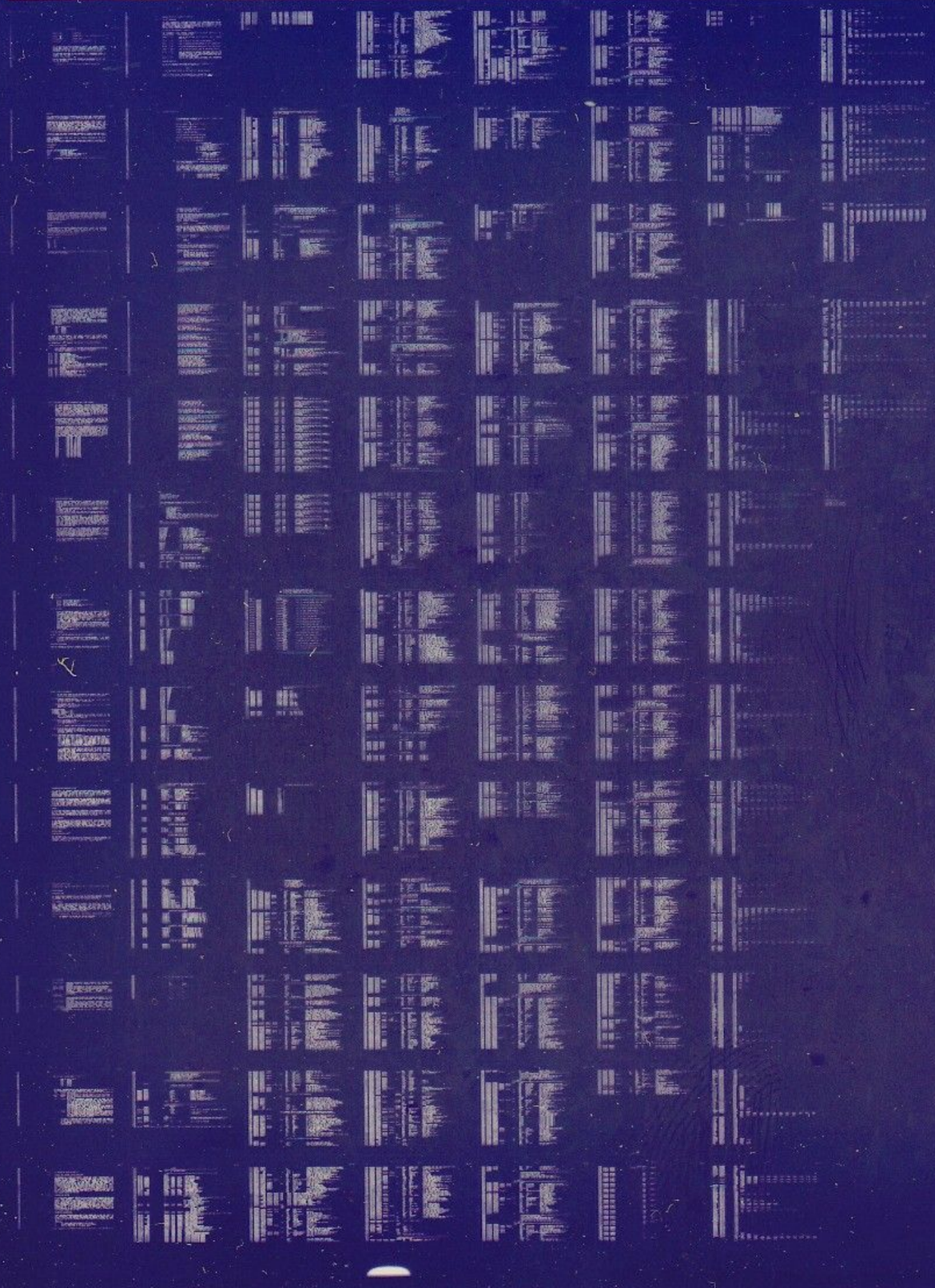
DZV11

DZV-11
CVDZAB0

AH-A878B-MC
FICHE 1 OF 1



```
.REM  &
```

IDENTIFICATION

PRODUCT CODE:	AC-A877B-MC
PRODUCT NAME:	CVDZAB0 DZV11 4 LINE ASYNC MUX TESTS PART 1 OF 2
DATE RELEASED:	JANUARY 1981
MAINTAINER:	DIAGNOSTIC ENGINEERING

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 (CVDZA,CVDZB AND CVDZC) ONE SYSTEM MODULE FOR DEC X/11 (CXDZBA), AND AN OVERLAY FOR ATEP (CVDZD).

CVDZA TOGETHER WITH CVDZB WILL TEST ALL LOGICAL FUNCTIONS OF THE DZV11 INTERFACE MODULE.
CVDZC 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.
VSR 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 (SW00=1); OR AFTER THE 'AUTO SIZING' HAS BEEN DONE. THESE LOCATIONS MAY BE CHANGED IF THE USER UNDERSTANDS THEIR MEANING AND DIFFERENT PARAMETERS ARE REQUIRED.

3. LOADING PROCEEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED USING THE ABSOLUTE LOADER. NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH AS DISK ,MAGTAPE,DECTAPE, OR CASSETTE; FOLLOW INSTRUCTIONS FOR THE MONITOR WHICH HAS BEEN PROVIDED ON THAT SPECIFIC MEDIA.

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY	* SIZE
4K	17
8K	37
12K	57
16K	77
20K	117
24K	137
28K	157

3.1.1 STARTING THE PROCESSOR AT THE ABSOLUTE LOADER STARTING ADDRESS WILL LOAD THE DIAGNOSTIC INTO MEMORY.

4. STARTING PROCEEDURE

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(8). 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 8.4 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' (^G) 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 LSI11 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 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 'SCOPE' 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 TROUBLE-SHOOTER 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 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOP1'). 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 ENABELED; 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 PROCEEDURE

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 H325 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 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 '$TSTNM' (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 CVDZA-B 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.

8.3 KEY LOCATIONS

$LPADR (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 PEFORMED.

$TSTNM (1246) CONTAINS THE NUMBER OF THE TEST NOW BEING PEFORMED.

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 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) 1406/0000000000011111 MEANS THAT DZV11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DZVACTV) 1406/0000000000010001 MEANS THAT DZ11 NO. 00,04 WILL BE TESTED.

$BASE (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 OCCURES, 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 NXM TRAP), 'MASTER SCAN ENABLE'' IS ATTEMPTED TO BE SET AND THE TCR BITS FOR ALL FOUR LINES ARE SET. 'TRDY'' 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 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 INTERUPT ENABLE AND MSTSCAN ENABLE) ARE SET INTO THE DZVCSR. ALL TCR BITS ARE SET, A DELAY OCCURS, AND IF NO INTERUPT OCCURES (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 INTERUPT OCCURRED, THE ADDRESS TO WHICH THE DZV11 INTERUPTED 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 IF 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 (CVDZA, AND CVDZB) CAN BE RUN AS STANDALONE DIAGNOSTICS OR IN EITHER OF THE APT MODES. CVDZC, 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
SDEVM -(1176)	A BIT MAP REPRESENTING WHICH DZV11'S WILL BE TESTED
SCSW1 -(1200)	USED TO INDICATE WHICH LINES TO RUN ON ALL DZV11'S
SCSW2 -(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).
SDSW0 -(1204)	EACH OF THE SDSW 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.

```
592
595
596                 '0.0    PROGRAM DESCRIPTION
597
598
599
600
601                         THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
602                         PACKAGE (MAINDEC-11-DZQAC-C3).
603
604
605                         INITIAL ADDRESS OF THE STACK POINTER ••• 1120 •••
606
607                         MISCELLANEOUS DEFINITIONS
608
609                         GENERAL PURPOSE REGISTER DEFINITIONS
610
611                         PRIORITY LEVEL DEFINITIONS
612
613                         'SWITCH REGISTER' SWITCH DEFINITIONS
614
615                         DATA BIT DEFINITIONS (BIT00 TO BIT15)
616
617                         BASIC 'CPU' TRAP VECTOR ADDRESSES
618
619                                                 BITS 15-11=CPU TYPE
620                                                         11/04=01,11/05=02,11/20=03,11/40
621                                                         11/70=05,PDQ=07,Q=10
622                                                 BIT 10=REAL TIME CLOCK
623                                                 BIT  9=FLOATING POINT PROCESSOR
624                                                 BIT  8=MEMORY MANAGEMENT
625
626                                                 MEM.TYPE BYTE   --  (HIGH BYTE)
627                                                         900 NSEC CORE=001
628                                                         300 NSEC BIPOLAR=002
629                                                         500 NSEC MOS=003
630
631                                                  MEM.LAST ADDR.=3 BYTES,THIS WORD AND L
632
633                 THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
634                 USED IN THE PROGRAM.
635
636                 THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCU
637                 THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
638                 LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE I
639                 NOTE1:  IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
640                 NOTE2:  EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS
641
642                         EM              ;;POINTS TO THE ERROR MESSAGE
643                         DH              ;;POINTS TO THE DATA HEADER
644                         DT              ;;POINTS TO THE DATA
645                         DF              ;;POINTS TO THE DATA FORMAT
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
693
694
695
696
697
```

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

THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7
AND LOAD THE ERROR FLAG (SERFLG) 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

ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 B
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE
NOTE1:  $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CH
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
OR
        TYPE
        MESADR


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.

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)


********************** TEST 1 ******************************
THIS TEST PROVES THE BUS REPLY RESPONSE
DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
        DZVCSR, DZVRBUF, DZVTCR, DZVMSR

********************** TEST 2 ******************************
THIS TEST PROVES THAT BIT 'DCLR''
CAN BE SET AND THAT IT WILL CLEAR
BY ITSELF
```

```
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
```

********** TEST 3 **********
TEST TO VERIFY THAT THE R/W BITS OF THE
DZVCSR REGISTER CAN BE SET. THEN VERIFY THAT
THESE BITS CAN BE CLEARED. AND FINALLY, VERIFY
THAT AFTER BEING SET AGAIN THEY CAN BE
CLEARED BY A 'DEVICE CLEAR''.
THE BITS TESTED ARE: MAINT, MSENAB, SILOEN,
RIE, AND TIE.

********** TEST 4 **********
THIS TESTS THAT ALL OF THE TCR BITS
CAN BE: SET, CLEARED, AND CLEARED BY A DEVICE CLEAR.
THIS TEST ALSO DETERMINES IF THE DTR BITS CAN
BE SET, CLEARED, AND CLEARED BY A RESET.

********** TEST 5 **********
THIS TEST VERIFIES THAT
BITS 'RDONE,TRDY, BIT9, BIT8,
AND SILOAL'' ARE READ ONLY AND THAT TRDY IS
ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.

********** TEST 6 **********
THIS TEST VERIFIES THAT:
TIE,SILOEN,RIE,MSENAB,AND MAINT ARE THE
ONLY R/W BITS IN THE DZVCSR AND THAT
SETTING 'DCLR'' IN THE CSR WILL CLEAR THESE BITS.

********** TEST 7 **********
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZVRBUF
AND TESTING OF WRITE ONLY REGISTER DZVLPR

********** TEST 10 **********
THIS TEST PERFORMS RESET TESTING AND
TESTING OF READ ONLY REGISTER DZVMSR
AND TESTING OF WRITE ONLY REGISTER DZVTDR

********** TEST 11 **********
VERIFY THAT SETTING 'DTR'' FOR A LINE WILL
BRING UP 'CO'' AND 'RING'' FOR:
THE SAME LINE IF IN EXTERNAL MODE
THE STAGGERED LINE IF IN STAGGERED MODE.
LINES ARE STAGGERED AS FOLLOWS:
LINE0 WITH LINE1; LINE2 WITH LINE3.
THIS TEST IS ONLY RUN IF AN H325,OR H329
IS CONNECTED ON THE DZV UNDER TEST.

********** TEST 12 **********
THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
IS READY TO BE LOADED, AND THAT THE LINE SPECI-
FIED IN BITS 8-9 OF DZVCSR CORRESPOND
TO THE LINE SELECTED IN DZVTCR

```
754
755
756                     ********** TEST 13 **********
757     TEST TO TRANSMIT ONE CHAR AND
758     RECEIVE ONE CHAR ON ONE LINE
759     AT A TIME. THE CHAR IS '252' AND
760     ALL SELECTED LINES WILL BE TURNED ON .
761
762     THIS IS THE FIRST TIME ANY
763     DATA IS CHECKED IN THE RECEIVER.
764     USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
765     WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.
766
767                     ********** TEST 14 **********
768     THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
769     DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
770     TO ZERO FOR EACH LINE.
771     THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
772     EMPTIED BY ISSUING A DEVICE MASTER CLEAR.
773
774                     ********** TEST 15 **********
775      THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
776     CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
777     (ONE LINE AT A TIME   BASED UPON VALID LINES)
778     THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
779
780                     ********** TEST 16 **********
781     THIS TEST WILL PROVE THAT:
782      1) THE TRANSMITTER 'BREAK BIT' WORKS
783      2) THE RECEIVER CAN FLAG 'FRAMING ERRORS'
784      3) THE RECEIVER CAN FLAG 'PARITY ERRORS'
785     ONLY ONE LINE AT A TIME WILL BE EXERCISED.
786
787                     ********** TEST 17 **********
788      THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
789     WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
790     BUT WILL INTERRUPT IF THE PROCESSOR STATUS
791     ALLOWS INTERRUPTS.
792
793                     ********** TEST 20 **********
794     THIS TEST VERIFIES THAT THE RECEIVER WILL
795     INTERRUPT BEFORE THE TRANSMITTER EVEN
796     THOUGH THE TRANSMITTER WAS ENABLED
797     FIRST.  SET PS TO HIGH (MASK INTERRUPTS);
798     GET RDONE AND TRDY TO SET;
799     SET TX IE AND RX IE;
800     CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
801
802
803                     &
```

```
                        .TITLE  CVDZA-B
                        ;*COPYRIGHT (C) 1977,1981
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
000001                  $TN=1
                                ;STARTING PROCEDURE
                                ;LOAD PROGRAM
                                ;LOAD ADDRESS 000200
                                ;PRESS START
                                ;PROGRAM WILL TYPE
                                ;''CVDZAB/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 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 TYPEOUT
                        SW12=10000              ;=1,DELETE TYPEOUT/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 INITAL 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 ***
001120                  STACK=  1120
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
```

```
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

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

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0
```

```
                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                          BIT15=  100000
040000                          BIT14=  40000
020000                          BIT13=  20000
010000                          BIT12=  10000
004000                          BIT11=  4000
002000                          BIT10=  2000
001000                          BIT09=  1000
000400                          BIT08=  400
000200                          BIT07=  200
000100                          BIT06=  100
000040                          BIT05=  40
000020                          BIT04=  20
000010                          BIT03=  10
000004                          BIT02=  4
000002                          BIT01=  2
000001                          BIT00=  1
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;"TRAP" TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR


                                ;INSTRUCTION DEFINITIONS
                                ;-----------------------

005746                          PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
005726                          POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
010046                          PUSHR0=10046    ;SAVE R0 ON STACK
012600                          POPR0=12600     ;RESTORE R0 FROM STACK
024646                          PUSH2SP=24646   ;DECREMENT STACK TWICE
022626                          POP2SP=22626    ;INCREMENT STACK TWICE
000200                          MASK=BIT7       ;SET INTERRUPT MASK (INHIBIT FURTHER INTERRUPTS)
000000                          CLEAR=0         ;ALLOW INTERRUPTS (CLEAR PROCESSOR STATUS)
```

```
                                ;DZV11 CONTROL AND STATUS REGISTER DEFINITIONS
                                ;(DZVCSR)       BIT DEFINITIONS
                                ;-----------------------------

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

                                ;DZVCSR WORD DEFINITIONS
                                ;----------------------
000000          TL0=0           ;TRANSMIT LINE 0
000400          TL1=BIT8        ;TRANSMIT LINE 1
001000          TL2=BIT9        ;TRANSMIT LINE 2
001400          TL3=BIT9!BIT8   ;TRANSMIT LINE 3


                                ;DZVRBUF BIT DEFINITIONS
                                ;-----------------------

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


                                ;DZVRBUF WORD DEFINITIONS
                                ;-----------------------

000000          RL0=0           ;RECEIVER LINE 0
000400          RL1=BIT8        ;RECEIVER LINE 1
001000          RL2=BIT9        ;RECEIVER LINE 2
001400          RL3=BIT9!BIT8   ;RECEIVER LINE 3

                                ;DZVLPR WORD DEFINITIONS
                                ;----------------------

000000          LP0=0           ;LINE PARAMETER 0
000001          LP1=BIT0        ;LINE PARAMETER 1
000002          LP2=BIT1        ;LINE PARAMETER 2
000003          LP3=BIT1!BIT0   ;LINE PARAMETER 3

