR NEXT TEST
1606 0 6744 012706 001120 MOV #STACK, SP ;RESET SP
1607 0 6750 000177 172276 JMP @LPA0R ;GOTO SPECIFIED TEST
1608 0 6754 000002 25: RTI ;RETURN
1609 0 6756 000001 ERTAB0: 1
1610 006760 006 002 .BYTE 6,2
1611 006762 001404 SAVPC
1612 006764 000001 XTSTN: 1
1613 006766 002 002 .BYTE 2,2
1614 006770 001246 $TSTNM

```

```

1615 006772 017746 172314      SERV.G: MOV    @STKB,-(SP)      ; OTHERWISE, GET THE LAST CHARACTER TYPED
1616 006776 042716 000200      BIC    #BIT7,(SP)           ; STRIP PARITY(EIGHTH) BIT
1617 007002 122726 000007      CMPB   #7,(SP)+            ; IS IT ^G?
1618 007006 001076 000000      BNE    6$                  ; IF NOT, IGNORE INPUT
1619 007010 002777 004000 172272      BIT    #4000,@STKS         ; RX BUSY?
1620 007016 001365 172260 007226      BNE    SERV.G              ; BR IF YES
1621 007020 017737 172260 007226      MOV    @SWR,90$           ; SAVE (SWR).
1622 007026 104402 007206 1$:      TYPE   '89$               ; TYPE HEADER FOR OLD SWITCH REGISTER
1623 007032 104412 007220      CNVRT  '88$               ; TYPE THE NUMBER ITSELF
1624 007036 104402 007230      TYPE   '91$               ; AFTER HAVING CONVERTED IT TO ASCII
1625 007042 100337 007234      CLRB   92$                ; CLEAR SWR CHANGE FLAG
1626 007046 000377 172232      CLR    @SWR               ; CLEAR THE SOFTWARE SWITCH REGISTER
1627 007052 100777 172232      3$:      TSTB   @STKS              ; WAIT FOR DONE.
1628 007056 100375 172232      BPL    3$                  ; CONTINUE WAITING FOR IT
1629 007060 017746 172226      MOV    @STKB,-(SP)         ; PUT THE CHARACTER ON THE STACK
1630 007064 042716 000200      BIC    #BIT7,(SP)         ; STRIP PARITY BIT
1631 007070 122726 000015      CMPB   #15,(SP)+          ; IS IT THE CARRIAGE RETURN CHAR?
1632 007074 001433 172212      BEQ    4$                  ; IF SO, GO PRINT CRLF
1633 007076 105777 172212      2$:      TSTB   @STPS              ; IS THE OUTPUT BUFFER AVAILABLE
1634 007102 100375 172212      BPL    2$                  ; IF NOT, WAIT FOR IT TO BE READY
1635 007104 105237 007234      INCB   92$                ; INDICATE THAT THE SWR WAS CHANGED
1636 007110 014677 172202      MOV    -(SP),@STPB        ; PLACE THE CHARACTER THERE(ECHO BACK)
1637 007114 000241 172162      LLC    @SWR                ; GET READY TO ROTATE
1638 007116 006177 172162      ROL    @SWR                ; MOVE THE EXISTING BITS OVER
1639 007122 006177 172156      ROL    @SWR                ; TO MAKE ROOM FOR THE INCOMING
1640 007126 006177 172152      ROL    @SWR                ; THREE BITS FROM THIS CHARACTER
1641 007132 103735 172152      BCS    1$                  ; ERROR
1642 007134 022627 000060      CMP    (SP)+,#60          ; IS IT LOWER THAN 0?
1643 007140 002732 172152      BLT    1$                  ; IF SO, GO ASK AGAIN
1644 007142 026627 177776 000067      CMP    -2(SP),#67         ; IS IT HIGHER THAN 7?
1645 007150 003326 177770      BGT    1$                  ; IF SO, GO ASK AGAIN
1646 007152 042746 172122      BIC    #1C<7>,-(SP)       ; ISOLATE INFORMATION BITS
1647 007156 052677 172122      BIS    (SP)+,@SWR         ; ADD THEM TO THE SWITCH REGISTER
1648 007162 000733 172122      BR     3$                  ; GO CHECK FOR THE NEXT CHARACTER
1649 007164 105737 007234      4$:      TSTB   92$                ; HAS THE SWR BEEN CHANGED?
1650 007170 001003 172104      BNE    5$                  ; IF YES GO TYPE CRLF
1651 007172 013777 007226 172104      MOV    90$,@SWR           ; IF NOT RESTORE SWR
1652 007200 104402 001357      5$:      TYPE   $CRLF              ; TYPE A CARRIAGE RETURN AND LINE FEED
1653 007204 000207 172104      6$:      RTS    PC                  ; RETURN TO CALLING PROCEDURE
1654
1655 007206 020200 051450 051127 89$:      .ASCIZ <200>? (SWR)=/?
1656 007214 036451 000057
1657
1658 007220 000001      .EVEN
1659 007222 006 000      88$:      1
1660 007224 007226      .BYTE 6,0
1661 007226 000000      90$:      .WORD 0
1662 007230 036457 000057      91$:      .ASCIZ ?/=/?
1663 007234 000      92$:      .BYTE 0
1664
1665      .EVEN
1666      .SBTTL POWER DOWN AND UP ROUTINES
1667
1668      ; *****
1669      ; POWER DOWN ROUTINE
1669 007236 012737 007402 000024 $PWRDN: MOV    #SILLUP,@PWRVEC ; SET FOR FAST UP
1670 007244 012737 000340 000026      MOV    #340,@PWRVEC+2 ; ;PRIO:7

```

```

1671 007252 010046      MOV      R0,-(SP)      ;; PUSH R0 ON STACK
1672 007254 010146      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
1673 007256 010246      MOV      R2,-(SP)      ;; PUSH R2 ON STACK
1674 007260 010346      MOV      R3,-(SP)      ;; PUSH R3 ON STACK
1675 007262 010446      MOV      R4,-(SP)      ;; PUSH R4 ON STACK
1676 007264 010546      MOV      R5,-(SP)      ;; PUSH R5 ON STACK
1677 007266 017746      MOV      @SWR,-(SP)    ;; PUSH @SWR ON STACK
1678 007272 010637 007406      MOV      SP,$SAVR6    ;; SAVE SP
1679 007276 012737 007310 000024      MOV      @SPWRUP,@@PWVVEC ;; SET UP VECTOR
1680 007304 000000      HALT
1681 007306 000776      BR      -2            ;; HANG UP
1682
1683      ;; *****
1684      ;; POWER UP ROUTINE
1685 007310 012737 007402 000024      $PWUP: MOV      @SILLUP,@@PWVVEC ;; SET FOR FAST DOWN
1686 007316 013706 007406      MOV      $SAVR6,SP    ;; GET SP
1687 007322 005037 007406      CLR      $$VR6        ;; WAIT LOOP FOR THE TTY
1688 007326 005237 007406      IS:     INC      $$SAVR6 ;; WAIT FOR THE INC
1689 007332 001375      BNE     IS           ;; OF WORD
1690 007334 012677 171744      MOV      (SP)+,@SWR   ;; POP STACK INTO @SWR
1691 007340 012605      MOV      (SP)+,R5     ;; POP STACK INTO R5
1692 007342 012604      MOV      (SP)+,R4     ;; POP STACK INTO R4
1693 007344 012603      MOV      (SP)+,R3     ;; POP STACK INTO R3
1694 007346 012602      MOV      (SP)+,R2     ;; POP STACK INTO R2
1695 007350 012601      MOV      (SP)+,R1     ;; POP STACK INTO R1
1696 007352 012600      MOV      (SP)+,R0     ;; POP STACK INTO R0
1697 007354 012737 007236 000024      MOV      @SPWRON,@@PWVVEC ;; SET UP THE POWER DOWN VECTOR
1698 007362 012737 000340 000026      MOV      @340,@@PWVVEC+2 ;; @340: ?
1699 007370 104402      TYPE
1700 007372 007410      $PWMSG: .WORD  MPFAIL ;; REPORT THE POWER FAILURE
1701 007374 012716      MOV      (PC)+,(SP)  ;; POWER FAIL MESSAGE POINTER
1702 007376 010776      $PWRAD: .WORD  RESTART ;; RESTART AT RESTART
1703 007400 000002      RTI
1704 007402 000000      $SILLUP: HALT      ;; RESTART ADDRESS
1705 007404 000776      BR      -2            ;; THE POWER UP SEQUENCE WAS STARTED
1706 007406 000000      $SAVR6: 0            ;; BEFORE THE POWER DOWN WAS COMPLETE
1707 007410 050200 051127 043040      MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 007453 200 047105 020104      MEPASS: .ASCIZ  <200>/END PASS DVDZB-A /
(2) 007477 200 052522 047116      MR:     .ASCIZ  <200>/RUNNING /
(2) 007513 200 051120 043517      MERR2: .ASCIZ  <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 007562 044600 051516 043125      MERR3: .ASCIZ  <200>/INSUFFICIENT DATA!/
(2) 007606 046200 041517 020113      MLOCK: .ASCIZ  <200>/LOCK ON SELECTED TEST/
(2) 007635 103 051123 020072      MCSRX: .ASCIZ  /CSR: /
(2) 007643 126 041505 020072      MVECX: .ASCIZ  /VEC: /
(2) 007651 120 051501 042523      MPASSX: .ASCIZ  /PASSES: /
(2) 007662 051105 047522 051522      MERRX: .ASCIZ  /ERRORS: /
(2) 007673 124 051505 020124      MTSTN: .ASCIZ  /TEST NO: /
(2) 007705 052 000040      MASTEK: .ASCIZ  /* /
(2) 007710 052200 050131 020105      MNEW:  .ASCIZ  <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE: /
(2) 007765 120 035103 000040      MERRPC: .ASCIZ  /PC: /
(2) 007772 046600 050101 047440      XHEAD: .ASCIZ  <200>/MAP OF DZV11 STATUS/<200>
(2) 010020 044600 046114 043505      MBADLN: .ASCIZ  <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)
(2) 010062 000002      .EVEN
(2) 010064 006 003      XSTATQ: 2
1708 010064 006 003      .BYTE  6,3
1709 010066 001344      $TMP1

```