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

000100          PARITY=BIT6             ;PARITY ENABLED
```

```
1028    000200          ODDPAR=BIT7             ;ODD PARITY ENABLED
1029    000000          ONESTOP=0               ;ONE STOP BIT ENABLED
1030    000040          TWOSTOP=BIT5            ;TWO STOP BITS ENABLED
1031    000000          EVEPAR=0                ;EVEN PARITY ENABLED
1032    010000          RCVON=BIT12             ;ENABLE RECEIVER (RECEIVER ON)
1033
1034    000000          S50=0                   ;SPEED 50 BAUD
1035    000400          S75=BIT8                ;SPEED 75 BAUD
1036    001000          S110=BIT9               ;SPEED 110 BAUD
1037    001400          S134=BIT9!BIT8          ;SPEED 134.5 BAUD
1038    002000          S150=BIT10              ;SPEED 150 BAUD
1039    002400          S300=BIT10!BIT8         ;SPEED 300 BAUD
1040    003000          S600=BIT10!BIT9         ;SPEED 600 BAUD
1041    003400          S1200=BIT10!BIT9!BIT8   ;SPEED 1200 BAUD
1042    004000          S1800=BIT11             ;SPEED 1800 BAUD
1043    004400          S2000=BIT11!BIT8        ;SPEED 2000 BAUD
1044    005000          S2400=BIT11!BIT9        ;SPEED 2400 BAUD
1045    005400          S3600=BIT11!BIT9!BIT8   ;SPEED 3600 BAUD
1046    006000          S4800=BIT11!BIT10       ;SPEED 4800 BAUD
1047    006400          S7200=BIT11!BIT10!BIT8  ;SPEED 7200 BAUD
1048    007000          S9600=BIT11!BIT10!BIT9  ;SPEED 9600 BAUD
1049    007400          S19200=BIT11!BIT10!BIT9!BIT8    ;SPEED 19200 BAUD
1050
1051                    ;DZVTCR BIT DEFINITIONS
1052                    ;----------------------
1053    000001          TCR0=BIT0               ;ENABLE TRANSMISSION ON LINE 0
1054    000002          TCR1=BIT1               ;ENABLE TRANSMISSION ON LINE 1
1055    000004          TCR2=BIT2               ;ENABLE TRANSMISSION ON LINE 2
1056    000010          TCR3=BIT3               ;ENABLE TRANSMISSION ON LINE 3
1057    000400          DTR0=BIT8               ;DATA TERMINAL READY FOR LINE 0
1058    001000          DTR1=BIT9               ;DATA TERMINAL READY FOR LINE 1
1059    002000          DTR2=BIT10              ;DATA TERMINAL READY FOR LINE 2
1060    004000          DTR3=BIT11              ;DATA TERMINAL READY FOR LINE 3
1061
1062                    ;DZVMSR BIT DEFINITIONS
1063                    ;----------------------
1064    000001          RING0=BIT0              ;RING INDICATED ON LINE 0
1065    000002          RING1=BIT1              ;RING INDICATED ON LINE 1
1066    000004          RING2=BIT2              ;RING INDICATED ON LINE 2
1067    000010          RING3=BIT3              ;RING INDICATED ON LINE 3
1068    000400          CO0=BIT8                ;CARRIER PRESENT ON LINE 0
1069    001000          CO1=BIT9                ;CARRIER PRESENT ON LINE 1
1070    002000          CO2=BIT10               ;CARRIER PRESENT ON LINE 2
1071    004000          CO3=BIT11               ;CARRIER PRESENT ON LINE 3
1072
1073                    ;DZVTDR BIT DEFINITIONS
1074                    ;----------------------
1075
1076    000400          BRK0=BIT8               ;BREAK FOR LINE 0
1077    001000          BRK1=BIT9               ;BREAK FOR LINE 1
1078    002000          BRK2=BIT10              ;BREAK FOR LINE 2
1079    004000          BRK3=BIT11              ;BREAK FOR LINE 3
```

```
1080
1081                .TABLE OF LOOP AROUND FUNCTIONS (H325)
1082                :
1083                :       ----------------
1084                :       I               ^
1085                :       V               ^
1086                :       REC             TRANS
1087                :       DATA            DATA
1088                :
1089                :       ----------------
1090                :       I               ^
1091                :       V               ^
1092                :       CO              RTS
1093                :
1094                :       ----------------
1095                :       I               ^
1096                :       V               ^
1097                :       RING            DTR
```

```
1098                                    ;;*****************************************************************
1099                                    ;-----------------------------------------------------------------
1100                                            ;TRAPCATCHER FOR ILLEGAL INTERRUPTS
1101                                            ;THE STANDARD 'TRAP CATCHER'' IS PLACED
1102                                            ;BETWEEN ADDRESS 0 TO ADDRESS 776.
1103                                            ;IT LOOKS LIKE 'PC+2 HALT''.
1104                                    ;-----------------------------------------------------------------
1105                                    ;;*****************************************************************
1106
1107             000000                 .=0
1108                                            ;STANDARD INTERRUPT VECTORS
1109                                            ;-------------------------
1110
1111             000020                 .=20
1112    000020   004314                        .SCOPE                  ;SCOPE LOOP HANDLER
1113    000022   000200                        MASK                    ;HANDLE AT PRIORITY 7
1114    000024   007252                        $PWRDN                  ;POWER FAIL HANDLER
1115    000026   000340                        340                     ;SERVICE AT PRIORITY LEVEL 7
1116    000030   006360                        $ERROR                  ;ERROR HANDLER
1117    000032   000340                        340                     ;SERVICE AT PRIORITY LEVEL 7
1118    000034   006152                        .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
1119    000036   000340                        340                     ;SERVICE AT PRIORITY LEVEL 7
1120                                    .SBTTL  ACT11 HOOKS
1121
1122                                    ;;*****************************************************************
1123                                    ;HOOKS REQUIRED BY ACT11
1124             000040                         $SVPC=.                 ;SAVE PC
1125             000046                         .=46
1126    000046   004250                        $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1127             000052                         .=52
1128    000052   000000                        .WORD   0               ;;2)SET LOC.52 TO ZERO
1129             000040                         .=$SVPC                 ;; RESTORE PC
1130
1131             000174                 .=174
1132    000174   000000                DISPREG:0                       ;SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 11S
1133    000176   000000                SWREG:  0                       ;SOFTARE SWITCH REGISTER FOR SWITCHLESS 11S
1134             000200                 .=200
1135    000200   000137   0021'6               JMP     .START          ;GO TO START OF PROGRAM
1136
1137
1138             001000                 .=1000
1139    001000   005200   053103  055104 MTITLE: .ASCIZ  <200><12>/CVDZAB/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2/<200>
  (2)
```

```
1140            001120                  .=1120
1141                            ;;*****************************************************************
1142                            .SBTTL  APT MAILBOX-ETABLE
1143
1144                            ;;*****************************************************************
1145                            .EVEN
1146    001120                  $MAIL:                          ;;APT MAILBOX
1147    001120  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
1148    001122  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
1149    001124  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
1150    001126  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
1151    001130  000000          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
1152    001132  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
1153    001134  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
1154    001136  000000          $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
1155    001140                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
1156    001140      000         $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1157    001141      000         $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1158    001142  000000          $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1159    001144  000000          $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1160    001146  000000          $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1161                            ;*                              BITS 15-11=CPU TYPE
1162                            ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1163                            ;*                                      11/70=06,PDQ=07,Q=10
1164                            ;*                              BIT 10=REAL TIME CLOCK
1165                            ;*                              BIT  9=FLOATING POINT PROCESSOR
1166                            ;*                              BIT  8=MEMORY MANAGEMENT
1167    001150      000         $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
1168    001151      000         $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
1169                            ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
1170                            ;*                                      900 NSEC CORE=001
1171                            ;*                                      300 NSEC BIPOLAR=002
1172                            ;*                                      500 NSEC MOS=003
1173    001152  000000          $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
1174                            ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE'' ABOVE
1175    001154      000         $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
1176    001155      000         $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
1177    001156  000000          $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
1178    001160      000         $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
1179    001161      000         $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
1180    001162  000000          $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
1181    001164      000         $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
1182    001165      000         $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
1183    001166  000000          $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
1184    001170  000300          $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1185    001172  000000          $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1186    001174  160C10          $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1187    001176  000001          $DEVM:  .WORD   ADEVM           ;;DEVICE MAP
1188    001200  000017          $CDW1:  .WORD   ACDW1           ;;CONTROLLER DESCRIPTION WORD#1
1189    001202  000000          $CDW2:  .WORD   ACDW2           ;;CONTROLLER DESCRIPTION WORD#2
1190    001204  017470          $DDW0:  .WORD   ADDW0           ;;DEVICE DESCRIPTOR WORD#0
1191    001206  017470          $DDW1:  .WORD   ADDW1           ;;DEVICE DESCRIPTOR WORD#1
1192    001210  017470          $DDW2:  .WORD   ADDW2           ;;DEVICE DESCRIPTOR WORD#2
1193    001212  017470          $DDW3:  .WORD   ADDW3           ;;DEVICE DESCRIPTOR WORD#3
1194    001214  017470          $DDW4:  .WORD   ADDW4           ;;DEVICE DESCRIPTOR WORD#4
1195    001216  017470          $DDW5:  .WORD   ADDW5           ;;DEVICE DESCRIPTOR WORD#5
```

```
1196  001220  017470          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1197  001222  017470          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1198  001224  017470          $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
1199  001226  017470          $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
1200  001230  017470          $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
1201  001232  017470          $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
1202  001234  017470          $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
1203  001236  017470          $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
1204  001240  017470          $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
1205  001242  017470          $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
1206
1207
1208  001244                  BE'END.
1209
```

```
1210                                    .SBTTL  COMMON TAGS
1211
1212                                    ;*****************************************************************
1213                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1214                                    ;*USED IN THE PROGRAM.
1215
1216  001244                            $CMTAG:                         ;;START OF COMMON TAGS
1217  001244  000000                            .WORD   0
1218  001246     000                    $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
1219  001247     000                    $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
1220  001250  000000                    $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
1221  001252  000000                    $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
1222  001254  000000                    $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
1223  001256  000000                    $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
1224  001260     000                    $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
1225  001261     001                    $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
1226  001262  000000                    $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1227  001264  000000                    $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
1228  001266  000000                    $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
1229  001270  000000                    $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
1230  001272  000000                    $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
1231  001274  000000                            .WORD   0               ;;RESERVED--NOT TO BE USED
1232  001276  000000                            .WORD   0
1233  001300     000                    $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
1234  001301     000                    $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
1235  001302  000000                            .WORD   0
1236  001304  177570                    $SWR:   .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
1237  001306  177570                    DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
1238  001310  177560                    $TKS:   177560                  ;;TTY KBD STATUS
1239  001312  177562                    $TKB:   177562                  ;;TTY KBD BUFFER
1240  001314  177564                    $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
1241  001316  177566                    $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
1242  001320     000                    $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
1243  001321     002                    $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1244  001322     012                    $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
1245  001323     000                    $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
1246  001324  000000                    $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
1247                                                                    ;;WHICH  ($REG0) WAS OBTAINED
1248  001326  000000                    $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
1249  001330  000000                    $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
1250  001332  000000                    $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
1251  001334  000000                    $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
1252  001336  000000                    $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
1253  001340  000000                    $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
1254  001342  000000                    $TMP0:  .WORD   0               ;;USER DEFINED
1255  001344  000000                    $TMP1:  .WORD   0               ;;USER DEFINED
1256  001346  000000                    $TMP2:  .WORD   0               ;;USER DEFINED
1257  001350  000000                    $TMP3:  .WORD   0               ;;USER DEFINED
1258  001352  000000                    $TMP4:  .WORD   0               ;;USER DEFINED
1259  001354  000000                    $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
1260  001356     077                    $QUES:  .ASCII  /?/             ;;QUESTION MARK
1261  001357     015                    $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1262  001360  000012                    $LF:    .ASCIZ  <12>            ;;LINE FEED
```

```
                                .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 (SERRPC).
                                ;*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


001362                          $ERRTB:

                                        ;PROGRAM CONTROL PARAMETERS
                                        ;--------------------------

001362  000000                  NEXT:   0                       ;ADDRESS OF NEXT TEST TO BE EXECUTED
001364  000000                  LOCK:   0                       ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP

                                        ;PROGRAM VARIABLES
                                        ;-----------------

001366  000017                  LINE:   17                      ;DEFAULT ALL FOUR LINES RUNNING
001370  017470                  PAR:    17470                   ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NO PARIT
001372  000000                  MODE:   0                       ;DEFAULT MAINTENANCE MODE
001374  000000                  SAV IN: 0                       ;LINE NUMBER
001376  000000                  XMT IN: 0                       ;TRANSMISSION LINE NUMBER
001400  000000                  XMTCNT: 0                       ;COUNT OF WORDS IN A TRANSMISSION PATTERN
001402  000000                  REGIST: 0                       ;DEVICE ADDRESS STORAGE LOCATION
001404  000000                  SAVPC:  0                       ;PROGRAM COUNTER STORAGE
001406  000001                  DZVACTV:.BLKW   1               ;*DZV11'S SELECTED ACTIVE.
001410  000001                  SAVACTV:.BLKW   1               ;*A BIT MAP OF DZV11'S IN THE SYSTEM
001412  000001                  RUN:    1                       ;*POINTER ONE PAST RUNNING DEVICE.
001414  000001                  DZVNUM: .BLKB 1                 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM
001415      001                 SAVNUM: .BYTE 1                 ;*WORKABLE NUMBER.
001416  000001                  SAVNO:  .BLKB   1               ;*OCTAL NO. OF DZV11'S BEING TESTED
001420                          .EVEN
001420  001500                  ACTIVE: DZV.MAP                 ;TABLE POINTER.
```

```
1304
1305                                    ;PROGRAM CONTROL FLAGS
1306                                    ;---------------------
1307
1308  001422      000               INIFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG
1309  001423      000               HDRFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
1310  001424      000               MNTFLG: .BYTE   0               ;MAINTENANCE BIT SET FLAG
1311  001425      000               DONFLG: .BYTE   0               ;TRANSMISSION COMPLETION FLAG
1312                                .EVEN
1313                                ;DATA VARIABLES
1314  001426  000000                TD0:    .WORD   0
1315  001430  000000                TD1:    .WORD   0
1316  001432  000000                TD2:    .WORD   0
1317  001434  000000                TD3:    .WORD   0
1318  001436  000000                TR0:    .WORD   0
1319  001440  000000                TR1:    .WORD   0
1320  001442  000000                TR2:    .WORD   0
1321  001444  000000                TR3:    .WORD   0
1322  001446                        STOP:
1323                                .SBTTL  APT PARAMETER BLOCK
1324
1325                                ;;**************************************************************
1326                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1327                                ;;**************************************************************
1328          001446                        .$X=.   ;;SAVE CURRENT LOCATION
1329          000024                        .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1330  000024  000200                        200     ;;FOR APT START UP
1331          000044                        .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1332  000044  001446                        $APTHDR ;;POINT TO APT HEADER BLOCK
1333          001446                        .=.$X   ;;RESET LOCATION COUNTER
1334                                ;;**************************************************************
1335                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1336                                ;INTERFACE SPEC.
1337
1338  001446                        $APTHD:
1339  001446  000000                $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1340  001450  001120                $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1341  001452  000120                $TSTM:  .WORD   80.     ;;RUN TIM OF LONGEST TEST
1342  001454  000024                $PASTM: .WORD   20.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1343  001456  000000                $UNITM: .WORD   0.      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1344  001460  000052                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1345                                        ;DZV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
1346                                        ;------------------------------------------
1347
1348          001500                .=1500
1349  001500                        DZV.MAP:
1350
1351  001500  000001                DZCR0:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 0
1352  001502  000001                DZVC0:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 0
1353  001504  000001                LINE0:  .BLKW   1       ;ALL LINES SELECTED
1354  001506  000001                PAR0:   .BLKW   1       ;PARAMETERS
1355  001510  000001                MANT0:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1356
1357  001512  000001                DZCR1:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 1
1358  001514  000001                DZVC1:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 1
1359  001516  000001                LINE1:  .BLKW   1       ;ALL LINES SELECTED
```

```
1360  001520  000001                PAR1:   .BLKW   1       ;PARAMETERS
1361  001522  000001                MANT1:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1362
1363  001524  000001                DZCR2:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 2
1364  001526  000001                DZVC2:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 2
1365  001530  000001                LINE2:  .BLKW   1       ;ALL LINES SELECTED
1366  001532  000001                PAR2:   .BLKW   1       ;PARAMETERS
1367  001534  000001                MANT2:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1368
1369  001536  000001                DZCR3:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 3
1370  001540  000001                DZVC3:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 3
1371  001542  000001                LINE3:  .BLKW   1       ;ALL LINES SELECTED
1372  001544  000001                PAR3:   .BLKW   1       ;PARAMETERS
1373  001546  000001                MANT3:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1374
1375  001550  000001                DZCR4:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 4
1376  001552  000001                DZVC4:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 4
1377  001554  000001                LINE4:  .BLKW   1       ;ALL LINES SELECTED
1378  001556  000001                PAR4:   .BLKW   1       ;PARAMETERS
1379  001560  000001                MANT4:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1380
1381  001562  000001                DZCR5:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 5
1382  001564  000001                DZVC5:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 5
1383  001566  000001                LINE5:  .BLKW   1       ;ALL LINES SELECTED
1384  001570  000001                PAR5:   .BLKW   1       ;PARAMETERS
1385  001572  000001                MANT5:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1386
1387  001574  000001                DZCR6:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 6
1388  001576  000001                DZVC6:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 6
1389  001600  000001                LINE6:  .BLKW   1       ;ALL LINES SELECTED
1390  001602  000001                PAR6:   .BLKW   1       ;PARAMETERS
1391  001604  000001                MANT6:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1392
1393  001606  000001                DZCR7:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 7
1394  001610  000001                DZVC7:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 7
1395  001612  000001                LINE7:  .BLKW   1       ;ALL LINES SELECTED
1396  001614  000001                PAR7:   .BLKW   1       ;PARAMETERS
1397  001616  000001                MANT7:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1398
1399  001620  000001                DZCR10: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 10
1400  001622  000001                DZVC10: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 10
1401  001624  000001                LINE10: .BLKW   1       ;ALL LINES SELECTED
1402  001626  000001                PAR10:  .BLKW   1       ;PARAMETERS
1403  001630  000001                MANT10: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1404
1405  001632  000001                DZCR11: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 11
1406  001634  000001                DZVC11: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 11
1407  001636  000001                LINE11: .BLKW   1       ;ALL LINES SELECTED
1408  001640  000001                PAR11:  .BLKW   1       ;PARAMETERS
1409  001642  000001                MANT11: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1410
1411  001644  000001                DZCR12: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 12
1412  001646  000001                DZVC12: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 12
1413  001650  000001                LINE12: .BLKW   1       ;ALL LINES SELECTED
1414  001652  000001                PAR12:  .BLKW   1       ;PARAMETERS
1415  001654  000001                MANT12: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
```

```
1416
1417  001656  000001          DZCR13: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 13
1418  001660  000001          DZVC13: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 13
1419  001662  000001          LINE13: .BLKW   1       ;ALL LINES SELECTED
1420  001664  000001          PAR13:  .BLKW   1       ;PARAMETERS
1421  001666  000001          MANT13: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1422
1423  001670  000001          DZCR14: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 14
1424  001672  000001          DZVC14: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 14
1425  001674  000001          LINE14: .BLKW   1       ;ALL LINES SELECTED
1426  001676  000001          PAR14:  .BLKW   1       ;PARAMETERS
1427  001700  000001          MANT14: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1428
1429  001702  000001          DZCR15: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 15
1430  001704  000001          DZVC15: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 15
1431  001706  000001          LINE15: .BLKW   1       ;ALL LINES SELECTED
1432  001710  000001          PAR15:  .BLKW   1       ;PARAMETERS
1433  001712  000001          MANT15: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1434
1435  001714  000001          DZCR16: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 16
1436  001716  000001          DZVC16: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 16
1437  001720  000001          LINE16: .BLKW   1       ;ALL LINES SELECTED
1438  001722  000001          PAR16:  .BLKW   1       ;PARAMETERS
1439  001724  000001          MANT16: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1440
1441  001726  000001          DZCR17: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 17
1442  001730  000001          DZVC17: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 17
1443  001732  000001          LINE17: .BLKW   1       ;ALL LINES SELECTED
1444  001734  000001          PAR17:  .BLKW   1       ;PARAMETERS
1445  001736  000001          MANT17: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
1446
1447  001740  177777          DZV.END:        177777
```

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

                    ;:**********************************************************************
                    ;----------------------------------------------------------------------
001742              .TRPTAB:
104400  ADVANCE=TRAP+0          ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
006246          .ADVANCE
104401  SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
004560          .SCOP1
104402  TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
004604          .TYPE
104403  INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
005352          .INSTR
104404  INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
005456          .INSTER
104405  PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
005476          .PARAM
104406  SETFLG=TRAP+6           ;CALL TO  SET FLAG ROUTINE
010110          .SETFLG
104407  SAV05=TRAP+7            ;CALL TO REGISTER SAVE ROUTINE
005676          .SAV05
104410  RES05=TRAP+10           ;CALL TO REGISTER RESTORE ROUTINE
005736          .RES05
104411  CONVRT=TRAP+11          ;CALL TO DATA OUTPUT ROUTINE
005770          .CONVRT
104412  CNVRT=TRAP+12           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
005774          .CNVRT
104413  DEVICE.CLR=TRAP+13              ;CALL TO ISSUE A DEVICE CLEAR
006174          .DEVICE.CLR
104414  DELAY=TRAP+14           ;CALL TO DELAY FOR FAST CPU'S
006226          .DELAY
104415  PARMD=TRAP+15           ;CONVERT DECIMAL STRING TO OCTAL
011156          .PARMD
104416  PAWCH=TRAP+16           ;SET FLAG    ECHO OR  CABLE
010230          .PAWCH
104417  DCLASM=TRAP+17          ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
006214          .DCLASM
104420  SHIFT=TRAP+20           ;CALL TO ROTATE LINE POINTER
006260          .SHIFT
104421  LPRSET=TRAP+21          ;CALL TO SET UP LPR DEVICE REGISTER
006276          .LPRSET
104422  BUFSET=TRAP+22          ;CALL TO ZERO BUFFER AREA
006336          .BUFSET

                    ;----------------------------------------------------------------------
                    ;:**********************************************************************
```

```
1496                                    ;DZV11 VECTOR AND REGISTER INDIRECT POINTERS
1497                                    ;WORKING AREA
1498
1499  002010  160040            DZVCSR: 160040  ;R/W
1500  002012  160041            HDZVCSR:160041  ;R/W
1501  002014  160042            DZVRBUF:160042  ;READ ONLY
1502  002016  160043            HDZVRBUF:160043 ;READ ONLY
1503  002020  160042            DZVLPR: 160042  ;WRITE ONLY
1504  002022  160043            HDZVLPR:160043  ;WRITE ONLY
1505  002024  160044            DZVTCR: 160044  ;R/W
1506  002026  160045            HDZVTCR:160045  ;R/W
1507  002030  160046            DZVMSR: 160046  ;READ ONLY
1508  002032  160047            HDZVMSR:160047  ;READ ONLY
1509  002034  160046            DZVTDR: 160046  ;WRITE ONLY
1510  002036  160047            HDZVTDR:160047  ;WRITE ONLY
1511
1512                                    ;DEFAULT DZV VECTORS
1513
1514  002040  000300            DZVRIV. 300     ;REC INTR VECTOR
1515  002042  000302            DZVRIS: 302     ;REC INTR STATUS
1516  002044  000304            DZVTIV: 304     ;XMIT INTR VECTOR
1517  002046  000306            DZVTIS: 306     ;XMIT INTR STATUS
1518
1519
```

```
1520
1521                                    ;TIME TABLE FOR RELATIVE TIMING TESTS
1522                                    ;-----------------------------------
1523
1524  002050                            TMTBL:
1525  002050  000000                    T50:    0
1526  002052  000000                    T75:    0
1527  002054  000000                    T110:   0
1528  002056  000000                    T134:   0
1529  002060  000000                    T150:   0
1530  002062  000000                    T300:   0
1531  002064  000000                    T600:   0
1532  002066  000000                    T1200:  0
1533  002070  000000                    T1800:  0
1534  002072  000000                    T2000:  0
1535  002074  000000                    T2400:  0
1536  002076  000000                    T3600:  0
1537  002100  000000                    T4800:  0
1538  002102  000000                    T7200:  0
1539  002104  000000                    T9600:  0
1540  002106  000000                    TEIGHT:0
1541  002110  000000                    TSEVEN: 0
1542  002112  000000                    TSIX:   0
1543  002114  000000                    TFIVE:  0
```

```
1544
1545                                    ;PROGRAM INITIALIZATION
1546                                    ;LOCK OUT INTERRUPTS
1547                                    ;SET UP PROCESSOR STACK
1548                                    ;SET UP POWER FAIL VECTOR
1549                                    ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1550                                    ;TYPE TITLE MESSAGE
1551
1552  002116                        .START:
1553  002116  000005                   RESET                   ;CLEAR THE WORLD. START NEW ENVIRONMENT
1554  002120  012706  001120           MOV     #STACK,SP       ;SET UP STACK
1555  002124  106427  000200           MTPS    #MASK           ;LOCK OUT INTERRUPTS
1556  002130  012737  007252  000024   MOV     #$PWRDN,@#24    ;SET UP POWER FAIL VECTOR
1557  002136  012737  006360  000030   MOV     #$ERROR,EMTVEC  ;SET UP ERROR VECTOR
1558  002144  012737  000340  000032   MOV     #340,EMTVEC+2
1559  002152  005037  001126           CLR     $PASS           ;CLEAR PASS COUNT
1560  002156  105037  001247           CLRB    $ERFLG          ;CLEAR ERROR FLAG
1561  002162  012737  001500  001420   MOV     #DZV.MAP,ACTIVE ;GET MAP POINTER.
1562  002170  012737  000001  001412   MOV     #1,RUN          ;POINT POINTER TO FIRST DEVICE.
1563  002176  005037  001256           CLR     $ERTTL          ;CLEAR ERROR COUNT
1564  002202  005037  001262           CLR     $ERRPC          ;CLEAR LAST ERROR POINTER
1565  002206  005037  001246           CLR     $TSTNM          ;SET UP FOR TEST 1
1566  002212  012737  002116  001252   MOV     #.START,$LPADR  ;SET UP FOR POWER FAIL BEFORE
1567                                                            ;TESTING STARTS
1568                                    ;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
1569  002220  012737  000176  001304   MOV     #SWREG,SWR      ;POINT TO SOFTWARE SWR
1570  002226  012737  000174  001306   MOV     #DISPREG,DISPLAY ;POINT TO SOFTWARE DISPLAY REGISTER
1571  002234  105737  001422           TSTB    INIFLG          ;HAVE WE ALREADY BEEN HERE TODAY?
1572  002240  001010                   BNE     10$             ;IF SO, SKIP PRINTING THE TITLE
1573  002242  023727  000042  004250   CMP     @#42,#SENDAD    ;IF RUNNING UNDER ACT
1574  002250  001402                   BEQ     1$              ;DON'T PRINT TITLE
1575  002252  104402  001000           TYPE    ,MTITLE         ;PRINT THE DIAGNOSTIC'S TITLE
1576  002256  105337  001422       1$: DECB    INIFLG          ;SET THE ONCE ONLY FLAG
1577  002262  105737  001141      10$: TSTB    $ENVM           ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
1578  002266  100004                   BPL     15$             ;IF NOT, GO CHECK FOR AUTO-SIZING
1579  002270  004737  011352           JSR     PC,SETAPT       ;OTHERWISE, GO DO APT SIZING FROM ETABLE
1580  002274  000137  003554           JMP     105$            ;GO PRINT DZV STATUS TABLE
1581  002300  032777  000001  176776  15$: BIT    #SW00,@SWR      ;RESELECT ?
1582  002306  001002                   BNE     20$             ;IF YES, GO SET UP THE INFORMATION
1583  002310  000137  002612           JMP     55$             ;IF NO, SKIP THE INTERROGATION
1584  002314  012700  001500      20$: MOV     #DZV.MAP,R0     ;POINT TO THE BEGINNING OF THE MAP TABLE
1585  002320  105037  001423           CLRB    HDRFLG          ;MAKE SURE A MAP GETS PRINTED
1586  002324  005020              25$: CLR     (R0)+           ;CLEAR A TABLE LOCATION
1587  002326  020027  001740           CMP     R0,#DZV.END     ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
1588  002332  001374                   BNE     25$             ;IF NOT ,CLEAR THE NEXT LOCATION IN THE TABLE
1589  002334  105337  00142c           DECB    INIFLG          ;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!
1590
1591                                    ;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
1592                                    ;TABLE AND SET UP THE DIAGNOSTIC.
1593
1594                                            ;GET THE BASE ADDRESS OF THE DZV11'S
1595
1596  002340  104403                           INSTR                   ;CALL THE STRING INPUT ROUTINE
1597  002342  003032                           91$                     ;POINTER TO MESSAGE TO BE PRINTED
1598  002344  104405                           PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1599  002346  160000                           160000                  ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
```

```
1600  002350  163770                          163770                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1601  002352  001500                          DZCR0                   ;POINTER TO MAP LOCATION TO BE FILLED
1602  002354     007                          .BYTE   7               ;MASK OF INVALID BITS FOR THIS PARAMETER
1603  002355     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1604  002356  013737  001500  001174          MOV     DZCR0,$BASE     ;COPY BASE ADDRESS TO ETABLE
1605
1606                                          ;GET THE BASE VECTOR ADDRESS
1607
1608  002364  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
1609  002366  003076                          92$                     ;POINTER TO MESSAGE TO BE PRINTED
1610  002370  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1611  002372  000300                          300                     ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1612  002374  000776                          776                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1613  002376  001502                          DZVC0                   ;POINTER TO MAP LOCATION TO BE FILLED
1614  002400     003                          .BYTE   3               ;MASK OF INVALID BITS FOR THIS PARAMETER
1615  002401     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1616  002402  013737  001502  001170          MOV     DZVC0,$VECT1    ;COPY VECTOR TO ETABLE
1617                                          ;GET THE MODE OF OPERATION (E,I,S)
1618
1619  002410  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
1620  002412  003325                          96$                     ;POINTER TO THE MESSAGE TO BE PRINTED
1621  002414  104406                          SETFLG                  ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
1622  002416  001510                          MANT0                   ;THIS IS THE FLAG BEING SETUP
1623
1624                                          ;GET THE NUMBER OF DZV11'S RUNNING
1625
1626  002420  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
1627  002422  003262                          95$                     ;POINTER TO MESSAGE TO BE PRINTED
1628  002424  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1629  002426  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1630  002430  000020                          16.                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1631  002432  001344                          $TMP1                   ;POINTER TO MAP LOCATION TO BE FILLED
1632  002434     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
1633  002435     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1634
1635  002436  012737  000017  001504          MOV     #17,LINE0       ;SET UP DEFAULT LINES
1636  002444  012737  017470  001506          MOV     #17470,PAR0     ;SET UP DEFAULT LPR PARAMETER
1637                                          ;RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
1638  002452  032777  000010  176624          BIT     #SW03,@SWR      ;DO YOU WANT PARAMETERS?
1639  002460  001402                          BEQ     30$             ;IF NO, SKIP THE PARAMETER CALL
1640  002462  004737  002642                  JSR     PC,65$          ;GET PARAMETERS
1641  002466  012737  000001  001410  30$:     MOV     #1,SAVACTV      ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
1642  002474  113737  001344  001414          MOVB    $TMP1,DZVNUM    ;COPY THE NUMBER OF DEVICES
1643  002502  005337  001344          35$:     DEC     $TMP1           ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
1644  002506  001404                          BEQ     40$             ;  SELECTED DEVICES
1645  002510  000261                          SEC                     ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
1646  002512  006137  001410                  ROL     SAVACTV         ;POINT TO THE NEXT DEVICE
1647  002516  000771                          BR      35$             ;GO DO THIS PROCEDURE AGAIN
1648  002520  013737  001410  001346  40$:     MOV     SAVACTV,$TMP2   ;# OF TIMES
1649  002526  012700  001500                  MOV     #DZCR0,R0       ;SET A POINTER TO THE SPECIFIED INFORMATION
1650  002532  012701  001512                  MOV     #DZCR1,R1       ;POINT R1 TO THE REST OF THE MAP TABLE
1651  002536  012702  001204                  MOV     #$DDW0,R2       ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
1652  002542  000241                          CLC                     ;INITIALIZE THE 'C' BIT FOR A ROTATION
1653  002544  006037  001346                  ROR     $TMP2           ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
1654  002550  006237  001346          45$:     ASR     $TMP2           ;ISOLATE A SELECTION FLAG IN THE 'C' BIT
1655  002554  103404                          BCS     50$             ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
```

```
1656  002556  012711  177777                    MOV     #-1,(R1)        ;TERMINATE THE LIST
1657  002562  000137  003530                    JMP     100$            ;GO TO THE NEXT BLOCK
1658  002566  012011                    50$:    MOV     (R0)+,(R1)      ;ADDRESS
1659  002570  062721  000010                    ADD     #10,(R1)+       ;POINT TO THE NEXT DZV11 ADDRESS VALUE
1660  002574  012011                            MOV     (R0)+,(R1)      ;VECTOR
1661  002576  062721  000010                    ADD     #10,(R1)+       ;POINT TO THE NEXT VECTOR VALUE
1662  002602  012021                            MOV     (R0)+,(R1)+     ;LINES
1663  002604  012021                            MOV     (R0)+,(R1)+     ;PARAMETERS
1664  002606  012021                            MOV     (R0)+,(R1)+     ;MAINTENANCE MODE
1665  002610  000757                            BR      45$
1666  002612  032777  000010  176464    55$:    BIT     #SW03,@SWR      ;ASK PARAMETERS ?
1667  002620  001002                            BNE     60$             ;IF NO, GO DO AUTO SIZING
1668  002622  000137  003530                    JMP     100$            ;GO SET UP FOR AUTO SIZING
1669  002626  004737  002642            60$:    JSR     PC,65$          ;GO ASK PARAMETERS
1670  002632  105337  001422                    DECB    INIFLG          ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
1671  002636  000137  003554                    JMP     105$            ;GO TO THE NEXT BLOCK
1672
1673                                            ;GET THE ACTIVE LINES PARAMETER
1674
1675  002642                            65$:
1676  002642  104403                            INSTR                   ;CALL THE STRING INPUT ROUTINE
1677  002644  003137                            93$                     ;POINTER TO MESSAGE TO BE PRINTED
1678  002646  104405                            PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1679  002650  000001                            1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1680  002652  000017                            17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1681  002654  001504                            LINE0                   ;POINTER TO MAP LOCATION TO BE FILLED
1682  002656     360                            .BYTE   360             ;MASK OF INVALID BITS FOR THIS PARAMETER
1683  002657     001                            .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1684  002660  105037  001423                    CLRB    HDRFLG          ;MAKE SURE THE CHANGES ARE PRINTED
1685
1686                                            ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
1687                                            ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
1688
1689  002664  005737  001510                            TST     MANT0           ;IS STAGGERED THE MODE OF OPERATION?
1690  002670  100021                                    BPL     85$             ;IF NOT, SKIP THIS SEGMENT
1691  002672  013703  001504                            MOV     LINE0,R3        ;GET A SCRATCH COPY OF THE ACTIVE LINES
1692  002676  006003                    70$:    ROR     R3              ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
1693  002700  103410                            BCS     80$             ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
1694  002702  001414                            BEQ     85$             ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
1695  002704  006203                            ASR     R3              ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
1696  002706  103373                            BCC     70$             ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
1697  002710  104402  001356            75$:    TYPE    ,$QUES          ;THIS IS AN INCORRECT PARAMETER
1698  002714  104402  010034                    TYPE    ,MBADLN         ;LET THE USER KNOW ABOUT IT
1699  002720  000750                            BR      65$             ;GO GET THE CORRECT PARAMETER
1700  002722  001772                    80$:    BEQ     75$             ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
1701  002724  006203                            ASR     R3              ;GET THE NEXT FLAG
1702  002726  103370                            BCC     75$             ;IF IT ISN'T SET, THERE'S AN ERROR
1703  002730  000241                            CLC                     ;INITIALIZE THE 'C' BIT FOR TESTING OF THE NEXT PAIR