```

1710 010070 006 002 .BYTE 6,2
1711 010072 001346 STMP2
1712 .EVEN
1713 ; THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
1714 -----
1715 ; E=EXTERNAL LOOP BACK
1716 ; I=INTERNAL LOOP BACK
1717 ; S=STAGGERED LOOP BACK
1718 010074 017605 000000 .SETFLG: MOV 2(SP),R5 ; PICK UP ADDRESS OF TAG
1719 010100 042737 000040 010270 BIC #40,INBUF ; STRIP LOWER CASE
1720 010106 122737 000105 010270 CMPB #'E,INBUF ; IS IT EXTERNAL LOOP BACK ?
1721 010114 001005 BNE 4$ ; NO
1722 010116 013715 010206 MOV 1$,(R5) ; YES STORE INFO
1723 010122 105037 001424 CLRB MNTFLG ; SET MAINT BIT =0
1724 010126 000422 BR 7$ ; GET OUT
1725 010130 122737 000111 010270 4$: CMPB #'I,INBUF ; IS IT INTERNAL LOOP BACK ?
1726 010136 001006 BNE 5$ ; NO
1727 010140 013715 010210 MOV 2$,(R5) ; YES STORE INFO
1728 010144 112737 000010 001424 MOVB #MAINT,MNTFLG ; SET UP THE MAINTENANCE FLAG LOADER
1729 010152 000410 BR 7$ ; GET OUT
1730 010154 122737 000123 010270 5$: CMPB #'S,INBUF ; IS IT STAGGERED LOOP BACK ?
1731 010162 001007 BNE 6$ ; WHAT ?
1732 010164 013715 010212 MOV 3$,(R5) ; YES STORE INFO
1733 010170 105037 001424 CLRB MNTFLG ; ZERO BITS
1734 010174 062716 000002 7$: ADD #2,(SP) ; POP AROUND
1735 010200 000002 RTI
1736 010202 104404 6$: INSTER ; RETRY
1737 010204 000733 BR .SETFLG ; DITTO
1738 010206 000200 1$: .WORD 200 ; EXTERNAL = E
1739 010210 000000 2$: .WORD 0 ; INTERNAL = I
1740 010212 100000 3$: .WORD 100000 ; STAGGERED = S
1741
    
```

```

1742                                     ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
1743                                     ;BUFFER TO THE CHARACTERS "E" AND "C".
1744                                     ;IF THE CHARACTER IS "E" CLEAR THE FLAG
1745                                     ;IF THE CHARACTER IS "C" SET THE FLAG
1746
1747 010214 017605 000000 .PAWCH:MOV      2(SP),R5
1748 010220 142737 000040 010270 BICB      #40,INBUF      ;SET FOR LOWER CASE INPUT
1749 010226 122737 000105 010270 CMPB      #'E,INBUF      ;IS IT "E" ?
1750 010234 001002 BNE      1$
1751 010236 105015 CLR      (R5)          ;000
1752 010240 000406 BR        2$
1753 010242 122737 000103 010270 1$: CMPB      #'C,INBUF      ;IS IT "C" ?
1754 010250 001005 BNE      3$
1755 010252 112715 177777 MOV      #1,(R5)        ;3177
1756 010256 062716 000002 2$: ADD      #2,(SP)
1757 010262 000002 RTI
1758 010264 104404 3$: INSTER
1759 010266 000752 BR        .PAWCH      ;RETRY
1760
1761                                     ;BUFFERS FOR INPUT-OUTPUT
1762
1763 010270 000000 INBUF: 0
1764 .=. +40
1765 010332 000000 TEMP: 0
1766 .=. +40
1767 010374 000000 MDATA: 0
1768 .=. +40
1769
    
```


E05

```

1770
1771
1772
1773
1774
1775
1776
1777
1778
1779 010436 005737 001406      CYCLE: TST      DZVACTV      ;ARE ANY DZV11'S TO BE TESTED?
1780 010442 001004                BNE      1$          ;BR IF OK.
1781 010444 104402 007513      TYPE     ,MERR2     ;NO DZV11'S SELECTED!!
1782 010450 000000                HALT                                ;STOP THE SHOW.
1783 010452 000776                BR      -2          ;DISQUALIFY CONT. SW.
1784 010454 013737 004542 001354 1$: MOV     SMAXCNT,STIMES ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
1785 010462 033737 001412 001406 BIT     RUN,DZVACTV ;IS THIS ONE "ACTIVE"
1786 010470 001017                BNE     2$          ;BR IF GOOD ONE FOUND.
1787 010472 006137 001412      ROL     RUN        ;UPDATE POINTER
1788 010476 005537 001412      ADC     RUN        ;CATCH CARRY FROM RUN
1789 010502 062737 000012 001420 ADD     #12,ACTIVE  ;UPDATE ADDRESS POINTER.
1790 010510 022737 001740 001420 CMP     #DZV.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
1791 010516 001356                BNE     1$          ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
1792 010520 012737 001500 001420 MOV     #DZV.MAP,ACTIVE ;RESET ADDRESS POINTER.
1793 010526 000752                BR      1$          ;KEEP LOOKING FOR ACTIVE DZV11
1794 010530 006137 001412      ROL     RUN        ;UPDATE POINTER.
1795 010534 005537 001412      ADC     RUN        ;CATCH CARRY.
1796 010540 013700 001420      MOV     ACTIVE,RO  ;GET ADDRESS POINTER.
1797 010544 062737 000012 001420 ADD     #12,ACTIVE  ;UPDATE.
1798 010552 022737 001740 001420 CMP     #DZV.END,ACTIVE
1799
1800 010560 001003                BNE     3$          ;ALL DONE?
1801 010562 012737 001500 001420 MOV     #DZV.MAP,ACTIVE ;BR IF NO.
1802 010570 012037 001174 3$: MOV     (RO)+,$BASE  ;RESTORE POINTER.
1803 010574 012037 002040      MOV     (RO)+,DZVRIV ;LOAD SYSTEM CTRL. REG
1804 010600 012037 001366      MOV     (RO)+,LINE  ;LOAD VECTOR
1805 010604 012037 001370      MOV     (RO)+,PAR   ;SET UP DZV LINES ACTIVE
1806 010610 012037 001372      MOV     (RO)+,MODE  ;SET UP PARAMETERIZATION
1807 010614 105037 001424      CLRB   MNTFLG ;RESET MAINT. FLAG IF
1808 010620 005737 001372      TST    MODE        ;RUNNING TESTS
1809 010624 001003                BNE     9$          ;IN
1810 010626 112737 000010 001424 MOVB   #MAINT,MNTFLG ;INTERNAL MAINT. MODE
1811 010634 004737 011002 9$: JSR    PC,DZVLEV  ;SET UP
1812 010640 005737 000042      TST    #42         ;ARE WE UNDER MONITOR CONTROL?
1813 010644 001051                BNE     7$          ;IF YES, SKIP THIS SETUP
1814 010646 032777 000002 170430 BIT     #SW01,#SWR  ;IF SW01=1, GET STARTING TEST #
1815 010654 001445                BEQ    7$          ;BR IF NO TEST IS TO BE INPUTTED
1816 010656 104402 001357 4$: TYPE   ,SCRLF
1817 010662 104403                INSTR
1818 010664 007673                MTSTN
1819 010666 104405                PARAM
1820 010670 000001                1
1821 010672 001000                1000
1822 010674 001246                $STSTM
1823 010676 000                .BYTE 0
1824 010677 001                .BYTE 1
1825 010700 012700 012170      MOV     #TST1,RO
  
```

```

1826 010704 022710 000004      5$:    CMP      #4,(R0)
1827 010710 001020                BNE      6$
1828 010712 022760 012737 000002      CMP      #12737,2(R0)
1829 010720 001014                BNE      6$
1830 010722 023760 001246 000004      CMP      $1STNM,4(R0)
1831 010730 001010                BNE      6$
1832 010732 010037 001252      MOV      R0,$LPAOR
1833 010736 062737 000002 001252      ADD      #2,$LPAOR
1834 010744 104402 001357      TYPE    $CALF
1835 010750 000412                BR       8$
1836 010752 005720                6$:    TST      (R0)+
1837 010754 020027 014322      CMP      R0,$TLAST+10
1838 010760 001351                BNE      5$
1839 010762 104402 001356      TYPE    $QUES
1840 010766 000733                BR       4$
1841 010770 012737 012170 001252  7$:    MOV      #TST1,$LPAOR ;PREPARE TEST ADDRESS
1842 010776                8$:
1843 010776 000177 170250      RESTART:JMP  #2$LPAOR ;GO START TESTING.***WARNING!****
1844                                     ;THIS JUMP IS USED BY POWER UP ROUTINE!!!!
1845
1846                                     ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
1847 011002 013700 002040      DZVLEV: MOV     DZVRIV,R0 ;PLACE THE BASE VECTOR ADDRESS IN R0
1848 011006 062700 000002      ADD      #2,R0 ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
1849 011012 010037 002042      MOV      R0,DZVRIS ;STORE IT HERE
1850 011016 062700 000002      ADD      #2,R0 ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
1851 011022 010037 002044      MOV      R0,DZVTIV ;STORE IT HERE
1852 011026 062700 000002      ADD      #2,R0 ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
1853 011032 010037 002046      MOV      R0,DZVTIS ;STORE IT HERE
1854
1855                                     ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
1856                                     ;OF THE DEVICE
1857 011036 013700 001174      MOV      $BASE,R0 ;COPY THE ADDRESS BEING LOADED
1858 011042 010037 002010      MOV      R0,DZVCSR ;XXX0
1859 011046 005200                INC      R0
1860 011050 010037 002012      MOV      R0,HDZVCSR ;XXX1
1861 011054 005200                INC      R0
1862 011056 010037 002014      MOV      R0,DZVRBUF ;XXX2
1863 011062 010037 002020      MOV      R0,DZVLPR ;XXX2
1864 011066 005200                INC      R0
1865 011070 010037 002016      MOV      R0,HDZVRBUF ;XXX3
1866 011074 010037 002022      MOV      R0,HDZVLPR ;XXX3
1867 011100 005200                INC      R0
1868 011102 010037 002024      MOV      R0,DZVTCR ;XXX4
1869 011106 005200                INC      R0
1870 011110 010037 002026      MOV      R0,HDZVTCR ;XXX5
1871 011114 005200                INC      R0
1872 011116 010037 002030      MOV      R0,DZVMSR ;XXX6
1873 011122 010037 002034      MOV      R0,DZVTDR ;XXX6
1874 011126 005200                INC      R0
1875 011130 010037 002032      MOV      R0,HDZVMSR ;XXX7
1876 011134 010037 002036      MOV      R0,HDZVTDR ;XXX7
1877 011140 000207                RTS      PC
    
```

```

1878
1879 011142 011605
1880 011144 012537 011326
1881 011150 012537 011330
1882 011154 012537 011332
1883 011160 112537 011334
1884 011164 112537 011335
1885 011170 010516
1886 011172 005005
1887 011174 012704 010270
1888 011200 122714 000015
1889 011204 001424
1890 011206 121427 000060
1891 011212 002421
1892 011214 121427 000071
1893 011220 003016
1894 011222 142714 000060
1895 011226 005002
1896 011230 152402
1897 011232 060205
1898 011234 122714 000015
1899 011240 001410
1900 011242 006305
1901 011244 010502
1902 011246 006305
1903 011250 006305
1904 011252 060205
1905 011254 000754
1906 011256 104404
1907 011260 000744
1908
1909
1910
1911 011262 020537 011330
1912 011266 101373
1913 011270 020537 011326
1914 011274 103770
1915 011276 133705 011334
1916 011302 001365
1917
1918
1919
1920 011304 013704 011332
1921 011310 010524
1922 011312 062705 000002
1923 011316 105337 011335
1924 011322 001372
1925 011324 000002
1926 011326 000000
1927 011330 000000
1928 011332 000000
1929 011334 000
1930 011335 000