1704  002732  000761                            BR      70$             ;GO TEST THE NEXT PAIR OF FLAGS
1705
1706                                            ;GET THE LINE PARAMETER REGISTER ARGUMENT
1707
1708  002734                            85$:
1709  002734  104403                            INSTR                   ;CALL THE STRING INPUT ROUTINE
1710  002736  003212                            94$                     ;POINTER TO MESSAGE TO BE PRINTED
1711  002740  104405                            PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
```

```
1712  002742  000000                          0                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1713  002744  000017                          17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1714  002746  001506                          PAR0                    ;POINTER TO MAP LOCATION TO BE FILLED
1715  002750     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
1716  002751     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1717  002752  012702  001504                  MOV     #LINE0,R2       ;POINT TO THE LINE SELECTION PARAMETER
1718  002756  012703  001506                  MOV     #PAR0,R3        ;POINT TO THE CHOSEN PARAMETERS
1719  002762  011304                          MOV     (R3),R4         ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
1720  002764  006304                          ASL     R4              ;ALIGN INDEX ON WORD BOUNDARY
1721  002766  016437  017374  006244          MOV     DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
1722  002774  000313                          SWAB    (R3)            ;PLACE IN HIGH BYTE
1723  002776  052713  010070                  BIS     #10070,(R3)     ;PLACE EXTRA PARAMETERS INTO LOC
1724  003002  011262  000012          90$:    MOV     (R2),12(R2)     ;LOAD THE LINES
1725  003006  011363  000012                  MOV     (R3),12(R3)     ;LOAD THE PARAMETERS
1726  003012  062702  000012                  ADD     #12,R2          ;POINT TO THE NEXT SET
1727  003016  062703  000012                  ADD     #12,R3          ;   .. OF BOTH PARAMETERS
1728  003022  020327  001734                  CMP     R3,#PAR17       ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
1729  003026  001365                          BNE     90$             ;IF NOT, GO LOAD SOME MORE PARAMETERS
1730  003030  000207                          RTS     PC              ;RETURN TO CALLING BLOCK
1731  003032  030600  052123  041440  91$:    .ASCIZ  <200>/1ST CSR ADDRESS (160000:163770):  /
  (1) 003076  030600  052123  053040  92$:    .ASCIZ  <200>/1ST VECTOR ADDRESS (300:770):  /
  (1) 003137     200  044514  042516  93$:    .ASCIZ  <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:17):  /
  (1) 003212  042200  043105  052501  94$:    .ASCIZ  <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17):  /
  (1) 003262  021600  047440  020106  95$:    .ASCIZ  <200>/# OF DZV11'S <IN OCTAL> (1:20):  /
  (1) 003325     200  040515  047111  96$:    .ASCII  <200>/MAINTENANCE MODE/
  (1) 003346  020200  042533  052130          .ASCII  <200>/ [EXTERNAL  <H325>     (E)]/
  (1) 003402  020200  044533  052116          .ASCII  <200>/ [INTERNAL  <DZVCSR03=1>(I)]/
  (1) 003437     200  055440  052123          .ASCIZ  <200>/ [STAGGERED <H329>     (S)]: /
  (1) 003476  042600  052116  051105  97$:    .ASCIZ  <200>/ENTER DELAY PARAMETER: /
  (1)         003530                          .EVEN
  (1) 003530                          100$:
1732  003530  122737  000377  001422          CMPB    #377,INIFLG     ;ONLY DO AUTO SIZE ON 1ST START
1733  003536  001006                          BNE     105$            ;
1734  003540  032777  000200  175536          BIT     #BIT7,@SWR      ;BIT7=1??
1735  003546  001002                          BNE     105$            ;BR IF NO AUTO SIZE
1736  003550  004737  011500                  JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
1737  003554  105737  001423          105$:   TSTB    HDRFLG          ;HAS THE TABLE BEEN TYPED YET?
1738  003560  001021                          BNE     120$            ;IF SO, DON'T TYPE IT AGAIN
1739  003562  105337  001423                  DECB    HDRFLG          ;INDICATE THAT THE TABLE WILL BE TYPED
1740  003566  104402  010006                  TYPE    ,XHEAD          ;TYPE MAP HEADER
1741  003572  012700  001500                  MOV     #DZV.MAP,R0     ;SET POINTER
1742  003576  010037  001344          110$:   MOV     R0,$TMP1        ;POINT TO THE MAP LOCATION
1743  003602  012037  001346                  MOV     (R0)+,$TMP2     ;SET DATA
1744  003606  022737  177777  001346          CMP     #-1,$TMP2       ;END OF LIST?
1745  003614  001403                          BEQ     120$            ;BR IF YES
1746  003616  104411                  115$:   CONVRT                  ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
1747  003620  010076                          XSTATQ                  ;CONVERT THE DATA AT THIS ADDRESS
1748  003622  000765                          BR      110$            ;GO PRINT THE NEXT PARAMETER
1749  003624  013737  001410  001406  120$:   MOV     SAVACTV,DZVACTV ;COPY BIT MAP OF SYSTEM DEVICES ACTIVE
1750  003632  113737  001414  001416          MOVB    DZVNUM,SAVNO    ;COPY NO. OF SYSTEM DEVICES ACTIVE
1751  003640  032777  000100  175436          BIT     #SW06,@SWR      ;DESELECT SPECIFIC DEVICES??
1752  003646  001431                          BEQ     135$            ;BR IF NO.
1753  003650                          121$:
1754  003650  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
1755  003652  007724                          MNEW                    ;POINTER TO MESSAGE TO BE PRINTED
1756  003654  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
```

```
1757  003656  000001                                  1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1758  003660  177777                                  177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1759  003662  001406                                  DZVACTV                 ;POINTER TO MAP LOCATION TO BE FILLED
1760  003664     000                                  .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
1761  003665     001                                  .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1762  003666  023737  001406  001410                  CMP     DZVACTV,SAVACTV ;IS THE VALUE VALID?
1763  003674  101403                                  BLOS    122$            ;BRANCH IF YES
1764  003676  104402  007576                         TYPE    .MERR3          ;IF NOT THEN TYPE ERROR
1765  003702  000762                                  BR      121$            ;GO REASK QUESTION
1766  003704  105037  001416          122$:           CLRB    SAVND           ;CLEAR NO. OF DEVICES BEING TESTED
1767  003710  013737  001406  001344                  MOV     DZVACTV,TMP1    ;COPY BIT MAP OF ACTIVE DEVICES BEING TESTED
1768  003716  006237  001344          126$:           ASR     TMP1            ;SHIFT OUT AN ACTIVE BIT
1769  003722  103002                                  BCC     127$            ;IF NOT ACTIVE SKIP INCREMENT
1770  003724  105237  001416                          INCB    SAVND           ;IF ACTIVE RECORD IT
1771  003730  001372          127$:           BNE     126$            ;IF ALL ACTIVE BITS RECORDED DON'T BRANCH
1772  003732  032777  000020  175344  135$:           BIT     #SW04,@SWR      ;CHECK TO SEE IF DELAY COUNT CHANGES
1773  003740  001407                                  BEQ     140$            ;IF NOT, GO CLEAR VECTOR AREA
1774  003742  104403                                  INSTR                   ;CALL THE STRING INPUT ROUTINE
1775  003744  003476                                  97$                     ;POINTER TO MESSAGE TO BE PRINTED
1776  003746  104405                                  PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
1777  003750  000001                                  1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1778  003752  177777                                  177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
1779  003754  006244                                  DLYCNT                  ;POINTER TO MAP LOCATION TO BE FILLED
1780  003756     000                                  .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
1781  003757     001                                  .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
1782  003760  012700  000300          140$:           MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
1783  003764  012701  000302                          MOV     #302,R1         ;VECTOR AREA. 300-776
1784  003770  010120          145$:           MOV     R1,(R0)+        ;START PUTTING 'PC+2 - HALT'
1785  003772  005021                          CLR     (R1)+           ;IN VECTOR AREA.
1786  003774  022021                          CMP     (R0)+,(R1)+     ;POP POINTERS
1787  003776  022700  001000                  CMP     #1000,R0        ;ALL DONE??
1788  004002  001372                          BNE     145$            ;BR IF NO.
1789
1790                                  .TEST START AND RESTART
1791                                  .----------------------
1792
1793  004004  012706  001120          .BEGIN: MOV     #STACK,SP       ;SET UP STACK
1794  004010  106427  000200                  MTPS    #PASK           ;LOCK OUT INTERRUPTS
1795  004014  005737  000042                  TST     @#42            ;IS PROGRAM UNDER MONITOR CONTROL
1796  004020  001015                          BNE     2$              ;BR IF YES
1797  004022  032777  000004  175254          BIT     #BIT2,@SWR      ;CHECK FOR LOCK ON TEST
1798  004030  001406                          BEQ     1$              ;BR IF NO LOCK DESIRED.
1799  004032  104402  007622                  TYPE    .MLOCK          ;TYPE LOCK SELECTED.
1800  004036  012737  000240  004326          MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
1801  004044  000403                          BR      2$              ;CONTINUE ALONG.
1802  004046  013737  004554  004326  1$      MOV     BPW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
1803  004054  012737  010452  001252  2$      MOV     #CYCLE,SLPADR   ;START AT 'CYCLE' FIND WHICH DEVICE TO TEST
1804  004062  113737  001416  001415          MOVB    SAVND,SAVNUM    ;COPY ACTIVE DEVICES BEING TESTED
1805  004070  104402  007513                  TYPE    .MR             ;TYPE 'RUNNING'
1806  004074  000177  175152                  JMP     @SLPADR         ;START TESTING
```

```
                        ;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

004100          $EOP:
                        SCOPE
000004
004100  005037  001262          CLR     $ERRPC          ;CLEAR LAST ERROR PC
004102  105037  001247          CLRB    $ERFLG          ;CLEAR ERROR FLAG
004106  104402  007467          TYPE    ,MEPASS         ;TYPE END PASS
004112  104402  007651          TYPE    ,MCSRX          ;TYPE CSR
004116  104412  004264          CNVRT   ,XCSR           ;SHOW IT
004122  104402  007657          TYPE    ,MVECX          ;TYPE VECTOR
004126  104412  004272          CNVRT   ,XVEC           ;SHOW IT
004132  005237  001126          INC     $PASS           ;RAISE PASS COUNT
004136  104402  007665          TYPE    ,MPASSX         ;TYPE PASSES
004142  104412  004300          CNVRT   ,XPASS          ;SHOW IT
004146  005337  001126          DEC     $PASS           ;RESTORE PASS COUNT
004152  104402  007676          TYPE    ,MERRX          ;TYPE ERRORS
004156  104412  004306          CNVRT   ,XERR           ;SHOW IT
004162  005237  001130          INC     $DEVCT          ;INC DEVCNT FOR APT
004166  105337  001415          DECB    SAVNUM          ;ARE ALL DEVICES TESTED?
004172  001030                  BNE     $DOAGN          ;BR IF NO.
004176  113737  001416  001415  MOVB    SAVND,SAVNUM    ;RESTORE THE COUNT
004200  005037  001354          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
004206  005237  001126          INC     $PASS           ;;INCREMENT THE PASS NUMBER
004212  042737  100000  001126  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004220  005327                  DEC     (PC)+           ;;LOOP?
004222  000001          $EOPCT: .WORD   1
004224  003013                  BGT     $DOAGN          ;;YES
004226  012737                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004230  000001          $ENDCT: .WORD   1
004232  004226                  $EOPCT
004234  013700  000042  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
004240  001405                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004242  000005                  RESET                   ;;CLEAR THE WORLD
004244  004710          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004246  000240                  NOP                     ;;SAVE ROOM
004250  000240                  NOP                     ;;FOR
004252  000240                  NOP                     ;;ACT11
004254          $DOAGN:
004254  000137                  JMP     @(PC)+          ;;RETURN
004256  010452          $RTNAD: .WORD   CYCLE
004260
004262  000001          YCSR:   1
004264     006     002          .BYTE   6,2
004266  002010                  DZVCSR
004270  000001          XVEC:   1
004272     003     002          .BYTE   3,2
```

```
1863  004276  002040                        DZVRIV
1864  004300  000001                xPASS:  1
1865  004302     006      002               .BYTE   6,2
1866  004304  001126                        $PASS
1867  004306  000001                xERR:   1
1868  004310     006      002               .BYTE   6,2
1869  004312  001256                        $ERTTL
1870
1871                                        .SCOPE LOOP AND ITERATION HANDLER
1872                                        ;-------------------------------
1873
1874                                .SBTTL  SCOPE HANDLER ROUTINE
1875
1876                                ;;*****************************************************************
1877                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1878                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1879                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1880                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1881                                ;*SW14=1         LOOP ON TEST
1882                                ;*SW11=1         INHIBIT ITERATIONS
1883                                ;*CALL
1884                                ;*      SCOPE            ;;SCOPE=IOT
1885
1886  004314                        $SCOPE:
1887  004314  005037  001262        .SCOPE: CLR     $ERRPC          ;CLEAR LAST ERROR PC.
1888  004320  022716  012206                CMP     #TST1+2,(SP)    ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
1889  004324  001413                        BEQ     $XTSTR          ;IF SO, DON'T LOOP ON IT
1890  004326  000406                TTST:   BR      1$              ;GOTO 1$    (IF LOCK SW02=1; THIS LOC =240)
1891  004330  105777  174754                TSTB    @$TKS           ;KEYBOARD DONE?
1892  004334  100067                        BPL     $OVER           ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
1893  004336  017766  174750  177776        MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
1894  004344  032777  040000  174732  1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
1895  004352  001060                        BNE     $OVER           ;;YES IF SW14=1
1896                                ;######START OF CODE FOR THE XOR TESTER######
1897  004354  000416                $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1898                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
1899  004356  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1900  004362  012737  004402  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
1901  004370  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
1902  004374  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
1903  004400  000436                        BR      $SVLAD          ;;GO TO THE NEXT TEST
1904  004402  022626                5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
1905  004404  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
1906  004410  000441                        BR      $OVER           ;;LOOP ON THE PRESENT TEST
1907  004412                        6$:,######END OF CODE FOR THE XOR TESTER######
1908  004412  105737  001247        2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
1909  004416  001404                        BEQ     3$              ;;BR IF NO
1910  004420  105037  001247        4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
1911  004424  005037  001354                CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1912  004430  032777  004000  174646  3$:   BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
1913  004436  001011                        BNE     1$              ;;BR IF YES
1914  004440  005737  001126                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
1915  004444  001406                        BEQ     1$              ;;      INHIBIT ITERATIONS
1916  004446  005237  001250                INC     $ICNT           ;;INCREMENT ITERATION COUNT
1917  004452  023737  001354  001250        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
1918  004460  002015                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
```

```
1919  004462  012737  000001  001250  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
1920  004470  013737  004556  001354          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
1921  004476  105237  001246          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
1922  004502  113737  001246  001124          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
1923  004510  011637  001252                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
1924  004514  013777  001246  174564  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
1925  004522  013716  001252                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
1926  004526  004737  007006                  JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
1927  004532  105037  001424                  CLRB    MNTFLG          ;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
1928  004536  005737  001372                  TST     MODF            ;HAS THE MODE BEEN CHANGED?
1929  004542  001003                          BNE     4$              ;IF NOT INTERNAL , GO DO A TEST
1930  004544  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
1931  004552  000002          4$:     RTI             ;GO DO THE TEST
1932  004554  000406          BRW:    406
1933  004556  000005          $MXCNT: 5               ;;MAX. NUMBER OF ITERATIONS
1934
1935                                  ;CHECK FOR FREEZE ON CURRENT DATA
1936                                  ;-------------------------------
1937
1938  004560  032777  001000  174516  .SCOP1: BIT     #SW09,@SWR      ;IS SW09=1(SET)?
1939  004566  001405                  BEQ     1$              ;BR IF NOT SET.
1940  004570  005737  001364                  TST     LOCK            ;IS THERE A TIGHT LOOP SPECIFIED?
1941  004574  001402                  BEQ     1$              ;IF NO, RETURN
1942  004576  013716  001364                  MOV     LOCK,(SP)       ;IF YES, GOTO THE ADDRESS IN LOCK.
1943  004602  000002          1$:     RTI                     ;GO BACK.
1944
1945  004604  032777  010000  174472  .TYPE:  BIT     #SW12,@SWR      ;INHIBIT ALL PRINTOUT??
1946  004612  001403                  BEQ     $TYPE           ;IF NOT, GO TYPE
1947  004614  062716  000002                  ADD     #2,(SP)         ;SKIP OVER MESSAGE POINTER
1948  004620  000002                  RTI                     ;RETURN TO WHERE PROCEDURE WAS INVOKED
1949                                  .SBTTL  TYPE ROUTINE
1950
1951                                  ;;**************************************************************
1952                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1953                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1954                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1955                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1956                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1957                                  ;*
1958                                  ;*CALL:
1959                                  ;*1) USING A TRAP INSTRUCTION
1960                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1961                                  ;*OR