      : CONVERT DECIMAL ASCII STRING TO OCTAL
.PARMD: MOV      (SP), R5
        MOV      (R5)+, 6$
        MOV      (R5)+, 7$
        MOV      (R5)+, 8$
        MOVB     (R5)+, 9$
        MOVB     (R5)+, 10$
        MOV      R5, (SP)
2$:    CLR      R5
        MOV      #INBUF, R4
        CMPB     #15, (R4)
1$:    BEQ      3$
        CMPB     (R4), #'0
        BLT      3$
        CMPB     (R4), #'9
        BGT      3$
        BICB     #'0, (R4)
        CLR      R2
        BISB     (R4)+, R2
        ROR      R2, R5
        CMPB     #15, (R4)
        BEQ      4$
        RSL      R5, R2      ; X2
        MOV      R5, R2      ; SAVE X2
        RSL      R5, R2      ; X4
        RSL      R5, R2      ; X8
        ROR      R2, R5      ; TIMES 10
        BR       1$
3$:    INSTER   2$
        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)+
        ROR      #2, R5
        DECB     10$
        BNE      5$
        RTI
6$:    0
7$:    0
8$:    0
9$:    .BYTE 0
10$:   .BYTE 0

```

```

1931                                     ;#ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
1932                                     ;*IF BIT7 IN THE ENVIRONMENT MODE (SEMVM) BYTE IS SET
1933                                     ;*THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
1934
1935 011336 012700 001500 SETAPT: MOV #DZV.MAP,R0 ;POINT TO THE DEVICE MAP TABLE
1936 011342 013701 001174 MOV $,R1 ;BUILD DEVICE ADDRESSES IN R1
1937 011346 013702 001170 MOV $VECT1,R2 ;BUILD DEVICE VECTORS IN R2
1938 011352 042702 177007 BIC #10<770>,R2 ;STRIP AWAY OTHER INFORMATION
1939 01136 012704 001204 MOV #SCDMD,R4 ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
1940 011362 013705 001176 MOV #DZVM,R5 ;GET THE MAP OF ACTIVE DEVICES
1941 011366 105037 001414 CLR B DZVNUM ;INITIALIZE THE NO. OF ACTIVE DEVICES
1942 011372 005037 001410 CLR SAVACTV ;CLEAR THE ACTIVE BIT MAP
1943 011376 006005 1$: ROR R5 ;GET A DEVICE SELECTION BIT
1944 011400 103407 BCS 3$ ;IF IT IS SELECTED, GO SET UP A MAP
1945 011402 001422 BEQ 5$ ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
1946 011404 005724 TST (R4)+ ;POINT TO NEXT DEVICE DESCRIPTOR
1947 011406 062701 000010 2$: ADD #10,R1 ;SET UP THE NEXT ADDRESS
1948 011412 062702 000010 ADD #10,R2 ;SET UP THE NEXT VECTOR GROUP
1949 011416 000767 BR 1$ ;GO SEE IF MORE DEVICES REMAIN
1950 011420 006137 001410 3$: ROL SAVACTV ;SET BIT IN ACTIVE DEVICE MAP
1951 011424 105237 001414 INCB DZVNUM ;INCREMENT NO. OF ACTIVE DEVICES
1952 011430 010120 MOV R1,(R0)+ ;LOAD DEVICE ADDRESS
1953 011432 010220 MOV R2,(R0)+ ;LOAD THE VECTOR ADDRESS
1954 011434 013720 001200 MOV #SCDM1,(R0)+ ;GET THE NUMBER OF LINES IN OPERATION
1955 011440 012420 MOV (R4)+,(R0)+ ;LOAD DEVICE PARAMETERS
1956 011442 013720 001202 MOV #SCDM2,(R0)+ ;LOAD DEFAULT TESTING MODE
1957 011446 000757 BR 2$ ;GO BUILD THE NEXT ADDRESS
1958 011450 012710 177777 5$: MOV #-1,(R0) ;TERMINATE THE DEVICE MAP
1959 011454 012737 001142 001304 MOV #SSWREG,SWR ;SET TO SOFTWARE APT SWITCH REGISTER
1960 011462 000207 RTS PC ;RETURN TO PRINT STATUS TABLE

```

```

1961
1962
1963                                     ;#ROUTINE USED TO "AUTO SIZE" THE DZV11
1964                                     ;*CSR AND VECTOR.
1965                                     ;*NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1966                                     ;* ADDRESS RANGE (160000:163770)
1967                                     ;* AND THE VECTOR MAY BE ANY WHERE IN THE
1968                                     ;* FLOATING VECTOR RANGE (300:770)
1969                                     ;*
1970

```

```

1971 011464 AUTO.SIZE: RESET ;INSURE A BUS INIT.
1972 011464 000005 DECB INIFLG ;SHOW THAT I WAS HERE
1973 011466 105337 001422 CSRMAP: MOV #DZV.MAP,R2 ;LOAD MAP POINTER.
1974 011472 012702 001500 MOV #SDMD,R3 ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
1975 011476 012703 001204 1$: CLR (R2)+ ;ZERO ENTIRE MAP
1976 011502 005022 CMP #DZV.END,R2 ;ALL DONE?
1977 011504 022702 001740 BNE 1$ ;BR IF NO
1978 011510 001374 CLR B DZVNUM ;SET OCTAL NUMBER OF DZV11'S TO 0
1979 011512 105037 001414 MOV #DZV.MAP,R2
1980 011516 012702 001500 MOV #160000,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
1981 011522 012701 160000 MOV #65,2#4 ;SET FOR NON-EXISTENT DEVICE TIME OUT
1982 011526 012737 011772 000004 2$: BIS #BITS,(R1) ;TRY TO SET MASTER SCAN ENABLE
1983 011534 052711 000040 BIS #17,4(R1) ;TRY TO TRANSMIT ON ANY LINE
1984 011540 052761 000017 000004 CLR R0 ;USE R0 AS A COUNTER
1985 011546 005000 7$: TST (R1) ;HAS TRANSMITTER READY COME UP?
1986 011550 005711

```

1987	011552	100403				BMI	8\$: IF SO, GO GET A FINAL CHECK
1988	011554	005300				DEC	R0		: REDUCE COUNT. TIME UP?
1989	011556	001374				BNE	7\$: IF NOT, KEEP WAITING
1990	011560	000437				BR	3\$: ASSUME IT'S NOT A DZV11
1991	011562	032761	000017	000004	8\$:	BIT	#17,4(R1)		: ARE ANY TCR BITS STILL SET? THEY SHOULD BE
1992	011570	001433				BEQ	3\$: IF IT'S NOT, ASSUME IT'S NOT A DZV11
1993	011572	032711	000040			BIT	#BITS,(R1)		: IS MASTER SCAN ENABLE STILL SET?
1994	011576	001430				BEQ	3\$: IF NOT, ASSUME IT'S NOT A DZV11
1995	011600	C 2711	000020			BIS	#20,(R1)		: SET DEVICE CLEAR
1996	011604	000240				NOP			
1997	011606	032711	000040			BIT	#40,(R1)		: DID SCANNER CLEAR
1998	011612	001022				BNE	3\$: IF NOT ASSUME IT IS NOT DZV
1999	011614	005061	000004			CLR	4(R1)		: GET RID OF TCR BITS
2000									: AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZV11 CSR ADDRESS.
2001	011620	010122				MOV	R1,(R2)+		: STORE CSR IN CORE TABLE
2002	011622	005722				TST	(R2)+		: POP OVER VECTOR STORE AREA
2003	011624	012722	000017			MOV	#17,(R2)+		: SET THE DEFAULT LINE SELECTION PARAMETER
2004	011630	012712	017470			MOV	#17470,(R2)		: SET THE DEFAULT PARAMETERS
2005	011634	012223				MOV	(R2)+,(R3)+		: COPY PARAMETERS INTO ETABLE DESCRIPTOR
2006	011636	005022				CLR	(R2)+		: SET THE DEFAULT MODE OF OPERATION
2007	011640	012712	177777			MOV	#-1,(R2)		: TERMINATE LIST
2008	011644	105237	001414			INCB	DZVNUM		: UPDATE DEVICE COUNTER
2009	011650	122737	000020	001414		CMPB	#20,DZVNUM		: ARE MAX. NO. OF DEV FOUND?
2010	011656	001405				BEQ	100\$: YES DON'T LOOK FOR ANY MORE.
2011	011660	062701	000010		3\$:	ADD	#10,R1		: UPDATE CSR POINTER ADDRESS
2012	011664	062701	164000			CMP	#164000,R1		
2013	011670	001321				BNE	2\$: BR IF MORE ADDRESS TO CHECK.
2014	011672				100\$:				
2015	011672	105737	001414			TSTB	DZVNUM		: WERE ANY DZV11'S FOUND AT ALL?
2016	011676	001430				BEQ	5\$: ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
2017	011700	113701	001414			MOV	DZVNUM,R1		
2018	011704	012737	000001	001410		MOV	#1,SAVACTV		: CREATE A BIT MAP OF
2019	011712	005301			4\$:	DEC	R1		: THE DEVICES IN THE SYSTEM
2020	011714	001404				BEQ	98\$		
2021	011716	000361				SEC			
2022	011720	006137	001410			ROL	SAVACTV		
2023	011724	000772				BR	4\$		
2024	011726	013737	001500	001174	98\$:	MOV	DZCR0,\$BASE		: POINT TO THE ADDRESS OF FIRST DEVICE
2025	011734	013737	001510	001202		MOV	MANT0,\$CDW2		: INDICATE TO ETABLE WHAT MODE IS BEING USED
2026	011742	012737	000006	000004	99\$:	MOV	#6,\$#4		: RESTORE TRAP VECTOR
2027	011750	013737	001410	001176		MOV	SAVACTV,\$DEVN		: SAVE ACTIVE REGISTER
2028	011756	000410				BR	VECMAP		: GO FIND THE VECTOR NOW.
2029	011760	104402	007513		5\$:	TYPE	,MERR2		: NOTIFY OPR THAT NO DZV11'S FOUND.
2030	011764	000000				CLR	R0		: MAKE DATA DISPLAY ZERO
2031	011766	000000				HALT			: STOP THE SHOW
2032	011770	000776				BR	.-2		: DISABLE CONT. SW.
2033	011772	012716	011660		6\$:	MOV	#3\$,(\$P)		: ENTERED BY NON-EXISTENT TIME-OUT
2034	011776	000002				RTI			: RETURN TO MAINSTREAM
2035									
2036	012000	012737	000200	000022	VECMAP:	MOV	#MASK,\$#22		: SET IOT TRAP PRIORITY
2037	012006	012737	012122	000020		MOV	#4\$,,\$#20		: SET IOT TRAP VECTOR
2038	012014	012702	001500			MOV	#DZV.MAP,R2		: SET SOFTWARE POINTER
2039	012020	012700	000300			MOV	#300,R0		: FLOATING VECTORS START HERE.
2040	012024	012701	000302			MOV	#302,R1		: PC OF IOT INSTR.
2041	012030	010120			1\$:	MOV	R1,(R0)+		: START FILLING VECTOR AREA
2042	012032	012721	000004			MOV	#4,(R1)+		: WITH .+2; IOT

J05

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 43
 DVDZBA.P11 28-JUL-77 07:37 POWER DOWN AND UP ROUTINES

SEQ 0061

2043	012036	022021				CMP	(R0)+,(R1)+	:ADD 2 TO R0 +R1
2044	012040	020127	001000			CMP	R1,#1000	:HAS THE VECTOR AREA BEEN EXCEEDED?
2045	012044	101771				BLOS	1\$:BR IF MORE TO FILL
2046	012046	013704	001410			MOV	SAVACTV,R4	:STORE TEMPORARILY
2047	012052	006004			2\$:	ROR	R4	:BRING OUT A BIT
2048	012054	103036				BCC	5\$:BR IF ALL DONE
2049	012056	106427	000000			MPS	#0	:ZERO CPU PRIO
2050	012062	012772	040040	000000		MOV	#BIT14+BITS,2(R2)	:SET TIE AND MAS SCAN
2051	012070	011201				MOV	(R2),R1	:GET CSR
2052	012072	112761	000017	000004		MOVB	#17,4(R1)	:SET THE TCR BITS FOR ALL LINES
2053								:ATTEMPT TO FORCE AN INTERRUPT
2054	012100	005200				INC	R0	:STALL
2055	012102	001376				BNE	.-2	:FOR TIME TO INTERRUPT
2056	012104	012762	000300	000002		MOV	#300,2(R2)	:NO INTERRUPT ASSUME 300 AND FIX DZV11 LATER
2057	012112	000005				RESET		:INIT
2058	012114	062702	000012		3\$:	ADD	#12,R2	:POP SOFTWARE POINTER
2059	012120	000754				BR	2\$:KEEP GOING
2060	012122	011662	000002		4\$:	MOV	(SP),2(R2)	:GET VECTOR ADDRESS
2061	012126	162762	000010	000002		SUB	#10,2(R2)	:POINT BACK TO THE CORRECT VECTOR
2062	012134	042762	000007	000002		BIC	#7,2(R2)	:CLEAR JUNK
2063	012142	022626				POP2SP		:POP IOT JUNK OFF STACK
2064	012144	012716	012114			MOV	#3\$, (SP)	:SET FOR RETURN
2065	012150	000002				RTI		
2066	012152	013737	001502	001170	5\$:	MOV	DZVCO,\$VECT1	:COPY VECTOR OF FIRST DEVICE INTO ETABLE
2067	012160	012737	004300	000020		MOV	#.SCOPE,IOTVEC	:RESTORE THE SCOPE TRAP
2068	012166	000207				RTS	PC	:ALL DONE WITH "AUTO SIZING"
2069								

K05

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 44
 DVDZBA.P11 28-JUL-77 07:37

DZV11 DEVICE DIAGNOSTICS. COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0062

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
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125

012170	000004		
012172	012737	000001	001246
012200	012737	012632	001362
012206	012737	012546	001364
012214	104417		
012216	104421		
012220	005037	001374	
012224	104422		
012236	012702	000001	
012232	027777	010040	167550
012240	030237	001366	
012244	001533		
012246	013700	001374	
012252	006300		
012254	010277	167544	
012260	105777	167524	
012264	100001		
012266	104020		
012270	005003		
012272	005004		
012274	005777	167510	
012300	100404		
012302	104414		
012304	005204		
012306	001372		
012310	104003		
012312	116077	001426	167514
012320	005260	001426	
012324	020327	000017	
012330	103006		
012332	032777	020000	167450
012340	001413		
012342	104013		
012344	000411		
012346	005004		
012350	032777	020000	167432
012356	001304		
012360	104414		
012362	005204		
012364	001371		

```

***** TEST 1 *****
*THIS TEST VERIFIES OVERRUN AND SILO ALARM
*ONE LINE AT A TIME - BASED UPON VALID LINES
*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
*CHAR PULLED OUT OF THE SILO.
*ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
*ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
*USED TO SCOPE SILO ALARM PULSES, ETC.

::* TEST 1
*****
TST1: SCOPE
      MOV      #1,STSTM      ;LOAD THE NUMBER OF THIS TEST
      MOV      #TST2,NEXT   ;POINT TO THE START OF THE NEXT TEST
      MOV      #18$,LOCK    ;SET FOR LOOP
      DCLASH      ;SET DCLR IN CSR AND SET MNTFLG
      LPRSET      ;LOAD LINE PARAMETERS
      CLR      SAVLIN      ;INIT LINE INDICATOR
      BUFSET      ;ZERO DATA BUFFER
      MOV      #1,R2        ;LINE POINTER
      BIS      #SENAB,SILOEN, @DZVCSR ;START SCANNER & SET SILO ENABLE
3$:   BIT      R2,LINE      ;VALID LINE?
      BEQ      21$,        ;IF NOT GO TO NEXT LINE
      MOV      SAVLIN,R0    ;MAKE OFFSET
      ASL      R0          ;MAKE POWER OF TWO
      MOV      R2,@DZVTCR  ;SET TCR BIT
4$:   TSTB     @DZVCSR     ;REC DONE = 1 ?
      BPL      .+4
      ERROR    20         ;REC DONE SHOULD NOT = 1
      CLR      R3         ;SET CHARACTER COUNT
5$:   CLR      R4
6$:   TST      @DZVCSR     ;IS TROY SET?
      BMI      7$         ;IF YES, LOAD CHAR.
      DELAY   R4          ;WAIT FOR TROY TO SET
      INC     R4          ;INC DELAY COUNTER
7$:   ERROR    3         ;*TROY FAILED TO SET
      MOVB   TDO(R0),@DZVTDR ;LOAD A CHARACTER
      INC   TDO(R0)       ;SET UP NEXT CHARACTER
      CMP   R3,#15.      ;16 CHARACTERS ?
      BHS   8$
      BIT   #SILOAL,@DZVCSR ;SILO ALARM = 0 ?
      BEQ   10$          ;YES
      ERROR 13         ;*SILO ALARM SHOULD NOT = 1
                        ;UNTIL 16. DATA CHARACTERS
8$:   BR     10$
      CLR   R4
9$:   BIT   #SILOAL,@DZVCSR
      BNE   10$
      DELAY R4
      INC   R4
      BNE   9$
  
```

L05

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37
 DVDZBA.P11 28-JUL-77 07:37

28-JUL-77 07:37 PAGE 45
 DZV11 DEVICE DIAGNOSTICS.

COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0063

```

2126 012366 104014          ERROR 14          ;*SILO ALARM FAILED TO SET!
2127                                ;SILO ALARM SHOULD =1 AFTER 16.
2128                                ;DATA CHARACTERS
2129 012370 005203          10$: INC R3          ;INC CHAR COUNT
2130 012372 022703 000102  CMP #66.,R3      ;FINISHED SENDING CHARACTERS ?
2131 012376 001335          BNE 5$          ;NO
2132 012400 005004          CLR R4
2133 012402 104414          DELAY
2134 012404 105204          INCB R4
2135 012406 001375          BNE -4
2136                                ;NOW LETS READ THE SILO
2137 012410 013705 001374  MOV SAVLIN,R5    ;MAKE EXPECTED LINE #
2138 012414 005737 001372  TST MODE        ;IS THIS TEST IN STAGGERED MODE?
2139 012420 100006          BPL 13$        ;IF NOT, SKIP STAGGERED SETUP
2140
2141                                ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
2142
2143 012422 006205          ASR R5          ;GET THE LAST BIT INTO THE CARRY BIT
2144 012424 103402          BCS 11$        ;IF IT IS SET, GO CLEAR IT
2145 012426 000261          SEC          ;IF IT IS CLEAR SET IT HERE
2146 012430 000401          BR 12$        ;SKIP THE CLEARING
2147 012432 000241          11$: CLC          ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
2148 012434 006105          12$: ROL R5      ;GET THE NEW BIT BACK INTO R5
2149 012436 000305          13$: SWAB R5      ;PUT IN UPPER BYTE
2150 012440 052705 100000  BIS #DVALID,R5 ;ADD DATA VALID
2151 012444 017704 167344  14$: MOV @DZVRBUF,R4 ;ACTUAL
2152 012450 020405          CMP R4,R5      ;ACTUAL VS. EXPECTED
2153 012452 001401          BEQ 15$        ;YES
2154 012454 104006          ERROR 6        ;*DATA/CONTENTS DID NOT COMPARE
2155 012456 032777 020000 167324 15$: BIT #SILOAL,@DZVCSR ;SILO ALARM= 0 ?
2156 012464 001401          BEQ 16$        ;YES
2157 012466 104016          ERROR 16       ;READING DZVRBUF DID NOT CLEAR SILO ALARM
2158 012470 005205          16$: INC R5
2159 012472 120527 000077  CMPB R5,#63.    ;UP CHARACTER
2160 012476 101762          BLOS 14$       ;LAST SILO CHAR ?...64TH CHAR
2161 012500 005205          INC R5
2162 012502 052705 040000  BIS #OVRUN,R5 ;ADD OVERRUN TO EXPECTED
2163 012506 120527 000101  CMPB R5,#65.    ;LAST CHARACTER ?
2164 012512 001754          BEQ 14$
2165 012514 017704 167274  MOV @DZVRBUF,R4 ;FOR GOOD MEASURE
2166 012520 005704          TST R4        ;DATA VALID SHOULD = 0
2167 012522 100001          BPL 17$        ;YES
2168 012524 104017          ERROR 17       ;DATA VALID SHOULD = 0
2169 012526 040277 167272  17$: BIC R2,@DZVTCR ;CLR TCR BIT
2170 012532 104401          SCOP1
2171 012534 005237 001374  21$: INC SAVLIN ;INC EXPECTED LINE
2172 012540 104420          SHIFT
2173 012542 000137 012240  JMP 3$         ;NEXT LINE
2174
2175                                ;TIGHT SCOPE LOOP FOR THIS TEST. SENDS 20. CHARACTERS
2176                                ;ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
2177                                ;USED TO SCOPE SILO ALARM PULSES, ETC.
2178
2179 012546 052777 010040 167234 18$: BIS #MSENAB!SILOEN,@DZVCSR ;SETUP DEVICE
2180 012554 012777 012622 167262  MOV #20,@DZVTIV ;SETUP TRANSMITTER VECTOR
2181 012562 012701 000024  MOV #20.,R1     ;TEMPORARY COUNT OF CHARACTER BURST

```


M05