1962                                  ;*      TYPE
1963                                  ;*      MESADR
1964                                  ;*
1965
1966  004622  105737  001323          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
1967  004626  100002                  BPL     1$              ;;BR IF YES
1968  004630  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
1969  004632  000430                  BR      3$              ;;LEAVE
1970  004634  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
1971  004636  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
1972  004642  122737  000001  001140          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
1973  004650  001011                  BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
1974  004652  132737  000100  001141          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
1975  004660  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
1976  004662  010037  004672                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
1977  004666  004737  005112                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
1978  004672  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
1979  004674  132737  000040  001141  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
1980  004702  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
1981  004704  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1982  004706  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
1983  004710  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
1984  004712  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
1985  004714  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
1986  004720  000002                          RTI                     ;;RETURN
1987  004722  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
1988  004726  001430                          BEQ     8$
1989  004730  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
1990  004734  001006                          BNE     5$
1991  004736  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
1992  004740  104402                          TYPE                    ;;TYPE A CR AND LF
1993  004742  001357                          $CRLF
1994  004744  105037  005100                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
1995  004750  000755                          BR      2$              ;;GET NEXT CHARACTER
1996  004752  004737  005034          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
1997  004756  123726  001322          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1998  004762  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
1999  004764  013746  001320                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2000                                                                  ;;AND THE NULL CHAR.
2001  004770  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2002  004774  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2003  004776  004737  005034                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2004  005002  105337  005100                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2005  005006  000770                          BR      7$              ;;LOOP
2006
2007                                          ;HORIZONTAL TAB PROCESSOR
2008
2009  005010  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2010  005014  004737  005034          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2011  005020  132737  000007  005100          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2012  005026  001372                          BNE     9$              ;;TAB STOP
2013  005030  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2014  005032  000724                          BR      2$              ;;GET NEXT CHARACTER
2015  005034  105777  174254          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2016  005040  100375                          BPL     $TYPEC
2017  005042  116677  000002  174246          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2018  005050  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2019  005056  001003                          BNE     1$              ;;BRANCH IF NO
2020  005060  105037  005100                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2021  005064  000406                          BR      $TYPEX          ;;EXIT
2022  005066  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2023  005074  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2024  005076  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2025  005100  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
2026  005102  000207          $TYPEX: RTS     PC
2027
2028                                          .SBTTL  APT COMMUNICATIONS ROUTINE
2029
2030                                          ;;**********************************************************************
```

```
2031  005104  112737  000001  005350  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2032  005112  112737  000001  005346  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2033  005120  000403                  BR      $ATYC
2034  005122  112737  000001  005350  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2035  005130                          $ATYC:
2036  005130  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2037  005132  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2038  005134  105737  005346          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2039  005140  001450                  BEQ     5$              ;;IF NOT:  BR
2040  005142  122737  000001  001140  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2041  005150  001031                  BNE     3$              ;;IF NOT:  BR
2042  005152  132737  000100  001141  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2043  005160  001425                  BEQ     3$              ;;IF NOT:  BR
2044  005162  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2045  005166  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2046  005174  005737  001120  1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2047  005200  001375                  BNE     1$              ;;IF NOT:  WAIT
2048  005202  010037  001134          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2049  005206  105720          2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2050  005210  001376                  BNE     2$
2051  005212  163700  001134          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2052  005216  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2053  005220  010037  001136          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2054  005224  012737  000004  001120  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2055  005232  000413                  BR      5$
2056  005234  017637  000004  005260  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2057  005242  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2058  005250  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2059  005254  004737  004622          JSR     PC,$TYPE        ;;CALL TYPE MACRO
2060  005260  000000          4$:     .WORD   0
2061  005262                  5$:
2062  005262  105737  005350  10$:     TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2063  005266  001416                  BEQ     12$             ;;IF NOT:  BR
2064  005270  005737  001140          TST     $ENV            ;;RUNNING UNDER APT?
2065  005274  001413                  BEQ     12$             ;;IF NOT:  BR
2066  005276  005737  001120  11$:     TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2067  005302  001375                  BNE     11$             ;;IF NOT:  WAIT
2068  005304  017637  000004  001122  MOV     @4(SP),$FATAL   ;;GET ERROR #
2069  005312  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2070  005320  005237  001120          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2071  005324  105037  005350  12$:     CLRB    $FFLG           ;;CLEAR FATAL FLAG
2072  005330  105037  005347          CLRB    $LFLG           ;;CLEAR LOG FLAG
2073  005334  105037  005346          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2074  005340  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
2075  005342  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
2076  005344  000207                  RTS     PC              ;;RETURN
2077  005346     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
2078  005347     000          $LFLG:  .BYTE   0               ;;LOG FLAG
2079  005350     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
2080        005352                  .EVEN
2081        000200          APTSIZE=200
2082        000001          APTENV=001
2083        000100          APTSPOOL=100
2084        000040          APTCSUP=040
2085
2086                        ;STRING INPUT ROUTINE
```

```
2087                                        ;-------------------------
2088
2089  005352  010346                .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
2090  005354  010446                        MOV     R4,-(SP)        ;SAVE R4 ON STACK
2091  005356  017637  000004 005374        MOV     @4(SP),,MSG     ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
2092  005364  062766  000002 000004        ADD     #2,4(SP)        ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
2093  005372  104402                .INST1: TYPE                    ;PRINT THE MESSAGE
2094  005374  000000                .MSG:   0                       ;MESSAGE IS POINTED TO FROM HERE
2095  005376  012704  010304                MOV     #INBUF,R4       ;POINT R4 TO THE INPUT BUFFER
2096  005402  012703  000007                MOV     #7,R3           ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
2097  005406  105777  173676        1$:     TSTB    @$TKS           ;HAS A CHARACTER BEEN RECEIVED?
2098  005412  100375                        BPL     1$              ;IF NO, KEEP WAITING FOR IT
2099  005414  117714  173672                MOVB    @$TKB,(R4)      ;IF YES, SAVE IT IN THE INPUT BUFFER
2100  005420  142714  000200                BICB    #200,(R4)       ;KEEP ONLY THE 7-BIT ASCII INFORMATION
2101  005424  122427  000015                CMPB    (R4)+,#15       ;IS THIS CHARACTER A LINE FEED?
2103  005432  105777  173656        2$:     TSTB    @$TPS           ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
2104  005436  100375                        BPL     2$              ;IF WE CAN'T, WAIT UNTIL WE CAN
2105  005440  017777  173646 173650        MOV     @$TKB,@$TPB     ;ECHO THE CHARACTER BACK
2106  005446  005303                        DEC     R3              ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
2107  005450  001356                        BNE     1$              ;IF WE DON'T HAVE 7, GO GET SOME MORE
2108  005452  012604                        MOV     (SP)+,R4        ;IF WE HAVE 7, RESTORE R4
2109  005454  012603                        MOV     (SP)+,R3        ;RESTORE R3
2110  005456  010346                .INSTE: MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
2111  005460  010446                        MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
2112  005462  104402  001356                TYPE    ,$QUES          ;PRINT A QUESTION MARK... WHAT'S GOING ON?
2113  005466  000741                        BR      .INST1          ;GO PRINT THE MESSAGE AGAIN
2114  005470  012604                INSTR2: MOV     (SP)+,R4        ;RESTORE R4
2115  005472  012603                        MOV     (SP)+,R3        ;RESTORE R3
2116  005474  000002                        RTI                     ;RETURN TO THE MAIN PROCEDURE
2117
2118                                        ;CONVERT ASCII STRING TO OCTAL
2119                                        ;-------------------------
2120
2121  005476  010546                .PARAM: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
2122  005500  010446                        MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
2123  005502  016605  000004                MOV     4(SP),R5        ;GET THE SETUP INFORMATION POINTER
2124  005506  012537  005666                MOV     (R5)+,LOLIM     ;SET THE LOW LIMIT FOR THE INPUT
2125  005512  012537  005670                MOV     (R5)+,HILIM     ;SET THE HIGH LIMIT FOR THE INPUT
2126  005516  012537  005672                MOV     (R5)+,DEVADR    ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
2127  005522  112537  005674                MOVB    (R5)+,LOBITS    ;GET THE MASK OF THE INCORRECT BITS
2128  005526  112537  005675                MOVB    (R5)+,ADRCNT    ;GET THE COUNT OF ITEMS TO BE STORED
2129  005532  010566  000004                MOV     R5,4(SP)        ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
2130  005536  005005                PARAM1: CLR     R5              ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
2131  005540  012704  010304                MOV     #INBUF,R4       ;POINT TO THE INPUT BUFFER
2132  005544  122714  000015                CMPB    #15,(R4)        ;IS THIS CHARACTER A CARRIAGE RETURN?
2133  005550  001420                        BEQ     PARERR          ;IF SO, PRINT THE MESSAGE AGAIN
2134  005552  121427  000060        1$:     CMPB    (R4),#60        ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
2135  005556  002415                        BLT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
2136  005560  121427  000067                CMPB    (R4),#67        ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
2137  005564  003012                        BGT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
2138  005566  142714  000060                BICB    #60,(R4)        ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
2139  005572  152405                        BISB    (R4)+,R5        ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
2140  005574  122714  000015                CMPB    #15,(R4)        ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
2141  005600  001406                        BEQ     LIMITS          ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
2142  005602  006305                        ASL     R5              ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
```

```
2143  005604  006305                  ASL     R5              ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
2144  005606  006305                  ASL     R5              ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
2145                                                          ;NEXT THREE BITS
2146  005610  000760                  BR      1$              ;GO GET THE NEXT CHARACTER
2147  005612  104404          PARERR: INSTER                  ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
2148  005614  000750                  BR      PARAM1          ;TRY GETTING THE PARAMETERS AGAIN
2149
2150                                  ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
2151                                  ;--------------------------------------
2152
2153  005616  020537  005670  LIMITS: CMP     R5,HILIM        ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
2154  005622  101373                  BHI     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
2155  005624  020537  005666          CMP     R5,LOLIM        ;IS THE RESULT LOWER THAN ALLOWED?
2156  005630  103770                  BLO     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
2157  005632  133705  005674          BITB    LOBITS,R5       ;ARE ANY INCORRECT BITS SET IN THE RESULT?
2158  005636  001365                  BNE     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
2159
2160                                  ;STORE NUMBER AT SPECIFIED ADDRESS
2161
2162  005640  013704  005672          MOV     DEVADR,R4       ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
2163  005644  010524          1$:     MOV     R5,(R4)+        ;STORE THE RESULT
2164  005646  062705  000002          ADD     #2,R5           ;CALCULATE THE NEXT DATUM
2165  005652  105337  005675          DECB    ADRCNT          ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
2166  005656  001372                  BNE     1$              ;IF NOT, GO STORE THE NEXT DATUM
2167  005660  012604                  MOV     (SP)+,R4        ;RESTORE R4
2168  005662  012605                  MOV     (SP)+,R5        ;RESTORE R5
2169  005664  000002                  RTI                     ;RETURN TO THE MAIN PROGRAM
2170
2171  005666  000000          LOLIM:  0                       ;LOWEST ACCEPTABLE VALUE
2172  005670  000000          HILIM:  0                       ;HIGHEST ACCEPTABLE
2173  005672  000000          DEVADR: 0                       ;LOCATION WHERE RESULT WILL BE STORED
2174  005674     000          LOBITS: .BYTE   0               ;INCORRECT BITS MASK
2175  005675     000          ADRCNT: .BYTE   0               ;COUNT OF ITEMS TO BE STORED
2176
2177                                  ;SAVE PC OF TEST THAT FAILED AND R0-R5
2178                                  ;-------------------------------------
2179
2180  005676  016637  000004  001404  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
2181
2182                                  ;SAVE R0-R5
2183
2184  005704  010537  001340          SV05:   MOV     R5,$REG5        ;SAVE R5
2185  005710  010437  001336                  MOV     R4,$REG4        ;SAVE R4
2186  005714  010337  001334                  MOV     R3,$REG3        ;SAVE R3
2187  005720  010237  001332                  MOV     R2,$REG2        ;SAVE R2
2188  005724  010137  001330                  MOV     R1,$REG1        ;SAVE R1
2189  005730  010037  001326                  MOV     R0,$REG0        ;SAVE R0
2190  005734  000002                  RTI                     ;LEAVE.
2191
2192                                  ;RESTORE R0-R5
2193
2194  005736  013700  001326          .RES05: MOV     $REG0,R0        ;RESTORE R0
2195  005742  013701  001330                  MOV     $REG1,R1        ;RESTORE R1
2196  005746  013702  001332                  MOV     $REG2,R2        ;RESTORE R2
2197  005752  013703  001334                  MOV     $REG3,R3        ;RESTORE R3
2198  005756  013704  001336                  MOV     $REG4,R4        ;RESTORE R4
```

```
2199  005762  013705  001340          MOV     $REGS,R5        ;RESTORE R5
2200  005766  000002                  RTI                     ;LEAVE
2201
2202                                  ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
2203                                  ;-----------------------------------------------------
2204
2205  005770  104402  001357  .CONVR: TYPE    ,$CRLF          ;PRINT A CARRIAGE RETURN
2206  005774  010046          .CNVRT: MOV     R0,-(SP)        ;SAVE R0
2207  005776  010146                  MOV     R1,-(SP)        ;SAVE R1
2208  006000  010346                  MOV     R3,-(SP)        ;SAVE R3
2209  006002  010446                  MOV     R4,-(SP)        ;SAVE R4
2210  006004  010546                  MOV     R5,-(SP)        ;SAVE R5
2211  006006  017601  000012          MOV     @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
2212  006012  062766  000002  000012  ADD     #2,12(SP)       ;POINT TO WHERE MAIN PROGRAM WILL RESUME
2213  006020  012137  006144          MOV     (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
2214  006024  112105          1$:     MOVB    (R1)+,R5        ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
2215  006026  112100                  MOVB    (R1)+,R0        ;GET THE NUMBER OF SPACES TO PRINT
2216  006030  013104                  MOV     @(R1)+,R4       ;COPY THE WORD TO BE CONVERTED
2217  006032  110537  006146          MOVB    R5,CHRCNT       ;COPY THE CHARACTER COUNT
2218  006036  010403          3$:     MOV     R4,R3           ;COPY THE ARGUMENT WORD AGAIN
2219  006040  042703  177770          BIC     #^C<7>,R3       ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
2220  006044  062703  000060          ADD     #060,R3         ;MAKE AN ASCII CHARACTER OUT OF THEM
2221  006050  110346                  MOVB    R3,-(SP)        ;SAVE THAT CHARACTER
2222  006052  006004                  ROR     R4              ;MOVE THE NEXT THREE BITS INTO PLACE
2223  006054  006204                  ASR     R4              ;MOVE THEM AGAIN
2224  006056  006204                  ASR     R4              ;AND FINALLY A THIRD TIME
2225  006060  005305                  DEC     R5              ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
2226                                                          ;BUILT?
2227  006062  001365                  BNE     3$              ;IF NO, GO BUILD THE NEXT ONE.