```

2182 012715 000777 167232      BIS      R2, 20ZVTCR      ; ENABLE LINE
2183 012716 000777 040000 167210  BIS      #TIE, 20ZVCSR   ; ENABLE INTERRUPTS
2184 012700 106427 000000      MTPS     #0             ; LOWER PRIORITY
2185 012714 000001      WAIT     ; ALLOW INTERRUPTS
2186 012706 077102      SOB      R1, 19$      ; REDUCE COUNT. ALL CHARACTERS SENT?
2187 012710 042777 050040 167172  BIC      #SILOEN!#SENAB!TIE, 20ZVCSR ; RESET SILO COUNTER, CLEAR STROBE
2188 012616 104401      SCOP1    ; LOOP AGAIN?
2189 012620 000742      BR       17$         ; IF NOT, RETURN TO WHERE YOU LEFT OFF
2190 012622 112777 000252 167204 20$:  MOVB     #252, 20ZVTOR ; SEND A CHARACTER
2191 012630 000002      RTI     ; ALLOW MORE CHARACTERS TO COME
2192      ; ***** TEST 2 *****
2193      ; *THIS TEST THAT "SILO ENABLE" WILL INHIBIT
2194      ; *RECEIVER INTERRUPTS AND THAT ON THE
2195      ; *16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
2196      ; *INTERRUPT WITH "RIE" SET.
2197      ; *THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
2198      ; *ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
2199
2200      ;::* TEST 2
2201      ;*****
2201 012632 000004      TST2:   SCOPE
2202 012634 012737 000002 001246  MOV      #2, $TSTNM    ; LOAD THE NUMBER OF THIS TEST
2203 012642 012737 013132 001362  MOV      #TST3, NEXT  ; POINT TO THE START OF THE NEXT TEST
2204 012650 012737 012674 001364  MOV      #3$, LOCK    ; SET FOR LOOP
2205 012656 104417      DCLASM   ; SET DCLR IN CSR AND SET MNTFLG
2206 012660 104421      LPRSET   ; LOAD LINE PARAMETERS
2207 012662 005037 001374  CLR      SAVLIN       ; INIT LINE INDICATOR
2208 012666 104422      BUFSET  ; ZERO DATA BUFFER
2209 012670 012702 000001      MOV      #1, R2      ; LINE POINTER
2210 012674 012777 013104 167136 3$:  MOV      #1$, 20ZVRIV ; SET FOR UNEXPECTED INTER.
2211 012702 012777 000200 167132  MOV      #MASK, 20ZVRIS ; SET PRIO.
2212 012710 052777 010140 167072  BIS      ##SENAB!#SILOEN!RIE, 20ZVCSR ; START SCANNER & SET SILO ENABLE
2213
2214 012716 030237 001366      BIT      R2, LINE     ; VALID LINE?
2215 012722 001477      BEQ     18$         ; IF NOT GO TO NEXT LINE
2216 012724 005777 167064      TST     20ZVRBUF     ; EMPTY THE SILO
2217 012730 100775      BMI     .-4        ; BR IF DATA VALID IS SET!
2218 012732 106427 000000      MTPS     #0         ; SET PROCESSOR PRIORITY TO 0
2219 012736 013700 001374      MOV      SAVLIN, R0  ; MAKE OFFSET
2220 012742 006300      ASL     R0          ; MAKE POWER OF TWO
2221 012744 010277 167054      MOV      R2, 20ZVTCR ; SET TCR BIT
2222 012750 005004      CLR     R4
2223 012752 005777 167032 5$:  TST     20ZVCSR
2224 012756 100404      BMI     7$
2225 012760 104414      DELAY
2226 012762 005204      INC     R4
2227 012764 001372      BNE     6$
2228 012766 104003      ERROR   3
2229 012770 116077 001426 167036 7$:  MOVB     T00(R0), 20ZVTOR ; *TRDY FAILED TO SET
2230 012776 005260 001426      INC     T00(R0)     ; LOAD A CHARACTER
2231 013002 022760 000017 001426  CMP      #15., T00(R0) ; SET UP NEXT CHARACTER
2232 013010 001406      BEQ     8$         ; 15 CHARS YET?
2233 013012 032777 020000 166770  BIT      #SILOAL, 20ZVCSR ; SILO ALARM = 0 ?
2234 013020 001401      BEQ     .+4        ; YES
2235 013022 104013      ERROR   13        ; *SILO ALARM SHOULD NOT = 1
2236
2237 013024 000752      BR      6$         ; UNTIL 16. DATA CHARACTERS
  
```

N05

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 47

DVDZBA.P11 28-JUL-77 07:37

DZV11 DEVICE DIAGNOSTICS.

COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0065

2238	013026	012777	013112	167004	8\$:	MOV	#12\$, 20ZVRIV	;SET NEW VECTOR
2239	013034	005777	166750			TST	20ZVCSR	;READY FOR 16TH CHAR
2240	013040	100375				BPL	-4	
2241	013043	016077	001426	166764		MOV	T00(R0), 20ZVTDR	;LOAD THE 16TH CHAR.
2242	013050	005004				CLR	R4	
2243	013052	032777	020000	166730	9\$:	BIT	#SILOAL, 20ZVCSR	
2244	013060	001005				BNE	10\$	
2245	013063	104414				DELAY		
2246	013064	005204				INC	R4	
2247	013066	001371				BNE	9\$	
2248	013070	104014				ERROR	14	;*SILO ALARM FAILED TO SET!
2249	013072	000410				BR	17\$;SILO ALARM SHOULD =1 AFTER 16.
2250								;DATA CHARACTERS
2251	013074	000240			10\$:	NOP		;STALL
2252	013076	000240				NOP		
2253	013100	104027				ERROR	27	;SILO ALARM NOT INTERRUPTING.
2254	013102	000404				BR	17\$;CONTINUE TEST.
2255	013104	022626			11\$:	POP2SP		;FAKE RTI
2256	013106	104012				ERROR	12	;RX SHOULD NOT INTERRUPT
2257	013110	000401				BR	17\$;CONTINUE
2258	013112	022626			12\$:	POP2SP		;GOOD INTERRUPT TO HERE.
2259	013114	040277	166704		17\$:	BIC	R2, 20ZVTDR	;CLR TCR BIT
2260	013120	104401				SCOP1		;LOOP?
2261	013122	005237	001374		18\$:	INC	SAVLIN	;INC EXPECTED LINE
2262	013126	104420				SHIFT		;NEXT LINE
2263	013130	000661				BR	3\$;YES

2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319

013132 000004
013134 012737 000003 001246
013142 012737 013674 001362
013150 104417
013152 013737 001366 013672
013160 013737 001366 013412
013166 104421
013170 104422
013172 012777 013414 166640
013200 012777 000200 166634
013206 012777 013300 166630
013214 012777 000200 166624
013222 052777 040140 166560
013230 113777 001366 166566
013236 106427 000000

013242 005037 013276
013246 104414
013250 105737 013672
013254 001002
013256 000137 013620
013262 005237 013276
013266 001367
013270 104007
013272 104011
013274 104400
013276 000000

013300 117703 166506
013304 042703 177774
013310 010304
013312 010337 001374
013316 005777 166466
013322 100401
013324 104003
013326 012702 000001
013332 105303
013334 100402
013336 006302
013340 000774
013342 030237 001366
013346 001001
013350 104015
013352 030237 013412
013356 001003
013360 040277 166440
013364 000411

```
***** TEST 3 *****  
: THIS TEST RUNS ALL LINES FULL BORE  
: BASED UPON QUALIFIED LINES  
: THIS IS AN INTERRUPT TEST ON THE RECEIVER AND  
: TRANSMITTER  
: : TEST 3  
: *****  
TST3: SCOPE  
MOV #3, $STSNM ; LOAD THE NUMBER OF THIS TEST  
MOV #TST4, NEXT ; POINT TO THE START OF THE NEXT TEST  
DCLASH ; SET DCLR IN CSR AND SET MNTFLG  
MOV LINE, RXTCR ; SET IMAGE OF TCR BITS  
MOV LINE, TXTCR ; SET IMAGE OF TCR BITS  
LPRSET ; LOAD LINE PARAMETERS  
BUFSET ; ZERO DATA BUFFER  
MOV #RXSVC, @DZVRIV ; SET UP REC INTR VECTOR  
MOV #MASK, @DZVRIS ; STATUS  
MOV #TXSVC, @DZVTIV ; SET UP TRANS INTR VECTOR  
MOV #MASK, @DZVTIS ; STATUS  
BIS #SENAB, RIE, @DZVCSR ; SET MASTER SCAN ENABLE  
MOVB LINE, @DZVTCR ; SET TCR BITS  
MTPS #CLEAR ; ALLOW INTERRUPTS  
  
SNAP: CLR 4$ ; CLEAR DELAY COUNTER  
2$: DELAY ; WAIT FOR RECEIVERS TO FINISH  
TSTB RXTCR ; WAIT FOR ALL RECIEVERS TO FINISH  
BNE 3$  
JMP OUT  
3$: INC 4$ ; INCREMENT DELAY COUNTER  
BNE 2$ ; DELAY FINISHED?  
ERROR 7 ; *TRANSMITTER FAILED TO INTERRUPT  
ERROR 11 ; *RECEIVER FAILED TO INTERRUPT  
ADVANCE ; LEAVE THIS TEST  
4$: 0  
  
: TRANS INTR SVC ROUTINE  
TXSVC: MOVB @DZVCSR, R3 ; FIND LINE NO.  
BIC #C(3), R3 ; ISOLATE LINE NO.  
MOV R3, R4 ; SAVE LINE NO.  
MOV R3, SAVLIN ; SAVE LINE NO.  
TST @DZVCSR ; TRANS READY SET ?  
BMI +4  
ERROR 3 ; *TRANSMITTER FAILED  
MOV #1, R2 ; SET UP POSITION POINTER  
3$: DECB R3 ; IS IT THIS LINE ?  
BMI 4$ ; YES  
ASL R2 ; UP THE LINE #  
BR 3$ ; GO 'ROUND AGAIN  
4$: BIT R2, LINE ; VALID LINE ?  
BNE +4 ; YES  
ERROR 15 ; NO INVALID LINE!!!!  
BIT R2, TXTCR ; DATA FINISHED?  
BNE 6$ ; IF NOT SEND CHAR.  
BIC R2, @DZVTCR ; CLEAR TCR BIT  
BR 5$ ; RETURN
```


2376
2377
2378
2379 013620 106427 000200
2380 013624 104413
2381 013626 005003
2382 013630 005037 001374
2383 013634 012702 000001
2384 013640 030237 001366
2385 013644 001405
2386 013646 022763 000400 001436
2387 013654 001401
2388 013656 104030
2389
2390 013660 005237 001374
2391 013664 005723
2392 013666 104420
2393 013670 000763
2394 013672 000000
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417 013674 000004
2418 013676 012737 000004 001246
2419 013704 012737 014312 001362
2420 013712 012737 014032 001364
2421 013720 132737 000001 001140
2422 013726 001405
2423 013730 005737 001126
2424 013734 001402
2425 013736 000177 165420
2426 013742 012737 000002 001354 10S:
2427 013750 005037 015314
2428 013754 005037 001374
2429 013760 005037 001376
2430 013764 012702 000001
2431 013770 012703 010070