2228  006064  012703  010410          MOV     #MDATA,R3       ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
2229  006070  112623          4$:     MOVB    (SP)+,(R3)+     ;STORE THE CHARACTER, STARTING WITH THE MOST
2230  006072  105337  006146          DECB    CHRCNT          ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
2231  006076  001374                  BNE     4$              ;IF NO, GO TRANSFER ANOTHER
2232  006100  105700                  TSTB    R0              ;ARE ANY SPACES TO BE PRINTED?
2233  006102  001404                  BEQ     6$              ;IF NO, DON'T SET UP ANY
2234  006104  112723  000040  5$:     MOVB    #040,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
2235  006110  105300                  DECB    R0              ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
2236  006112  001374                  BNE     5$              ;IF YES, GO ADD ANOTHER SPACE
2237  006114  105013          6$:     CLRB    (R3)            ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
2238  006116  104402  010410          TYPE    ,MDATA          ;PRINT THE STRING WE JUST BUILT
2239  006122  005337  006144          DEC     WRDCNT          ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
2240  006126  001336                  BNE     1$              ;IF YES, GO CONVERT THEM
2241  006130  012605                  MOV     (SP)+,R5        ;RESTORE R5
2242  006132  012604                  MOV     (SP)+,R4        ;RESTORE R4
2243  006134  012603                  MOV     (SP)+,R3        ;RESTORE R3
2244  006136  012601                  MOV     (SP)+,R1        ;RESTORE R1
2245  006140  012600                  MOV     (SP)+,R0        ;RESTORE R0
2246  006142  000002                  RTI                     ;RETURN TO THE MAIN PROGRAM
2247  006144  000000  WRDCNT: 0
2248  006146     000  CHRCNT: .BYTE                           ;NUMBER OF CHARACTERS TO PRINT
2249  006147     000  SPACNT: .BYTE   0                       ;NUMBER OF SPACES TO PRINT
2250
2251  006150  000000  BINWRD: 0
2252
2253
2254                                  ;TRAP DISPATCH SERVICE
```

```
                                        ;ARGUMENT OF TRAP IS EXTRACTED
                                        ;AND USED AS OFFSET TO OBTAIN POINTER
                                        ;TO SELECTED SUBROUTINE
                                        ;
006152  010046              .TRPSR: MOV     R0,-(SP)        ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
006154  016600  000002              MOV     2(SP),R0        ;GET TRAP ADDRESS
006160  005740                      TST     -(R0)           ;GET TRAP
006162  111000                      MOVB    (R0),R0         ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
006164  006300                      ASL     R0              ;POSITION OFFSET FOR TABLE INDEXING
006166  016000  001742              MOV     .TRPTAB(R0),R0  ;PLACE INDEXED ADDRESS OF TABLE IN R0
006172  000200                      RTS     R0              ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
                                        ;
                                        ;DEVICE CLEAR ROUTINE
                                        ;ISSUE A DEVICE CLEAR
                                        ;--------------------
006174                          .DEVICE.CLR:
006174  052777  000020  173606          BIS     #DCLR,@DZVCSR   ;SET DCLR
006202  032777  000020  173600  1$:     BIT     #DCLR,@DZVCSR   ;DID IT CLEAR?
006210  001374                      BNE     1$              ;BR IF NO
006212  000002                      RTI                     ;EXIT ROUTINE
                                        ;
                                        ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
                                        ;----------------------------------------------------------
006214  104413              .DCLASM:DEVICE.CLR              ;ISSUE A DEVICE CLEAR
006216  153777  001424  173564          BISB    MNTFLG,@DZVCSR  ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
006224  000002                      RTI                     ;RETURN TO CALLING ROUTINE
                                        ;
006226                          .DELAY:
006226  010046                      MOV     R0,-(SP)        ;SAVE R0
006230  013700  006244              MOV     DLYCNT,R0       ;SET COUNT
006234  005300                  1$: DEC     R0              ;DELAY
006236  001376                      BNE     1$              ;
006240  012600                      MOV     (SP)+,R0        ;RESTORE R0
006242  000002                      RTI                     ;LEAVE ROUTINE
006244  000001              DLYCNT: .WORD     1             ;PATCHABLE LOC FOR MORE TIME
                                        ;
                                        ;ADVANCE TO NEXT TEST HANDLER
                                        ;----------------------------
                                        ;
006246  013716  001362      .ADVANCE:MOV    NEXT,(SP)       ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
006252  005037  001364              CLR     LOCK            ;RESET TIGHT LOOP ADDRESS
006256  000002                      RTI                     ;CHECK TO SEE IF OLD TEST GETS REPEATED
                                        ;
                                        ;ROUTINE TO SHIFT LINE POINTER
                                        ;AND SWITCH TESTS IF NECESSARY
                                        ;-----------------------------
                                        ;
006260  106302              .SHIFT: ASLB    R2              ;POINT TO THE NEXT LINE
006262  032702  000020              BIT     #BIT4,R2        ;HAVE WE PASSED ALL LINE POINTERS?
006266  001402                      BEQ     1$              ;IF NOT, RETURN TO THE TEST
006270  022626                      POP2SP                  ;REMOVE THE TRAP CALL FROM THE STACK
006272  104400                      ADVANCE                 ;GO TO THE NEXT TEST
006274  000002                  1$: RTI                     ;RETURN TO THE PRESENT TEST
```

```
                                        ;LINE PARAMETER REGISTER SETUP ROUTINE
                                        ;
006276  010146          .LPRSET:MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
006300  010246                  MOV     R2,-(SP)        ;SAVE CONTENTS OF R2
006302  013701  001370          MOV     PAR,R1          ;MOVE DEFAULT PARAM. INTO R1
006306  012702  000001          MOV     #1,R2   ;INIT. FOR LINE 1
006312  010177  173502  1$:     MOV     R1,@DZVLPR      ;LOAD PARAM. REGISTER
006316  005201                  INC     R1              ;SET R1 FOR NEXT LINE
006320  106302                  ASLB    R2              ;SET R2 FOR NEXT LINE
006322  032702  000020          BIT     #BIT4,R2        ;ALL LINES DONE?
006326  001771                  BEQ     1$              ;IF NO LOAD NEXT LINE
006330  012602                  MOV     (SP)+,R2        ;RELOAD R2
006332  012601                  MOV     (SP)+,R1        ;RELOAD R1
006334  000002                  RTI                     ;RETURN
                                        ;
                                        ;ROUTINE TO ZERO DATA BUFFER
                                        ;
006336  010046          .BUFSET:MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
006340  012700  001426          MOV     #TDO,R0         ;SET R0 TO TOP OF BUFFER
006344  005020          1$:     CLR     (R0)+           ;CLEAR BUFFER LOCATION
006346  022700  001446          CMP     #STOP,R0        ;IS BUFFER ALL CLEARED
006352  001374                  BNE     1$              ;IF NOT CLEAR NEXT LOCATION
006354  012600                  MOV     (SP)+,R0        ;RELOAD R0
006356  000002                  RTI                     ;RETURN
                                        ;
                                        ;ERROR HANDLER
                                        ;-------------
                                        ;
004737  007006          $ERROR: JSR     PC,SERV.G       ;FIND OUT IF <^G> WAS HIT
032777  010000  172712          BIT     #SW12,@SWR      ;BELL ON ERROR?
001406                  BEQ     XBX             ;BR IF NO BELL
105777  172714          TSTB    @$TPS           ;TTY READY.
100003                  BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
112777  000207  172706          MOVB    #207,@$TPB      ;PUSH A BELL AT THE TTY.
032777  020000  172666  XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
001113                  BNE     HALT$           ;BR IF NO PRINT OUT WANTED.
021637  001262                  CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
001404                  BEQ     1$              ;BR IF YES
011637  001262                  MOV     (SP),$ERRPC     ;RECORD BEING HERE
105037  001247                  CLRB    $ERFLG          ;PREPARE HEADER
104407          1$:     SAV05                   ;SAVE ALL PROC REGISTERS
011605                  MOV     (SP),R5         ;GET THE PC OF ERROR
162705  000002                  SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
011504                  MOV     (R5),R4         ;GET ERROR INSTRUCTION
110437  001260                  MOVB    R4,$ITEMB       ;COPY TEST NUMBER FOR APT HANDLING
006304                  ASL     R4              ;MULT BY TWO
061504                  ADD     (R5),R4         ;DOUBLE IT
006304                  ASL     R4              ;MULT AGAIN
042704  177001                  BIC     #177001,R4      ;CLEAR JUNK
062704  016214                  ADD     #.ERRTAB,R4     ;GET POINTER
012437  006616                  MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
012437  006630                  MOV     (R4)+,DATAHD    ;GET DATA HEADRER
011437  006642                  MOV     (R4),DATABP     ;GET DATA TABLE
105737  001247                  TSTB    $ERFLG          ;TYPE HEADER
001403                  BEQ     TYPMSG          ;BR IF YES
005737  006642                  TST     DATABP          ;DOES DATA TABLE EXIST?
```

```
2364  006520  001044                        BNE     TYPDAT          ;BR IF YES.
2365  006522  104402  001357        TYPMSG: TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN
2366  006526  104402  001357                TYPE    ,$CRLF          ;AND TYPE ANOTHER
2367  006532  005737  001364                TST     LOCK
2368  006536  001402                        BEQ     1$
2369  006540  104402  007721                TYPE    ,MASTEK
2370  006544  104402  007707        1$:     TYPE    ,MTSTN
2371  006550  104412  007000                CNVRT   ,XTSTN          ;SHOW IT
2372  006554  104402  010001                TYPE    ,MERRPC         ;TYPE PC.
2373  006560  104412  006772                CNVRT   ,ERTABO         ;SHOW IT
2374  006564  104402  007651                TYPE    ,MCSRX
2375  006570  104412  004264                CNVRT   ,XCSR
2376  006574  104402  001357                TYPE    ,$CRLF          ;GIVE A CR/LF
2377  006600  112737  177777  001247        MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
2378  006606  005737  006616                TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
2379  006612  001402                        BEQ     WTBS.FM         ;BR IF NO.
2380  006614  104402                        TYPE                    ;TYPE
2381  006616  000000        ERRMSG: 0                               ;    ERROR MESSAGE
2382  006620                WTBS.FM:                                ;
2383  006620  005737  006630                TST     DATAHD          ;DATA HEADER?
2384  006624  001402                        BEQ     TYPDAT          ;BR IF NO
2385  006626  104402                        TYPE                    ;TYPE
2386  006630  000000        DATAHD: 0                               ;    DATA HEADER
2387  006632  005737  006642        TYPDAT: TST     DATABP          ;DATA TABLE?
2388  006636  001402                        BEQ     RESREG          ;BR IF NO.
2389  006640  104411                        CONVRT                  ;SHOW
2390  006642  000000        DATABP: 0                               ;    DATA TABLE
2391  006644  104410        RESREG: RES05                           ;RESTORE PROC REGISTERS
2392  006646  122737  000001  001140  HALTS:  CMPB    #APTENV,$ENV    ;IS APT RUNNING?
2393  006654  001007                        BNE     15$             ;SKIP APT CALL IF NOT
2394  006656  113737  001260  006670        MOVB    $ITEMB,5$       ;COPY ERROR NUMBER
2395  006664  004737  005122                JSR     PC,$ATY4        ;CALL APT SERVICE
2396  006670  000000        5$:     .WORD   0                       ;ERROR NUMBER STUCK HERE
2397  006672  000777        10$:    BR      10$                     ;LOCK UP HERE
2398  006674  022737  004250  000042  15$:    CMP     #$ENDAD,@#42    ;CHECK TO SEE IF IN ACT-11 MODE
2399  006702  001403                        BEQ     20$             ;IF SO, HANDLE ACCORDINGLY
2400  006704  005777  172374                TST     @SWR            ;HALT ON ERROR?
2401  006710  100004                        BPL     EXITER          ;BR IF NO HALT ON ERROR
2402  006712  016677  000002  172366  20$:    MOV     2(SP),@DISPLAY  ;SHOW ERROR PC IN DATA DISPLAY
2403  006720  000000                        HALT                    ;HALT
2404  006722  005237  001256        EXITER: INC     $ERTTL          ;UPDATE ERROR COUNT
2405  006726  004737  007006                JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
2406  006732  032777  000400  172344        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
2407  006740  001007                        BNE     1$              ;BR IF YES
2408  006742  032777  002000  172334        BIT     #SW10,@SWR      ;GOTO NEXT TEST?
2409  006750  001407                        BEQ     2$              ;BR IF NO
2410  006752  013737  001362  001252        MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
2411  006760  012706  001120        1$:     MOV     #STACK,SP       ;RESET SP
2412  006764  000177  172262                JMP     @$LPADR         ;GOTO SPECIFIED TEST
2413  006770  000002        2$:     RTI                             ;RETURN
2414  006772  000001        ERTABO: 1
2415  006774     006     002                .BYTE   6,2
2416  006776  001404                        SAVPC
2417  007000  000001        XTSTN:  1
2418  007002     002     002                .BYTE   2,2
2419  007004  001246                        $TSTNM
```

```
2420  007006  017746  172300          SERV.G: MOV     @#TKB,-(SP)     ;OTHERWISE, GET THE LAST CHARACTER TYPED
2421  007012  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY(EIGHTH) BIT
2422  007016  122726  000007                  CMPB    #7,(SP)+        ;IS IT ^G?
2423  007022  001076                          BNE     6$              ;IF NOT, IGNORE INPUT
2424  007024  032777  004000  172256          BIT     #4000,@#TKS     ;RX BUSY?
2425  007032  001365                          BNE     SERV.G          ;BR IF YES
2426  007034  017737  172244  007242          MOV     @#SWR,90$       ;SAVE (SWR).
2427  007042  104402  007222          1$:     TYPE    ,89$            ;TYPE HEADER FOR OLD SWITCH REGISTER
2428  007046  104412  007234                  CNVRT   ,88$            ;TYPE THE NUMBER ITSELF
2429  007052  104402  007244                  TYPE    ,91$            ;AFTER HAVING CONVERTED IT TO ASCII
2430  007056  105037  007250                  CLRB    92$             ;CLEAR SWR CHANGE FLAG
2431  007062  005077  172216                  CLR     @#SWR           ;CLEAR THE SOFTWARE SWITCH REGISTER
2432  007066  105777  172216          3$:     TSTB    @#TKS           ;WAIT FOR DONE.
2433  007072  100375                          BPL     3$              ;CONTINUE WAITING FOR IT
2434  007074  017746  172212                  MOV     @#TKB,-(SP)     ;PUT THE CHARACTER ON THE STACK
2435  007100  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY BIT
2436  007104  122726  000015                  CMPB    #15,(SP)+       ;IS IT THE CARRIAGE RETURN CHAR?
2437  007110  001433                          BEQ     4$              ;IF SO, GO PRINT CRLF
2438  007112  105777  172176          2$:     TSTB    @#TPS           ;IS THE OUTPUT BUFFER AVAILABLE
2439  007116  100375                          BPL     2$              ;IF NOT, WAIT FOR IT TO BE READY
2440  007120  105237  007250                  INCB    92$             ;INDICATE THAT THE SWR WAS CHANGED
2441  007124  014677  172166                  MOV     -(SP),@#TPB     ;PLACE THE CHARACTER THERE(ECHO BACK)
2442  007130  000241                          CLC                     ;GET READY TO ROTATE
2443  007132  006177  172146                  ROL     @#SWR           ;MOVE THE EXISTING BITS OVER
2444  007136  006177  172142                  ROL     @#SWR           ;TO MAKE ROOM FOR THE INCOMING
2445  007142  006177  172136                  ROL     @#SWR           ;THREE BITS FROM THIS CHARACTER
2446  007146  103735                          BCS     1$              ;ERROR
2447  007150  022627  000060                  CMP     (SP)+,#60       ;IS IT LOWER THAN 0?
2448  007154  002732                          BLT     1$              ;IF SO, GO ASK AGAIN
2449  007156  026627  177776  000067          CMP     -2(SP),#67      ;IS IT HIGHER THAN 7?
2450  007164  003326                          BGT     1$              ;IF SO, GO ASK AGAIN
2451  007166  042746  177770                  BIC     #^C<7>,-(SP)    ;ISOLATE INFORMATION BITS
2452  007172  052677  172106                  BIS     (SP)+,@#SWR     ;ADD THEM TO THE SWITCH REGISTER
2453  007176  000733                          BR      3$              ;GO CHECK FOR THE NEXT CHARACTER
2454  007200  105737  007250          4$:     TSTB    92$             ;HAS THE SWR BEEN CHANGED?
2455  007204  001003                          BNE     5$              ;IF YES GO TYPE CRLF
2456  007206  013777  007242  172070          MOV     90$,@#SWR       ;IF NOT RESTORE SWR
2457  007214  104402  001357          5$:     TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
2458  007220  000207                  6$:     RTS     PC              ;RETURN TO CALLING PROCEDURE
2459
2460  007222  020200  051450  051127  89$:     .ASCIZ  <200>? (SWR)=/?
2461  007230  036451  000057
2462                                  .EVEN
2463  007234  000001                  88$:    1
2464  007236     006     000                  .BYTE   6,0
2465  007240  007242                          90$
2466  007242  000000                  90$:    .WORD   0
2467  007244  036457  000057          91$:    .ASCIZ  ?/=/?
2468  007250     000                  92$:    .BYTE   0
2469          007252                  .EVEN
2470                                  .SBTTL  POWER DOWN AND UP ROUTINES
2471
2472                                  ;;**************************************************************************
2473                                  ;POWER DOWN ROUTINE
2474  007252  012737  007416  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2475  007260  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
```

```
2476  007266  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2477  007270  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2478  007272  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2479  007274  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2480  007276  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2481  007300  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2482  007302  017746  171776                MOV     @#PS,-(SP)      ;;PUSH @#PS ON STACK
2483  007306  010637  007422                MOV     SP,$SAVR6       ;;SAVE SP
2484  007312  012737  007324  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2485  007320  000000                        HALT
2486  007322  000776                        BR      .-2             ;;HANG UP
2487
2488                                ;;*************************************************************
2489                                ;;POWER UP ROUTINE
2490  007324  012737  007416  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2491  007332  013706  007422                MOV     $SAVR6,SP       ;;GET SP
2492  007336  005037  007422                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2493  007342  005237  007422          1$    INC     $SAVR6          ;;WAIT FOR THE INC
2494  007346  001375                        BNE     1$              ;;OF  WORD
2495  007350  012677  171730                MOV     (SP)+,@#PS      ;;POP STACK INTO @#PS
2496  007354  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2497  007356  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
2498  007360  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2499  007362  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2500  007364  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2501  007366  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2502  007370  012737  007252  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2503  007376  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
2504  007404  104402                        TYPE                    ;;REPORT THE POWER FAILURE
2505  007406  007424                $PWRMG  .WORD   MPFAIL          ;;POWER FAIL MESSAGE POINTER
2506  007410  012716                        MOV     (PC)+,(SP)      ;;RESTART AT RESTART
2507  007412  011012                $PWRAD: .WORD   RESTART         ;;RESTART ADDRESS
2508  007414  000002                        RTI
2510  007420  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2511  007422  000000                $SAVR6: 0                       ;;PUT THE SP HERE
2512  007424  050200  051127  043040 MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT LAST TEST /
 (2)  007467     200  047105  020104 MEPASS: .ASCIZ  <200>/END PASS CVDZA-B  /
 (2)  007513     200  052522  047116 MR:     .ASCIZ  <200>/RUNNING   /
 (2)  007527     200  051120  043517 MERR2:  .ASCIZ  <200>/PROGRAM INDICATES NO DEVICES PRESENT./
 (2)  007576  044600  051516  043125 MERR3:  .ASCIZ  <200>/INSUFFICIENT DATA'/
 (2)  007622  046200  041517  020113 MLOCK:  .ASCIZ  <200>/LOCK ON SELECTED TEST/
 (2)  007651     103  051123  020072 MCSRX:  .ASCIZ   /CSR: /
 (2)  007657     126  041505  020072 MVECX:  .ASCIZ  /VEC: /
 (2)  007665     120  051501  042523 MPASSX: .ASCIZ  /PASSES: /
 (2)  007676  051105  047522  051522 MERRX:  .ASCIZ  /ERRORS: /
 (2)  007707     124  051505  020124 MTSTN:  .ASCIZ  /TEST NO: /
 (2)  007721     052  000040        MASTEK: .ASCIZ  /* /
 (2)  007724  052200  050131  020105 MNEW:   .ASCIZ  <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE    /
 (2)  010001     120  035103  000040 MERRPC: .ASCIZ  /PC: /
 (2)  010006  046600  050101  047440 XHEAD:  .ASCIZ  <200>/MAP OF DZV11 STATUS/<200>
 (2)  010034  044600  046114  043505 MBADLN: .ASCIZ  <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
 (2)                                .EVEN
 (2)  010076  000002                XSTATQ: 2
2513  010100     006     003                .BYTE   6,3
2514  010102  001344                        $TMP1
```

```
010104  006     002             .BYTE   6.2
010106  001346                  $TMP2
                        .EVEN
                        ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
                        ;-----------------------------------------------------------
                        ;E=EXTERNAL LOOP BACK
                        ;I=INTERNAL LOOP BACK
                        ;S=STAGGERED LOOP BACK
010110  017605  000000          .SETFLG:MOV     @(SP),R5        ;PICK UP ADDRESS OF TAG
010114  042737  000040  010304          BIC     #40,INBUF       ;STRIP LOWER CASE
010122  122737  000105  010304          CMPB    #'E,INBUF       ;IS IT EXTERNAL LOOP BACK ?
010130  001005                          BNE     4$              ;NO
010132  013715  010222                  MOV     1$,(R5)         ;YES STORE INFO
010136  105037  001424                  CLRB    MNTFLG          ;SET MAINT BIT =0
010142  000422                          BR      7$              ;GET OUT
010144  122737  000111  010304  4$      CMPB    #'I,INBUF       ;IS IT INTERNAL LOOP BACK ?
010152  001006                          BNE     5$              ;NO
010154  013715  010224                  MOV     2$,(R5)         ;YES STORE INFO
010160  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;SET UP THE MAINTENANCE FLAG LOADER
010166  000410                          BR      7$              ;GET OUT
010170  122737  000123  010304  5$      CMPB    #'S,INBUF       ;IS IT STAGGERED LOOP BACK ?
010176  001007                          BNE     6$              ;WHAT ?
010200  013715  010226                  MOV     3$,(R5)         ;YES STORE INFO
010204  105037  001424                  CLRB    MNTFLG          ;ZERO BITS
010210  062716  000002          7$:     ADD     #2,(SP)         ;POP AROUND
010214  000002                          RTI
010216  104404                  6$:     INSTER                  ;RETRY
010220  000733                          BR      .SETFLG         ;DITTO
010222  000200                  1$:     .WORD   200             ;EXTERNAL =  E
010224  000000                  2$      .WORD   0               ;INTERNAL = I
010226  100000                  3$      .WORD   100000          ;STAGGERED = S
```

```
2547                                        ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
2548                                        ;BUFFER TO THE CHARACTERS 'E' AND 'C'.
2549                                        ;IF THE CHARACTER IS 'E' CLEAR THE FLAG
2550                                        ;IF THE CHARACTER IS 'C' SET THE FLAG
2551
2552  010230  017605  000000          .PAUCH: MOV     @(SP),R5
2553  010234  142737  000040  010304          BICB    #40,INBUF       ;SET FOR LOWER CASE INPUT
2554  010242  122737  000105  010304          CMPB    #'E,INBUF       ;IS IT 'E' ?
2555  010250  001002                          BNE     1$
2556  010252  105015                          CLRB    (R5)            ;000
2557  010254  000406                          BR      2$
2558  010256  122737  000103  010304  1$:     CMPB    #'C,INBUF       ;IS IT 'C' ?
2559  010264  001005                          BNE     3$
2560  010266  112715  177777                  MOVB    #-1,(R5)        ;3177
2561  010272  062716  000002          2$:     ADD     #2,(SP)
2562  010276  000002                          RTI
2563  010300  104404                  3$:     INSTER                  ;RETRY
2564  010302  000752                          BR      .PAUCH
2565
2566                                        ;BUFFERS FOR INPUT-OUTPUT
2567
2568  010304  000000                  INBUF:  0
2569          010346                  .=.+40
2570  010346  000000                  TEMP    0
2571          010410                  .=.+40
2572  010410  000000                  MDATA:  0
2573          010452                  .=.+40
2574
```

```
2575
2576                                            ;
2577                                            ;ROUTINE USED TO "CYCLE" THROUGH UP TO SIXTEEN DZV11'S
2578                                            ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
2579                                            ;AND RUNS THE SPECIFIED DZV11'S.   THIS ROUTINE *MUST*
2580                                            ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
2581                                            ;SETUP NECESSARY.
2582                                            ;
2583
2584  010452  005737  001406          CYCLE:  TST     DZVACTV         ;ARE ANY DZV11'S TO BE TESTED?
2585  010456  001004                          BNE     1$              ;BR IF OK.
2586  010460  104402  007527                  TYPE    ,MERR2          ;NO DZV11'S SELECTED!!
2587  010464  000000                          HALT                    ;STOP THE SHOW.
2588  010466  000776                          BR      .-2             ;DISQUALIFY CONT. SW.
2589  010470  013737  004556  001354  1$:     MOV     $MXCNT,$TIMES   ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
2590  010476  033737  001412  001406          BIT     RUN,DZVACTV     ;IS THIS ONE "ACTIVE"
2591  010504  001017                          BNE     2$              ;BR IF GOOD ONE FOUND.
2592  010506  006137  001412                  ROL     RUN             ;UPDATE POINTER
2593  010512  005537  001412                  ADC     RUN             ;CATCH CARRY FROM RUN
2594  010516  062737  000012  001420          ADD     #12,ACTIVE      ;UPDATE ADDRESS POINTER.
2595  010524  022737  001740  001420          CMP     #DZV.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
2596  010532  001356                          BNE     1$              ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
2597  010534  012737  001500  001420          MOV     #DZV.MAP,ACTIVE ;RESET ADDRESS POINTER.
2598  010542  000752                          BR      1$              ;KEEP LOOKING FOR ACTIVE DZV11
2599  010544  006137  001412          2$:     ROL     RUN             ;UPDATE POINTER.
2600  010550  005537  001412                  ADC     RUN             ;CATCH CARRY.
2601  010554  013700  001420                  MOV     ACTIVE,R0       ;GET ADDRESS POINTER.
2602  010560  062737  000012  001420          ADD     #12,ACTIVE      ;UPDATE.
2603  010566  022737  001740  001420          CMP     #DZV.END,ACTIVE
2604                                                                  ;ALL DONE?
2605  010574  001003                          BNE     3$              ;BR IF NO.
2606  010576  012737  001500  001420          MOV     #DZV.MAP,ACTIVE ;RESTORE POINTER.
2607  010604  012037  001174          3$:     MOV     (R0)+,$BASE     ;LOAD SYSTEM CTRL. REG
2608  010610  012037  002040                  MOV     (R0)+,DZVRIV    ;LOAD VECTOR
2609  010614  012037  001366                  MOV     (R0)+,LINE      ;SET UP DZV LINES ACTIVE
2610  010620  012037  001370                  MOV     (R0)+,PAR       ;SET UP PARAMETERIZATION
2611  010624  012037  001372                  MOV     (R0)+,MODE      ;SET UP MAINTENANCE MODE
2612  010630  105037  001424                  CLRB    MNTFLG  ;RESET MAINT. FLAG IF
2613  010634  005737  001372                  TST     MODE            ;RUNNING TESTS
2614  010640  001003                          BNE     9$              ;IN
2615  010642  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;INTERNAL MAINT. MODE
2616  010650  004737  011016          9$:     JSR     PC,DZVLEV       ;SET UP
2617  010654  005737  000042                  TST     @#42            ;ARE WE UNDER MONITOR CONTROL?
2618  010660  001051                          BNE     7$              ;IF YES, SKIP THIS SETUP
2619  010662  032777  000002  170414          BIT     #SW01,@SWR      ;IF SW01=1, GET STARTING TEST #
2620  010670  001445                          BEQ     7$              ;BR IF NO TEST IS TO BE INPUTTED
2621  010672  104402  001357          4$:     TYPE    ,$CRLF
2622  010676  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
2623  010700  007707                          MTSTN                   ;POINTER TO MESSAGE TO BE PRINTED
2624  010702  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
2625  010704  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
2626  010706  001000                          1000                    ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
2627  010710  001246                          $TSTNM                  ;POINTER TO MAP LOCATION TO BE FILLED
2628  010712     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
2629  010713     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
2630  010714  012700  012204                  MOV     #TST1,R0
```

```
2631  010720  022710  000004          5$:     CMP     #4,(R0)
2632  010724  001020                          BNE     6$
2633  010726  022760  012737  C00002          CMP     #12737,2(R0)
2634  010734  001014                          BNE     6$
2635  010736  023760  001246  000004          CMP     $TSTNM,4(R0)    ;IS THIS THE TEST ?
2636  010744  001010                          BNE     6$              ;IF NOT, DON'T PROCESS NUMBER
2637  010746  010037  001252                  MOV     R0,$LPADR       ;SAVE PC
2638  010752  062737  000002  001252          ADD     #2,$LPADR       ;POP OVER PREVIOUS SCOPE
2639  010760  104402  001357                  TYPE    ,$CRLF
2640  010764  000412                          BR      8$
2641  010766  005720                  6$:     TST     (R0)+
2642  010770  020027  015662                  CMP     R0,#TLAST+10
2643  010774  001351                          BNE     5$
2644  010776  104402  001356                  TYPE    ,$QUES
2645  011002  000733                          BR      4$
2646  011004  012737  012204  001252  7$:     MOV     #TST1,$LPADR    ;PREPARE TEST ADDRESS
2647  011012                          8$:
2648  011012  000177  170234          RESTART:JMP     @$LPADR         ;GO START TESTING.***WARNING'****
2649                                                                  ;THIS JUMP IS USED BY POWER UP ROUTINE!''
2650
2651                                  ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
2652  011016  013700  002040          DZVLEV: MOV     DZVRIV,R0       ;PLACE THE BASE VECTOR ADDRESS IN R0
2653  011022  062700  000002                  ADD     #2,R0           ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
2654  011026  010037  002042                  MOV     R0,DZVRIS       ;STORE IT HERE
2655  011032  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
2656  011036  010037  002044                  MOV     R0,DZVTIV       ;STORE IT HERE
2657  011042  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
2658  011046  010037  002046                  MOV     R0,DZVTIS       ;STORE IT HERE
2659
2660                                  ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
2661                                  ;OF THE DEVICE
2662  011052  013700  001174                  MOV     $BASE,R0        ;COPY THE ADDRESS BEING LOADED
2663  011056  010037  002010                  MOV     R0,DZVCSR       ;XXX0
2664  011062  005200                          INC     R0
2665  011064  010037  002012                  MOV     R0,HDZVCSR      ;XXX1
2666  011070  005200                          INC     R0
2667  011072  010037  002014                  MOV     R0,DZVRBUF      ;XXX2
2668  011076  010037  002020                  MOV     R0 DZVLPR       ;XXX2
2669  011102  005200                          INC     R0
2670  011104  010037  002016                  MOV     R0,HDZVRBUF     ;XXX3
2671  011110  010037  002022                  MOV     R0,HDZVLPR      ;XXX3
2672  011114  005200                          INC     R0
2673  011116  010037  002024                  MOV     R0,DZVTCR       ;XXX4
2674  011122  005200                          INC     R0
2675  011124  010037  002026                  MOV     R0,HDZVTCR      ;XXX5
2676  011130  005200                          INC     R0
2677  011132  010037  002030                  MOV     R0,DZVMSR       ;XXX6
2678  011136  010037  002034                  MOV     R0,DZVTDR       ;XXX6
2679  011142  005200                          INC     R0
2680  011144  010037  002032                  MOV     R0,HDZVMSR      ;XXX7
2681  011150  010037  002036                  MOV     R0,HDZVTDR      ;XXX7
2682  011154  000207                          RTS     PC
```

```
2683                                        ;CONVERT DECIMAL ASCII STRING TO OCTAL
2684  011156  011605              .PARMD: MOV     (SP),R5
2685  011160  012537  011342              MOV     (R5)+,6$
2686  011164  012537  011344              MOV     (R5)+,7$
2687  011170  012537  011346              MOV     (R5)+,8$
2688  011174  112537  011350              MOVB    (R5)+,9$
2689  011200  112537  011351              MOVB    (R5)+,10$
2690  011204  010516                      MOV     R5,(SP)
2691  011206  005005                2$:   CLR     R5
2692  011210  012704  010304              MOV     #INBUF,R4
2693  011214  122714  000015              CMPB    #15,(R4)
2694  011220  001424                      BEQ     3$
2695  011222  121427  000060        1$:   CMPB    (R4),#'0
2696  011226  002421                      BLT     3$
2697  011230  121427  000071              CMPB    (R4),#'9
2698  011234  003016                      BGT     3$
2699  011236  142714  000060              BICB    #'0,(R4)
2700  011242  005002                      CLR     R2
2701  011244  152402                      BISB    (R4)+,R2
2702  011246  060205                      ADD     R2,R5
2703  011250  122714  000015              CMPB    #15,(R4)
2704  011254  001410                      BEQ     4$
2705  011256  006305                      ASL     R5          ;X2
2706  011260  010502                      MOV     R5,R2       ;SAVE X2
2707  011262  006305                      ASL     R5          ;X4
2708  011264  006305                      ASL     R5          ;X8
2709  011266  060205                      ADD     R2,R5       ;TIMES 10
2710  011270  000754                      BR      1$
2711  011272  104404                3$:   INSTER
2712  011274  000744                      BR      2$
2713
2714                                        ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
2715
2716  011276  020537  011344        4$:   CMP     R5,7$
2717  011302  101373                      BHI     3$
2718  011304  020537  011342              CMP     R5,6$
2719  011310  103770                      BLO     3$
2720  011312  133705  011350              BITB    9$,R5
2721  011316  001365                      BNE     3$
2722
2723                                        ;STORE NUMBER AT SPECIFIED ADDRESS
2724
2725  011320  013704  011346              MOV     8$,R4
2726  011324  010524                5$:   MOV     R5,(R4)+
2727  011326  062705  000002              ADD     #2,R5
2728  011332  105337  011351              DECB    10$
2729  011336  001372                      BNE     5$
2730  011340  000002                      RTI
2731  011342  000000                6$:   0
2732  011344  000000                7$:   0
2733  011346  000000                8$:   0
2734  011350     000                9$:   .BYTE 0
2735  011351     000                10$:  .BYTE 0
```

```
2736                                    ;*ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
2737                                    ;*IF BIT7 IN THE ENVIRONMENT MODE (SENVM) BYTE IS SET,
2738                                    ;*THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
2739
2740  011352  012700  001500   SETAPT: MOV     #DZV.MAP,R0     ;POINT TO THE DEVICE MAP TABLE
2741  011356  013701  001174           MOV     $BASE,R1        ;BUILD DEVICE ADDRESSES IN R1
2742  011362  013702  001170           MOV     $VECT1,R2       ;BUILD DEVICE VECTORS IN R2
2743  011366  042702  177007           BIC     #^C<770>,R2     ;STRIP AWAY OTHER INFORMATION
2744
2745  011372  012704  001204           MOV     #$DDW0,R4       ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
2746  011376  013705  001176           MOV     $DEVM,R5        ;GET THE MAP OF ACTIVE DEVICES
2747  011402  105037  001414           CLRB    DZVNUM          ;INITIALIZE NO. OF DEVICES IN SYSTEM
2748  011406  005037  001410           CLR     SAVACTV         ;CLEAR THE ACTIVE BIT MAP
2749  011412  006005           1$:     ROR     R5              ;GET A DEVICE SELECTION BIT
2750  011414  103407                   BCS     3$              ;IF IT IS SELECTED, GO SET UP A MAP
2751  011416  001422                   BEQ     5$              ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
2752  011420  005724                   TST     (R4)+           ;POINT TO NEXT DEVICE DESCRIPTOR
2753  011422  062701  000010   2$:     ADD     #10,R1          ;SET UP THE NEXT ADDRESS
2754  011426  062702  000010           ADD     #10,R2          ;SET UP THE NEXT VECTOR GROUP
2755  011432  000767                   BR      1$              ;GO SEE IF MORE DEVICES REMAIN
2756  011434  006137  001410   3$:     ROL     SAVACTV         ;SET BIT IN ACTIVE DEVICE MAP
2757  011440  105237  001414           INCB    DZVNUM          ;INCREMENT NO. OF ACTIVE DEVICES IN SYSTEM
2758  011444  010120                   MOV     R1,(R0)+        ;LOAD DEVICE ADDRESS
2759  011446  010220                   MOV     R2,(R0)+        ;LOAD THE VECTOR ADDRESS
2760  011450  013720  001200           MOV     $CDW1,(R0)+     ;GET THE NUMBER OF LINES IN OPERATION
2761  011454  012420                   MOV     (R4)+,(R0)+     ;LOAD DEVICE PARAMETERS
2762  011456  013720  001202           MOV     $CDW2,(R0)+     ;LOAD DEFAULT TESTING MODE
2763  011462  000757                   BR      2$              ;GO BUILD THE NEXT ADDRESS
2764  011464  012710  177777   5$:     MOV     #-1,(R0)        ;TERMINATE THE DEVICE MAP
2765  011470  012737  001142  001304   MOV     #$SWREG,SWR     ;SET TO SOFTWARE APT SWITCH REGISTER
2766  011476  000207                   RTS     PC              ;RETURN TO PRINT STATUS TABLE
2767
2768
2769                                    ;*ROUTINE USED TO "AUTO SIZE" THE DZV11
2770                                    ;*CSR AND VECTOR.
2771                                    ;*NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
2772                                    ;*      ADDRESS RANGE (160000:163770)