```

:FINISH UP ROUTINE
OUT:  MTPS  #MASK          ;STOP ALL INTERRUPTS
      DEVICE.CLR          ;CLEAR ALL INTERRUPTS AWAY
      CLR   R3
      CLR   SAVLIN
      MOV   #1,R2
1S:   BIT   R2,LINE        ;VALID LINE ?
      BEQ   2S             ;NO
      CMP   #400,TR0(R3)  ;RECEIVED A BINARY COUNT PATTERN ?
      BEQ   .+4            ;YES
      ERROR 30            ;THE LINE FAILED TO RECEIVE A FULL
                          ;BINARY COUNT PATTERN
2S:   INC   SAVLIN        ;SET UP FOR NEXT LINE
      TST   (R3)+         ;ADD 2
      SHIFT
      BR    1S            ;SET UP NEXT LINE POINTER
RXTCR: 0                 ;FINISHED ?
                          ;RX IMAGE OF TCR BITS

```

```

***** TEST 4 *****
* DZV11 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 THEN TESTED.
* WHEN RUNNING UNDER THE APT MANUFACTURING SYSTEM
* THIS TEST IS ONLY RUN THE FIRST PASS

```

```

::* TEST 4
*****
TST4: SCOPE
      MOV   #4,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
      MOV   #TST5,NEXT    ;POINT TO THE START OF THE NEXT TEST
      MOV   #3S,LOCK      ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
      BITB  #1,$ENV       ;RUNNING UNDER APT?
      BEQ   10S           ;IF NOT CONTINUE WITH TEST
      TST   $PASS         ;IF YES IS THIS FIRS. PASS
      BEQ   10S           ;IF NOT 1ST PASS SKIP TEST
10S:  JMP   @NEXT
      MOV   #2,$TIMES     ;SET UP FOR 2 ITERATIONS
      CLR   OFFSET        ;RESET THIS VARIABLE
      CLR   SAVLIN        ;RESET LINE NUMBER INDICATOR
      CLR   XMTLIN        ;USE THIS WORD TO TELL WHAT LINE TRANSMITTED
      MOV   #1,R2         ;USE R2 AS A BIT POINTER
      MOV   @R0VON!550!EIGHT!TWOSTOP,R3 ;BUILD TEMPORARY PARAMETERS

```

2432	013774	030237	001366	15:	BIT	R2,LINE	: IS THIS LINE ACTIVE?	
2433	014000	001014			BNE	35	: IF SO, GO GET STARTED	
2434	014002	012703	010070	25:	MOV	#RCVON!SSO!EIGHT!	: TWO STOP, R3 : LOAD PARAMETERS TEMPORARILY	
2435	014006	005237	001376		INC	XMTLIN	: POINT TO THE NEXT LINE TO TRANSMIT	
2436	014012	042703	000007		BIC	#7, R3	: MAKE SURE TEMPORARY PARAMETERS POINT TO 0	
2437	014016	053703	001376		BIS	XMTLIN, R3	: ADD DESIRED LINE NUMBER	
2438	014022	005037	015314		CLR	OFFSET		
2439	014026	104420			SHIFT		: POINT TO THE NEXT LINE	
2440	014030	000761			BR	15	: PROCESS THE NEXT LINE	
2441	014034			35:				
2442	014038	134417			DCLASM		: CLEAR DEVICE AND SET MAINT BIT IF I MODE	
2443	014042	042703	010000		BIC	#RCVON, R3	: ZERO PARAMETERS FOR TX LINE	
2444	014046	010377	165754		MOV	R3, #ZVLP	: LOAD PARAMETERS FOR TX	
2445	014050	005737	001372		TST	MODE	: STAGGERED?	
2446	014054	101007			BPL	1005	: BR IF NO	
2447	014058	000241			CLC		: SET UP LINE	
2448	014062	005003			ROR	R3		
2449	014066	103002			BCC	985	: BR IF LINE WAS EVEN	
2450	014070	000241			CLC		: PREPARE TO MAKE LINE EVEN	
2451	014074	000401			BR	995	: CONTINUE	
2452	014078	000261		985:	SEC		: PREPARE TO MAKE LINE ODD	
2453	014082	006103		995:	ROL	R3	: SET ALTERED LINE	
2454	014086	052703	010000	1005:	BIS	#RCVON, R3	: SET RX ON	
2455	014090	010377	165720		MOV	R3, #ZVLP	: LOAD RX PARAMETERS	
2456	014094	010337	001374		MOV	R3, SAVLIN	: SET FOR RECEIV. LINE	
2457	014098	042737	177774	001374	BIC	#1<(3), SAVLIN	: ISOLATE LINE NO.	
2458	014102	042703	000003		BIC	#3, R3	: CLEAR OLD LINE #	
2459	014106	053703	001376		BIS	XMTLIN, R3	: SET LINE UP AGAIN	
2460	014110	010337	001402		MOV	R3, REGIST	: SAVE PARAMETERS FOR PRINTOUT	
2461	014114	104422			BUFSET		: ZERO DATA BUFFER	
2462	014118	005037	001342		CLR	\$TMP0	: USE \$TMP0 TO COUNT TOTAL NUMBER OF TRANSMISSIONS	
2463	014122	005037	001344		CLR	\$TMP1	: INITIALIZE THE TIMER	
2464	014126	005037	001350		CLR	\$TMP3	: INITIALIZE THESE BITS ALSO	
2465	014130	012737	000020	001400	MOV	#20, XMTCNT	: SET HOW MANY CHARACTERS TO TRANSMIT	
2466	014134	012777	014742	165664	MOV	#XMTSRV, #ZVTIV		
2467	014138	012777	015112	165652	MOV	#RXISR1, #ZVRIV		
2468	014142	012777	000200	165646	MOV	#MASK, #ZVRIS		
2469	014146	012777	000200	165644	MOV	#MASK, #ZVTIS		
2470	014150	110277	165616		MOVB	R2, #ZVTCR	: START THE VALID LINE	
2471	014154	052777	040140	165574	BIS	#TIE!RIE!MSENA8, #ZVCSR		
2472	014158	106427	000000		MTPS	#0	: LOWER THE PRIORITY TO ALLOW INTERRUPTS	
2473	014162	032777	000100	165562	45:	BIT	#RIE, #ZVCSR	: IS ROUTINE DONE?
2474	014166	001407			BEQ	55	: WHEN ALL IS DONE RX IE IS CLEARED IN ISR.	
2475	014170	005237	001344		INC	\$TMP1	: INCREMENT TIMER	
2476	014174	001371			BNE	45	: WHEN IT OVERFLOWS	
2477	014178	005237	001350		INC	\$TMP3	: CATCH CARRY	
2478	014182	001366			BNE	45	: CONTINUE TEST	
2479	014186	104011			ERROR	11	: INTERRUPTS NOT FINISHED	
2480	014190	004737	006772	55:	JSR	PC, SERV.G	: (<G>?)	
2481	014194	104401			SCOPI		: LOOP?	
2482	014198	062737	000002	015314	ADD	#2, OFFSET		
2483	014202	022703	017400		CMP	#17400, R3		
2484	014206	003006			BGT	65		
2485	014210	032703	000030		BIT	#BIT4+BIT3, R3	: IS CHARACTER SIZE DONE?	
2486	014214	001642			BEQ	25		
2487	014218	162703	000010		SUB	#BIT3, R3		

```

014302 000653
014304 062703 000400
014310 000650
014312 000004
014314 012737 000005 001246
014322 012737 004064 001362
014330 005737 001372
014334 100131
014336 105037 001425
014342 104413
014344 013701 001370
014350 042701 000200
014354 052701 000100
014360 012702 000001
014364 030237 001366
014370 001420
014372 105737 001425
014376 001004
014400 032701 000001
014404 001006
014406 000403
014410 032701 000001
014414 001402
014416 012701 000200
014426 010177 165372
014428 042701 000200
014432 012701
014436 000302
014438 032702 000020
014442 001750
014444 005037 001374
014450 012701 001342
014454 012701
014456 012737 000040 001400
014464 104422
014466 012777 014742 165350
014474 012777 014622 165336
014502 012777 000200 165332
014510 012777 000200 165330
014516 052777 040140 165264
014524 113777 001366 165272
014532 106427 000000
014536 005037 014614
014542 005037 014616
014546 032777 000100 165234

```

```

65: BR 35
ADD #400,R3
BR 35
***** TEST 5 *****
*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(B) CHARS ARE USED FOR THIS TEST.
*ALL SELECTED LINES WILL BE ENABLED AT THE SAME TIME.
*THIS TEST FIRST CHECKS EVEN PARITY FOR ODD LINES AND
*ODD PARITY FOR EVEN LINES, THEN IT CHECKS THE REVERSE.
::* TEST 5
*****
T5T5: SCOPE
MOV #5,STSTNM ;LOAD THE NUMBER OF THIS TEST
MOV #SEOP,NEXT ;POINT TO THE END-OF-PASS HANDLER
TST MODE ;IS THIS STAGGERED MODE?
BPL 175 ;IF NOT, DON'T DO THIS TEST
CLRB DONFLG ;SET UP FOR FIRST TEST PASS
145: DEVICE.CLR ;SET DCLR IN CSR
MOV PAR,R1 ;USE R1 TO BUILD PARAMETERS TO BE LOADED
BIC #000PAR,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
15: BIT R2,LINE ;IS THIS A VALID LINE?
BEQ 35 ;IF NOT, SKIP TO THE NEXT LINE
TSTB DONFLG ;FIRST PASS THROUGH TEST?
BNE 155 ;IF NO BRANCH
BIT #BIT0,R1 ;IS THIS LINE AN ODD LINE?
BNE 25 ;IF IT'S ODD, USE EVEN PARITY
BR 165 ;IF EVEN SET FOR ODD PARITY
155: BIT #BIT0,R1 ;IF THE LINE IS EVEN SET FOR EVEN PAR.
BEQ 25 ;GO LOAD PARAMETER
165: BIS #000PAR,R1 ;IF IT'S ODD, USE ODD PARITY
25: MOV R1,20ZVLP ;LOAD THE LINE PARAMETER REGISTER
BIC #000PAR,R1 ;SET UP THE NEXT PARITY TO EVEN
35: INC R1 ;POINT TO THE NEXT LINE
ASL R2
BIT #BIT4,R2 ;ALL LINES DONE?
BEQ 15 ;IF NOT, GO CHECK THE NEXT LINE
CLR SAVLIN ;CLEAR THE LINE NUMBER INDICATOR
CLR $TMPD ;USE $TMPD 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)
BUFSET ;ZERO BUFFER A A
MOV #XMTSRV,20ZVTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
MOV #95,20ZVRIV ;SET UP THE RECEIVER INTERRUPT VECTOR
MOV #MASK,20ZVRIS ;SET THE INTERRUPT VECTOR STATUS
MOV #MASK,20ZVTIS ;SET TRANSMITTER INTERRUPT PRIORITY
BIS #RIE!TIE!MSENAB,20ZVCSR ;ENABLE THE DEVICE
MOV# LINE,20ZVTCR ;ENABLE ALL SELECTED LINES
MTPS #0 ;ALLOW INTERRUPTS
45: CLR 75
CLR 85
55: BIT #RIE,20ZVCSR ;WHEN RX DONE; RIE WILL =0

```


2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629

014742	117701	165044	
014746	042701	177774	
014752	013705	001374	
014756	010137	001374	
014762	006301		
014764	023761	001400	001426
014772	003414		
014774	005777	165010	
015000	100401		
015002	104003		
015004	116177	001426	165022 2S:
015012	005261	001426	
015016	002237	001342	
015022	000415		
015024	012700	000001	6S:
015030	006201		
015032	005301		12S:
015034	100402		
015036	006300		
015040	000774		
015042	140077	164756	3S:
015046	001003		
015050	042777	040000	164732
015056	010537	001374	7S:
015062	000002		

 ; TRANSMITTER INTERRUPT SERVICE

```

XMTSRV:  MOVB  @DZVCSR,R1      ; GET THE LINE NUMBER.
          BIC   #C(3),R1      ; CLEAR JUNK
          MOV   SAVLIN,R5     ; SAVE REC. LINE NO.
          MOV   R1,SAVLIN     ; LOAD TRANS LINE NO FOR ERROR PRINTOUT
          ASL   R1            ; ADJUST R1 FOR OFFSET
          CMP   XMTCNT,TDO(R1) ; HAVE ALL CHAR. BEEN SENT
          BLE   6S           ; IF YES GO CLEAR TCR
          TST   @DZVCSR       ; TRDY SET?
          BMI   2S           ; IF YES GO LOAD CHAR.
          ERROR 3             ; *TRANSMITTER NOT READY- FALSE INTERRUPT
          MOVB  TDO(R1),@DZVTDR ; LOAD THE CURRENT CHARACTER FOR THIS LINE
          INC   TDO(R1)       ; SET UP NEXT CHARACTER FOR THIS LINE
          INC   $TMP0         ; UP THE NUMBER OF TRANSMISSIONS
          BR    7S           ; GO RETURN
          MOV   #1,R0         ; SET UP A DESELECTION POINTER
          ASR   R1            ; GET LINE NO. AGAIN
          DEC   R1            ; REDUCE THE COUNT. WAS THIS THE LINE?
          BMI   3S           ; IF SO, GO DISABLE THE ENABLE BIT FOR IT
          ASL   R0            ; MOVE THE POINTER TO THE NEXT LINE
          BR    12S          ; GO CHECK THE NEXT LINE
          BICB  R0,@DZVTCR    ; DISABLE THE LINE POINTED TO BY R0
          BNE   7S           ; IF MORE LINES ARE ACTIVE, GO CONTINUE TRANSMIT
          BIC   #TIE,@DZVCSR  ; IF NOT, DISABLE TRANSMITTER INTERRUPTS
          MOV   R5,SAVLIN     ; RESTORE RECEIV. LINE
          RTI                    ; RETURN TO THE TIMING LOOP
    
```

 ; RELATIVE TIME BUILDING ROUTINE

```

BUILD:  MOV   #2,$TMP2       ; ROTATE 2 BITS BACK INTO $TMP1
1S:     ROR   $TMP3         ; GET THE BITS FROM $TMP3, THE HIGH BYTE
          ROR   $TMP1         ; OF THE RELATIVE TIME COUNTER. PUT THEM BACK
          DEC   $TMP2         ; INTO $TMP1 USING THE CARRY BIT WITH
          ROTATE INSTRUCTIONS ; ROTATE INSTRUCTIONS
          BNE   1S           ; REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
          RTS   PC           ; RETURN TO CALLING TEST
    
```

;RECEIVER SERVICE ROUTINE

```

2630
2631
2632 015112 105777 164672 RXISR1: TSTB 20ZVCSR ; IS THE RECEIVER REALLY READY?
2633 015116 100401 BMI 15 ; IF SO, GO SERVICE IT
2634 015120 104004 ERROR 4 ; ERROR- RECEIVER DONE FLAG ISN'T SET
2635 015122 017704 164666 15: MOV 20ZVRBUF,R4 ; SAVE THE RECEIVER INFORMATION
2636 015126 100401 BMI 25 ; IF IT WAS VALID, GO PROCESS IT
2637 015130 104023 ERROR 23 ; ERROR- DATA VALID WASN'T SET
2638 015132 032704 040000 25: BIT 80VRRUN,R4 ; OVERRUN ERROR FLAG SET?
2639 015136 001401 BEQ 65 ; IF NOT DON'T TYPE ERROR
2640 015140 104024 ERROR 24 ; OVERRUN ERROR
2641 015142 032704 020000 65: BIT 8FRMERR,R4 ; FRAMING ERROR FLAG SET?
2642 015146 001401 BEQ 95 ; IF NOT DON'T TYPE ERROR
2643 015150 104025 ERROR 25 ; FRAMING ERROR
2644 015152 032704 010000 95: BIT 8PARER,R4 ; PARITY ERROR FLAG SET?
2645 015156 001401 BEQ 35 ; IF NOT, GO CONTINUE PROCESSING
2646 015160 104026 ERROR 26 ; ERROR- RECEIVER E...OR FLAG SET
2647 015162 013701 001374 35: MOV SAVLIN,R1 ; CALCULATE THE DATA OFFSET
2648 015166 006301 ASL R1 ; ALIGN IT ON A WORD BOUNDARY
2649 015170 120461 001436 CMPB R4,TR0(R1) ; IS THE CHARACTER WHAT IT SHOULD BE?
2650 015174 001407 BEQ 45 ; IF SO, GO CONTINUE PROCESSING
2651 015176 116105 001436 MOVB TR0(R1),R5 ; GET WHAT WAS EXPECTED FOR ERROR REPORTING
2652 015178 042705 177400 BIC 8(C(377)),R5 ; ELIMINATE PROPAGATED SIGN
2653 015180 042704 177400 BIC 8(C(377)),R4 ; ISOLATE THE ACTUAL CHARACTER
2654 015182 104005 ERROR 5 ; *DATA ERROR
2655 015184 001436 45: INC TR0(R1) ; SET UP THE NEXT EXPECTED CHARACTER
2656 015186 122761 000020 001436 CMPB 820,TR0(R1) ; HAVE ALL CHARACTERS BEEN RECEIVED?
2657 015188 001031 BNE 85 ; IF NOT RETURN
2658 015190 126137 001436 001342 CMPB TR0(R1),STMP0 ; ALL CHARAC. RECEIVED?
2659 015192 001025 BNE 85 ; IF SO, GO DETERMINE THE TIMING
2660 015240 004737 015064 JSR PC,BUILD ; GET THE RELATIVE TIME (SIGNIFICANT BITS)
2661 015244 001700 015314 MOV OFFSET,R0 ; GET POINTER
2662 015250 013760 001344 002050 MOV STMP1,TMTBL(R0) ; SAVE THIS TEST'S TIME
2663 015256 005737 015314 TST OFFSET ; FIRST TEST?
2664 015262 001410 BEQ 75 ; IF NOT, GO CHECK THE TIME
2665 015264 005740 TST -(R0) ; POINT TO THE PREVIOUS TIME TAKEN
2666 015266 026037 002050 001344 CMP TMTBL(R0),STMP1 ; IS THIS TIME WHAT IT SHOULD BE?
2667 015274 101003 BHI 75 ; IF SO, GO TO THE NEXT TEST
2668 015276 016005 002050 MOV TMTBL(R0),R5 ; PLACE WHAT WAS EXPECTED IN R5
2669 015302 104021 ERROR 21 ; TIMING ERROR
2670 015304 042777 000140 164476 75: BIC 8RIE!MSENAB,20ZVCSR ; DISABLE THE DEVICE
2671 015312 000002 85: RTI ; RETURN TO THE PROGRAM
2672 015314 000000 OFFSET: 0

```

JOB

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 56
 D:\DZBA.P11 28-JUL-77 07:37

DZV11 DEVICE DIAGNOSTICS. COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0074

Address	Code 1	Code 2	Code 3	Code 4	Description
2673					
2674	015316	000000			
2675	015320	000000			
2676	015322	000000			
2677					
2678	015324	015544		EM1	;ERROR
2679	015326	016670		DH1	
2680	015330	017070		DT1	
2681					
2682	015332	015617		EM2	;ERROR 2
2683	015334	016714		DH2	
2684	015336	017102		DT2	
2685					
2686	015340	015645		EM3	;ERROR 3
2687	015342	016747		DH3	
2688	015344	017120		DT3	
2689					
2690	015346	015704		EM4	;ERROR 4
2691	015350	016747		DH3	
2692	015352	017120		DT3	
2693					
2694	015354	015733		EM5	;ERROR 5
2695	015356	016761		DH4	
2696	015360	017126		DT4	
2697					
2698	015362	015762		EM6	;ERROR 6
2699	015364	016761		DH4	
2700	015366	017126		DT4	
2701					
2702	015370	000000		0	
2703	015372	000000		0	
2704	015374	000000		0	
2705					
2706	015376	000000		0	
2707	015400	000000		0	
2708	015402	000000		0	
2709					
2710	015404	016021		EM11	;ERROR 11
2711	015406	016747		DH3	
2712	015410	017120		DT3	
2713					
2714	015412	000000		0	
2715	015414	000000		0	
2716	015416	000000		0	
2717					
2718	015420	016057		EM13	;ERROR 13
2719	015422	016747		DH3	
2720	015424	017120		DT3	
2721					
2722	015426	016110		EM14	;ERROR 14
2723	015430	016747		DH3	
2724	015432	017120		DT3	
2725					
2726	015434	016142		EM15	;ERROR 15
2727	015436	000000		0	
2728	015440	000000		0	

K06

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37 PAGE 57
DVDZBA.P11 28-JUL-77 07:37

DZV11 DEVICE DIAGNOSTICS. COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0075

2729				
2730	015442	016204	EM16	
2731	015444	016747	DH3	
2732	015446	017120	DT3	
2733				
2734	015450	016256	EM17	;ERROR 17
2735	015452	016747	DH3	
2736	015454	017120	DT3	
2737				
2738	015456	016314	EM20	
2739	015460	016747	DH3	
2740	015462	017120	DT3	
2741				
2742	015464	016355	EM21	;ERROR 21
2743	015466	017010	DH5	
2744	015470	017144	DT5	
2745				
2746	015472	000000	0	
2747	015474	000000	0	
2748	015476	000000	0	
2749				
2750	015500	016405	EM23	;ERROR 23
2751	015502	016747	DH3	
2752	015504	017120	DT3	
2753				
2754	015506	016435	EM24	
2755	015510	016747	DH3	
2756	015512	017120	DT3	
2757				
2758	015514	016463	EM25	
2759	015516	016747	DH3	
2760	015520	017120	DT3	
2761				
2762	015522	016513	EM26	
2763	015524	016747	DH3	
2764	015526	017120	DT3	
2765				
2766	015530	016542	EM27	
2767	015532	016747	DH3	
2768	015534	017120	DT3	
2769				
2770	015536	016610	EM30	
2771	015540	016747	DH3	
2772	015542	017120	DT3	

```

2773 ;ERROR MESSAGES
2774 015544 047200 020117 052502 EM1: .ASCIZ <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
      015617 200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE/
      015645 200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
      015704 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
      015733 200 040504 046524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
      015762 042200 053132 030461 EM6: .ASCIZ <200>/DZV11 *RECEIVER BUFFER* ERROR/
      016021 200 042522 042503 EM11: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
      016057 200 044523 047514 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
      016110 051600 046111 020117 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
      016142 040600 052103 047511 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
      016204 051200 041505 044504 EM16: .ASCIZ <200>/READING DZVF IF DID NOT CLEAR SILO ALARM/
      016256 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
      016314 051200 041505 041505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
      016355 200 041505 041505 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
      016405 200 041505 041505 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
      016435 200 041505 041505 EM24: .ASCIZ <200>/DATA OVERFLOW IS SET!/
      016463 200 051106 046501 EM25: .ASCIZ <200>/FORMING ERROR OCCURRED/
      016513 200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
      016542 051600 046111 020117 EM27: .ASCIZ <200>/SILO ALARM FAILED TO CAUSE INTERRUPT/
      016610 046200 047111 020105 EM30: .ASCIZ <200>/LINE DID NOT RECEIVE FULL BINARY COUNT PATTERN/

      016670 052200 040522 020120 DM1: .ASCIZ <200>/TRAP PC DZV11 REG/
      016714 042600 050130 041505 DM2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
      016747 200 044514 042516 DM3: .ASCIZ <200>/LINE NO./
      016761 200 054105 042520 DM4: .ASCIZ <200>/EXPECTED FOUND LINE/
      017010 052200 020130 044514 DM5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/
  
```

017070 .EVEN

```

2775 017070 000002 DT1: 2 ;DATA TABLES FOR ERROR MESSAGES
2776 017072 006 003 .BYTE 6,3
2777 017074 001330 $REG1
2778 017076 006 001 .BYTE 6,1
2779 017100 001326 $REG0
2780
2781 017102 000003 DT2: 3
2782 017104 006 004 .BYTE 6,4
2783 017106 001340 $REG5
2784 017110 006 001 .BYTE 6,1
2785 017112 001336 $REG4
2786 017114 006 001 .BYTE 6,1
2787 017116 001326 $REG0
2788
2789 017120 000001 DT3: 1
2790 017122 003 001 .BYTE 3,1
2791 017124 001374 SAVLIN
2792
2793 017126 000003 DT4: 3
2794 017130 006 004 .BYTE 6,4
2795 017132 001340 $REG5
2796 017134 006 001 .BYTE 6,1
2797 017136 001336 $REG4
2798 017140 003 001 .BYTE 3,1
2799 017142 001374 SAVLIN
2800
  
```

M06

MD-11-DVDZB-A MACY11 30(1046) 28-JUL-77 07:37
DVDZBA.P11 28-JUL-77 07:37

28-JUL-77 07:37 PAGE 59

DZV11 DEVICE DIAGNOSTICS. COPYRIGHT 1977 DIGITAL EQUIP. CORP.

SEQ 0077

```

2801 017144 000004
2802 017146 000003 005
2803 017150 001374
2804 017152 000006 011
2805 017154 001340
2806 017156 000006 007
2807 017160 001344
2808 017162 000006 001
2809 017164 001402
2810
2811
2812
2813 017166 002450
2814 017170 001560
2815 017172 001120
2816 017174 000750
2817 017176 000660
2818 017200 000330
2819 017202 000150
2820 017204 000060
2821 017206 000040
2822 017210 000030
2823 017212 000020
2824 017214 000010
2825 017216 000001
2826 017220 000001
2827 017222 000001
2828 017224 000001
2829
2830
2831
2832
2833 017226 000001
2834

```

```

DTS: 4
      .BYTE 3,5
      SAVLIN
      .BYTE 6,9.
      $REG5
      .BYTE 6,7
      $TMP1
      .BYTE 6,1
      REGIST

```

TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

```

DLYTBL: 2450      : TIME FOR 50 BAUD
          1560      : TIME FOR 75 BAUD
          1120      : TIME FOR 110 BAUD
           750      : TIME FOR 134 BAUD
           660      : TIME FOR 150 BAUD
           330      : TIME FOR 300 BAUD
           150      : TIME FOR 600 BAUD
            60      : TIME FOR 1200 BAUD
            40      : TIME FOR 1800 BAUD
            30      : TIME FOR 2000 BAUD
            20      : TIME FOR 2400 BAUD
            10      : TIME FOR 3600 BAUD
             1      : TIME FOR 4800 BAUD
             1      : TIME FOR 7200 BAUD
             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 LSI11.

CORMAX:
.END

ABASE	=	160010	1#	342	383						
ACOM1	=	000017	1#	342	385						
ACOM2	=	000000	342	386							
ACPUOP	=	000000	342	357							
ACTIVE	=	001420	500#	756*	1789*	1790	1792*	1796	1797*	1798	1801*
ROOM0	=	017470	1#	342	387						
ROOM1	=	017470	1#	342	388						
ROOM10	=	017470	1#	342	397						
ROOM11	=	017470	1#	342	398						
ROOM12	=	017470	1#	342	399						
ROOM13	=	017470	1#	342	400						
ROOM14	=	017470	1#	342	401						
ROOM15	=	017470	1#	342	402						
ROOM2	=	017470	1#	342	389						
ROOM3	=	017470	1#	342	390						
ROOM4	=	017470	1#	342	391						
ROOM5	=	017470	1#	342	392						
ROOM6	=	017470	1#	342	393						
ROOM7	=	017470	1#	342	394						
ROOM8	=	017470	1#	342	395						
ROOM9	=	017470	1#	342	396						
AEVCT	=	000000	342	348							
AEVH	=	000001	1#	342	384						
ADACT	=	005661	1323*	1360*	1370#						
ADVANC	=	104400	652#	1500	2297	2557					
RENV	=	000000	342	353							
RENVH	=	000000	342	354							
AFATAL	=	000000	342	345							
AMADR1	=	000000	342	370							
AMADR2	=	000000	342	374							
AMADR3	=	000000	342	377							
AMADR4	=	000000	342	380							
AMANS1	=	000000	342	354							
AMANS2	=	000000	342	372							
AMANS3	=	000000	342	375							
AMANS4	=	000000	342	378							
AMSCAD	=	000000	342	350							
AMSCAL	=	000000	342	351							
AMSGTY	=	000000	342	344							
AMTYP1	=	000000	342	365							
AMTYP2	=	000000	342	373							
AMTYP3	=	000000	342	376							
AMTYP4	=	000000	342	379							
APASS	=	000000	342	347							
APRION	=	000000	342								
APTCSU	=	000040	1174	1279#							
APTENV	=	000001	1167	1235	1277#	1587					
APTSIZ	=	000200	1276#								
APTSPO	=	000100	1169	1237	1278#						
ASWREG	=	000000	342	355							
ATESTN	=	000000	342	346							
AUNIT	=	000000	342	349							
AUSWR	=	000000	342	356							
AUTO.S	=	011464	931	1971#							
AVECT1	=	000300	1#	342	381						
AVECT2	=	000000	342	382							

SFNDCT	004220	1040#																
SEMV	001140	353#	1167	1235	1259	1587	2421											
SEVMH	001141	354#	772	1169	1174	1237												
SEOP	004064	1014#	2504															
SEOPCT	004212	1037#	1041															
SEFLG	001247	416#	755#	1017#	1074	1103	1105#	1129	1542#	1556	1572#							
SEMAX	001261	422#	1129															
SEORR	006344	313	1531#															
SEFAPC	001262	423#	759#	1016#	1082#	1539	1541#											
SEWRTM	001263	474#																
SERTTL	001256	420#	758#	1064	1599#													
SETABL	001140	352#																
SETEND	001244	405#	541															
SFATAL	001122	345#	1263#															
SFFLG	005334	1226#	1229#	1257	1266#	1274#												
SFILLC	001322	441#	1192	1223														
SFILLS	001321	440#	1223															
SFLIP	177777	1#	2071#	2083#	2192#	2199#	2264#	2269#	2397#	2415#	2491#	2500#						
SGOADR	001264	424#																
SGOAT	001270	425#																
SGE142	004224	4#																
SHO	000001	10	11															
SHIBTS	001446	536#																
SICNT	001250	417#	1111#	1112	1114#	1128												
SILLUP	007402	1669	1685	1704#														
SINTAG	001301	431#																
SITEMB	001260	421#	1547#	1589														
SLF	001360	459#	1223															
SLFLG	005333	1267#	1273#															
SLPADR	001252	418#	761#	998#	1001	1118#	1120	1128	1605#	1607	1832#	1833#	1841#	1843				
SLPE	001254	419#																
SLP	001152	370#																
SLT	001156	374#																
SLT	001162	377#																
SLT	001166	380#																
SLT	001120	343#	537	541	1117	1167												
SLT	001150	364#																
SLT	001154	372#																
SLT	001160	375#																
SLT	001164	378#																
SLT	001450	537#																
SLG	005332	1227#	1233	1268#	1272#													
SLG	001134	350#	1243#	1246														
SLG	001136	351#	1248#															
SLTY	001120	344#	1241	1249#	1261	1265#												
SHYPI	001151	365#																
SHYPI	001155	373#																
SHYPI	001161	376#																
SHYPI	001165	379#																
SHXCNT	004542	1115	1128#	1784														
SN	000005	1#	2071	2083	2088#	2192	2199	2204#	2264	2269	2274#	2397	2415	2421#				
SN		2491	2500	2505#	2810#													
SNLL	001320	439#	1194	1223														
SNWTST	000000	2084#	2200#	2270#	2416#	2501#												
SOVER	004500	1087	1090	1101	1113	1119#												
SPASS	001126	347#	754#	1023#	1026#	1034#	1035#	1052	1061	1109	1129	2423						

SSCHM	3378	451	452	453	454	455
SSFSCA	1548					
SSK.P	1548	2084	2200	2270	2416	2501
.EQUAT	18	44				
.HEADE	18					
.SETUP	18					
.SACT1	18	317				
.SAPT8	18	3398				
.SAPTH	18	520				
.SAPTY	18	1223				
.SCATC	18					
.SCHTA	3378					
.SEOP	18	1007				
.SERRO	18					
.SPUNE	18	1665				
.SSCOP	18	1069				
.STRAP	18					
.STYPE	18	1144				

. ABS. 017226 000

ERRORS DETECTED: 0

DVDZBA, DVDZBA, SEQ=DVDZBA.P11
RUN-TIME: 19 11 1 SECONDS
RUN-TIME RATIO: 119/32=3.6
CORE USED: 34K (67 PAGES)

C08