2773                                    ;*      AND THE VECTOR MAY BE ANY WHERE IN THE
2774                                    ;*      FLOATING VECTOR RANGE (300:770)
2775                                    ;*
2776
2777  011500                   AUTO.SIZE:
2778  011500  000005                   RESET                   ;INSURE A BUS INIT.
2779  011502  105337  001422           DECB    INIFLG          ;SHOW THAT I WAS HERE
2780  011506  012702  001500   CSRMAP: MOV     #DZV.MAP,R2     ;LOAD MAP POINTER.
2781  011512  012703  001204           MOV     #$DDW0,R3       ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
2782  011516  005022           1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
2783  011520  022702  001740           CMP     #DZV.END,R2     ;ALL DONE?
2784  011524  001374                   BNE     1$              ;BR IF NO
2785  011526  105037  001414           CLRB    DZVNUM          ;SET OCTAL NUMBER OF DZV11'S TO 0
2786  011532  012702  001500           MOV     #DZV.MAP,R2
2787  011536  012701  160000           MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
2788  011542  012737  012006  000004   MOV     #6$,@#4         ;SET FOR NON-EXISTENT DEVICE TIME OUT
2789  011550  052711  000040   2$:     BIS     #BIT5,(R1)      ;TRY TO SET MASTER SCAN ENABLE
2790  011554  052761  000017  000004   BIS     #17,4(R1)       ;TRY TO TRANSMIT ON ANY LINE
2791  011562  005000                   CLR     R0              ;USE R0 AS A COUNTER
```

```
2792  011564  005711                    7$:     TST     (R1)            ;HAS TRANSMITTER READY COME UP?
2793  011566  100403                            BMI     8$              ;IF SO, GO GET A FINAL CHECK
2794  011570  005300                            DEC     R0              ;REDUCE COUNT. TIME UP?
2795  011572  001374                            BNE     7$              ;IF NOT, KEEP WAITING
2796  011574  000437                            BR      3$              ;ASSUME IT'S NOT A DZV11
2797  011576  032761  000017  000004    8$:     BIT     #17,4(R1)       ;ARE ANY TCR BITS  STILL SET? THEY SHOULD BE
2798  011604  001433                            BEQ     3$              ;IF IT'S NOT, ASSUME IT'S NOT A DZV11
2799  011606  032711  000040                    BIT     #BIT5,(R1)      ;IS MASTER SCAN ENABLE STILL SET?
2800  011612  001430                            BEQ     3$              ;IF NOT, ASSUME IT'S NOT A DZV11
2801  011614  052711  000020                    BIS     #20,(R1)        ;SET DEVICE CLEAR
2802  011620  000240                            NOP
2803  011622  032711  00004C                    BIT     #40,(R1)        ;DID SCANNER CLEAR
2804  011626  001022                            BNE     3$              ;IF NOT ASSUME IT IS NOT DZV
2805  011630  005061  000004                    CLR     4(R1)           ;GET RID OF TCR BITS
2806                                    ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZV11 CSR ADDRESS.
2807  011634  010122                            MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
2808  011636  005722                            TST     (R2)+           ;POP OVER VECTOR STORE AREA
2809  011640  012722  000017                    MOV     #17,(R2)+       ;SET THE DEFAULT LINE SELECTION PARAMETER
2810  011644  012712  017470                    MOV     #17470,(R2)     ;SET THE DEFAULT PARAMETERS
2811  011650  012223                            MOV     (R2)+,(R3)+     ;COPY PARAMETERS INTO ETABLE DESCRIPTOR
2812  011652  005022                            CLR     (R2)+           ;SET THE DEFAULT MODE OF OPERATION
2813  011654  012712  177777                    MOV     #-1,(R2)        ;TERMINATE LIST
2814  011660  105237  001414                    INCB    DZVNUM          ;UPDATE DEVICE COUNTER
2815  011664  122737  000020  001414          CMPB    #20,DZVNUM      ;ARE MAX. NO. OF DEV FOUND?
2816  011672  001405                            BEQ     100$            ;YES DON'T LOOK FOR ANY MORE.
2817  011674  062701  000010            3$:     ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
2818  011700  022701  164000                    CMP     #164000,R1
2819  011704  001321                            BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
2820  011706                            100$:
2821  011706  105737  001414                    TSTB    DZVNUM          ;WERE ANY DZV11'S FOUND AT ALL?
2822  011712  001430                            BEQ     5$              ;ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
2823  011714  113701  001414                    MOVB    DZVNUM,R1
2824  011720  012737  000001  001410          MOV     #1,SAVACTV      ;CREATE A BIT MAP OF THE ACTIVE
2825  011726  005301                    4$:     DEC     R1              ;DEVICES IN THE SYSTEM
2826  011730  001404                            BEQ     98$
2827  011732  000261                            SEC
2828  011734  006137  001410                    ROL     SAVACTV
2829  011740  000772                            BR      4$
2830  011742  013737  001500  001174    98$:    MOV     DZCR0,$BASE     ;POINT TO THE ADDRESS OF FIRST DEVICE
2831  011750  013737  001510  001202            MOV     MAN10,$CDW2     ;INDICATE TO ETABLE WHAT MODE IS BEING USED
2832  011756  012737  000006  000004    99$:    MOV     #6,a#4          ;RESTORE TRAP VECTOR
2833  011764  013737  001410  001176            MOV     SAVACTV,$DEVM   ;SAVE ACTIVE REGISTER
2834  011772  000410                            BR      VECMAP          ;GO FIND THE VECTOR NOW.
2835  011774  104402  007527            5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO DZV11'S FOUND.
2836  012000  005000                            CLR     R0              ;MAKE DATA DISPLAY ZERO
2837  012002  000000                            HALT                    ;STOP THE SHOW
2838  012004  000776                            BR      .-2             ;DISABLE CONT. SW.
2839  012006  012716  011674            6$:     MOV     #3$,(SP)        ;ENTERED BY NON-EXISTENT TIME-OUT
2840  012012  000002                            RTI                     ;RETURN TO MAINSTREAM
2841
2842  012014  012737  000200  000022    VECMAP: MOV     #MASK,a#22      ;SET IOT TRAP PRIORITY
2843  012022  012737  012136  000020            MOV     #4$,a#20        ;SET IOT TRAP VECTOR
2844  012030  012702  001500                    MOV     #DZV.MAP,R2     ;SET SOFTWARE POINTER
2845  012034  012700  000300                    MOV     #300,R0         ;FLOATING VECTORS START HERE.
2846  012040  012701  000302                    MOV     #302,R1         ;PC OF IOT INSTR.
2847  012044  010120                    1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
```

```
2848  012046  012721  000004                MOV     #4,(R1)+        ;WITH .+2; IOT
2849  012052  022021                       CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
2850  012054  020127  001000                CMP     R1,#1000        ;HAS THE VECTOR AREA BEEN EXCEEDED?
2851  012060  101771                       BLOS    1$              ;BR IF MORE TO FILL
2852  012062  013704  001410                MOV     SAVAC^V,R4      ;STORE TEMPORARILY
2853  012066  006004              2$:       ROR     R4              ;BRING OUT A BIT
2854  012070  103036                       BCC     5$              ;BR IF ALL DONE
2855  012072  106427  000000                MTPS    #0              ;ZERO CPU PRIO
2856  012076  012772  040040  000000        MOV     #BIT14+BIT5,@(R2)       ;SET TIE AND MAS SCAN
2857  012104  011201                       MOV     (R2),R1         ;GET CSR
2858  012106  112761  000017  000004        MOVB    #17,4(R1)       ;SET THE TCR BITS FOR ALL LINES
2859                                        ;ATTEMPT TO FORCE AN INTERRUPT
2860  012114  005200                       INC     R0              ;STALL
2861  012116  001376                       BNE     .-2             ;       FOR TIME TO INTERRUPT
2862  012120  012762  000300  000002        MOV     #300,2(R2)      ;NO INTERRUPT ASSUME 300 AND FIX DZV11 LATER
2863  012126  000005                       RESET                   ;INIT
2864  012130  062702  000012              3$:       ADD     #12,R2          ;POP SOFTWARE POINTER
2865  012134  000754                       BR      2$              ;KEEP GOING
2866  012136  011662  000002              4$:       MOV     (SP),2(R2)      ;GET VECTOR ADDRESS
2867  012142  162762  000010  000002        SUB     #10,2(R2)       ;POINT BACK TO THE CORRECT VECTOR
2868  012150  042762  000007  000002        BIC     #7,2(R2)        ;CLEAR JUNK
2869  012156  022626                       POP2SP                  ;POP IOT JUNK OFF STACK
2870  012160  012716  012130                MOV     #3$,(SP)        ;SET FOR RETURN
2871  012164  000002                       RTI
2872  012166  013737  001502  001170  5$:   MOV     DZVC0,$VECT1    ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
2873  012174  012737  004314  000020        MOV     #.SCOPE,IOTVEC  ;RESTORE THE SCOPE TRAP
2874  012202  000207                       RTS     PC              ;ALL DONE WITH "AUTO SIZING"
2875
```

```
2876
2877                                    ;********************** TEST 1 ****************************
2878                                    ;*THIS TEST PROVES THE BUS REPLY RESPONSE
2879                                    ;*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
2880                                    ;*      DZVCSR, DZVRBUF, DZVTCR, DZVMSR
2881                                ;:*  TEST 1
2882                                ;:*************************************************************
2883  012204  000004               TST1:   SCOPE
2884  012206  012737  000001  001246       MOV     #1,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
2885  012214  012737  012374  001362       MOV     #TST2,NEXT      ;POINT TO THE START OF THE NEXT TEST
2886  012222  012737  012362  000004       MOV     #5$,4           ;SET TRAP VECTOR
2887  012230  012737  000200  000006       MOV     #MASK,6         ;SET PRIORITY TO HIGH(MASK INTERRUPTS)
2888  012236  012737  012244  001364       MOV     #1$,LOCK        ;SET RETURN IF SW09=11
2889  012244  013700  002010          1$:  MOV     DZVCSR,R0       ;SET ADDRESS TO TEST
2890  012250  011001                       MOV     (R0),R1         ;READ THE ADDRESS
2891  012252  000240                       NOP                     ;WASTE TIME
2892  012254  005010                       CLR     (R0)            ;WRITE THE ADDRESS
2893  012256  000240                       NOP                     ;WASTE TIME
2894  012260  012737  012266  001364       MOV     #2$,LOCK        ;SET RETURN ADDRESS FOR SW09
2895  012266  013700  002014          2$:  MOV     DZVRBUF,R0      ;SET ADDRESS TO TEST
2896  012272  011001                       MOV     (R0),R1         ;READ THE ADDRESS
2897  012274  000240                       NOP                     ;
2898  012276  005010                       CLR     (R0)            ;WRITE THE ADDRESS
2899  012300  000240                       NOP                     ;WASTE TIME
2900  012302  012737  012310  001364       MOV     #3$,LOCK        ;SET RETURN ADDRESS FOR SW09
2901  012310  013700  002024          3$:  MOV     DZVTCR,R0       ;SET ADDRESS TO TEST
2902  012314  011001                       MOV     (R0),R1         ;READ THE ADDRESS
2903  012316  000240                       NOP
2904  012320  005010                       CLR     (R0)            ;WRITE THE ADDRESS
2905  012322  000240                       NOP
2906  012324  012737  012332  001364       MOV     #4$,LOCK        ;SET RETURN ADDRESS
2907  012332  013700  002030          4$:  MOV     DZVMSR,R0       ;SET ADDRESS TO TEST
2908  012336  011001                       MOV     (R0),R1         ;READ FROM ADDRESS
2909  012340  000240                       NOP
2910  012342  005010                       CLR     (R0)            ;WRITE THE ADDRESS
2911  012344  000240                       NOP                     ;
2912  012346  012737  000006  000004       MOV     #6,4            ;SET TRAP CATCHER BACK TO NORMAL
2913  012354  005037  000006               CLR     6               ;
2914  012360  104400                       ADVANCE                 ;SCOPE THIS TEST
2915  012362  011601                  5$:  MOV     (SP),R1         ;SAVE PC OF TRAP
2916  012364  022626                       POP2SP                  ;POP TRAP OFF STACK
2917  012366  104001                       ERROR   1               ;*NO BUS REPLY RESPONSE.
2918  012370  104401                       SCOP1                   ;SW09=1?
2919  012372  000111                       JMP     (R1)            ;RTI
2920                                    ;********************** TEST 2 ****************************
2921                                    ;*THIS TEST PROVES THAT BIT 'DCLR'
2922                                    ;*CAN BE SET AND THAT IT WILL CLEAR
2923                                    ;*BY ITSELF
2924                                ;:*  TEST 2
2925                                ;:*************************************************************
2926  012374  000004               TST2:   SCOPE
2927  012376  012737  000002  001246       MOV     #2,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
2928  012404  012737  012440  001362       MOV     #TST3,NEXT      ;POINT TO THE START OF THE NEXT TEST
2929  012412  013700  002010               MOV     DZVCSR,R0       ;SET POINTER
2930  012416  012710  000020               MOV     #DCLR,(R0)      ;SET DCLR
2931  012422  005005                       CLR     R5              ;SET EXPECTED TO 0
```

```
012424  005003                          CLR     R3              ;DUAL LOOP COUNTER
012426  011004                  2$:     MOV     (R0),R4         ;IS DCLR CLEAR?
012430  001403                          BEQ     3$              ;IF YES , GO TO THE NEXT TEST
012432  105203                          INCB    R3              ;IF NO,COUNT 1 OF 256 TICKS
012434  001374                          BNE     2$              ;HAS THE TIME EXPIRED? IF NO, GO TEST BIT AGAIN
012436  104002                          ERROR   2               ;*DCLR FAILED TO CLEAR
012440                          3$:
                                        ;*********************** TEST 3 ******************************
                                        ;*TEST TO VERIFY THAT THE R/W BITS OF THE
                                        ;*DZVCSR REGISTER CAN BE SET. THEN VERIFY THAT
                                        ;*THESE BITS CAN BE CLEARED. AND FINALLY, VERIFY
                                        ;*THAT AFTER BEING SET AGAIN THEY CAN BE
                                        ;*CLEARED BY A 'DEVICE CLEAR'.
                                        ;*THE BITS TESTED ARE:  MAINT, MSENAB, SILOEN,
                                        ;*RIE, AND TIE.
                                ;.*  TEST 3
                                ;;*************************************************************
012440  000004                  TST3:   SCOPE
012442  012737  000003  001246          MOV     #3,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
012450  012737  012616  001362          MOV     #TST4,NEXT      ;POINT TO THE START OF THE NEXT TEST
012456  013700  002010                  MOV     DZVCSR,R0       ;GET BASE ADDRESS
012462  012703  012576                  MOV     #5$,R3          ;SET R3 TO TOP OF TABLE
012466  011305                  1$:     MOV     (R3),R5         ;SET BIT
012470  012737  012476  001364          MOV     #11$,LOCK       ;SETUP FOR TIGHT SCOPE LOOP
012476  010510                  11$:    MOV     R5,(R0)         ;SET BIT IN DEVICE
012500  011004                          MOV     (R0),R4         ;READ THE BIT FROM DEVICE
012502  020504                          CMP     R5,R4           ;WAS BIT SET?
012504  001401                          BEQ     2$              ;BR IF YES
012506  104002                          ERROR   2               ;*BIT R/W FAILURE
012510  104401                  2$:     SCOP1                   ;IS SWITCH 9 SET?
012512  012737  012520  001364          MOV     #12$,LOCK       ;SET FOR NEXT TIGHT SCOPE LOOP
012520  040510                  12$:    BIC     R5,(R0)         ;CLEAR THE BIT.
012522  011004                          MOV     (R0),R4         ;READ DEVICE
012524  001403                          BEQ     3$              ;BR IF BITS WERE CLEARED.
012526  005005                          CLR     R5              ;CLEAR FOR ERROR PRINTOUT
012530  104002                          ERROR   2               ;*BIT FAILED TO CLEAR
012532  011305                          MOV     (R3),R5         ;RESTORE THE BIT.
012534  104401                  3$:     SCOP1                   ;SW09 SET?
012536  012737  012544  001364          MOV     #13$,LOCK       ;SET UP FOR NEXT TIGHT SCOPE
012544  010510                  13$:    MOV     R5,(R0)         ;SET THE BIT AGAIN
012546  104413                          DEVICE.CLR              ;ISSUE DEVICE CLEAR
012550  011004                          MOV     (R0),R4         ;READ THE BIT.
012552  001403                          BEQ     4$              ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
012554  005005                          CLR     R5              ;SET EXPECTED TO ZERO
012556  104002                          ERROR   2               ;*BIT NOT CLEARED BY DEVICE CLEAR
012560  011305                          MOV     (R3),R5         ;RESTORE BIT AGAIN
012562  104401                  4$:     SCOP1                   ;SW09 SET?
012564  062703  000002                  ADD     #2,R3           ;POP R3
012570  005713                          TST     (R3)            ;IS THIS THE END OF TABLE?
012572  001407                          BEQ     6$              ;IF YES GET OUT
012574  000734                          BR      1$              ;OTHERWISE TEST NEXT BIT
012576  000010                  5$:     #MAINT                  ;CSR BIT: INTERNAL MAINTENANCE
012600  000040                          #MSENAB                 ;CSR BIT: MASTER SCAN ENABLE
012602  010000                          #SILOEN                 ;CSR BIT: SILO ENABLE
012604  000100                          #RIE                    ;CSR BIT: RECEIVER INTER. ENABLE
012606  040000                          #TIE                    ;CSR BIT: TRANS. INTER. ENABLE
```

```
2988  012610  000000                          #0                      ;END OF TABLE
2989  012612  005037  001364          6$:     CLR     LOCK            ;ZERO LOCK INDICATOR
2990                                          ;************************ TEST 4 ******************************
2991                                          ;*THIS TESTS THAT ALL OF THE TCR BITS
2992                                          ;*CAN BE: SET, CLEARED, AND CLEARED BY A DEVICE CLEAR.
2993                                          ;*THIS TEST ALSO DETERMINES IF THE DTR BITS CAN
2994                                          ;*BE SET, CLEARED, AND CLEARED BY A RESET.
2995                                  ;:*  TEST 4
2996                                  ;;*************************************************************************
2997  012616  000004                  TST4:   SCOPE
2998  012620  012737  000004  001246          MOV     #4,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
2999  012626  012737  013022  001362          MOV     #TST5,NEXT      ;POINT TO THE START OF THE NEXT TEST
3000  012634  013700  002024                  MOV     DZVTCR,R0       ;SET DEVICE ADDRESS
3001  012640  012703  012726                  MOV     #5$,R3          ;SET R3 POINTER TO TOP OF TABLE
3002  012644  012737  012654  001364  1$:     MOV     #1,$,LOCK       ;SET LOCK FOR SW09 SCOPE LOOP
3003  012652  011305                          MOV     (R3),R5         ;SET EXPECTED RESULTS
3004  012654  010510                  11$:    MOV     R5,(R0)         ;SET THE BIT
3005  012656  011004                          MOV     (R0),R4         ;READ THE BIT FROM THE DEVICE
3006  012660  020504                          CMP     R5,R4           ;DID THE BIT SET?
3007  012662  001401                          BEQ     2$              ;BR IF YES
3008  012664  104002                          ERROR   2               ;*BIT FAILED TO SET.
3009  012666  104401                  2$:     SCOP1                   ;SW09 SET?
3010  012670  012737  012676  001364          MOV     #3$,LOCK        ;SET UP FOR NEXT TIGHT SCOPE LOOP
3011  012676  040510                  3$:     BIC     R5,(R0)         ;CLEAR THE BIT
3012  012700  011004                          MOV     (R0),R4         ;READ THE REGISTER
3013  012702  001403                          BEQ     4$              ;BR IF YES
3014  012704  005005                          CLR     R5              ;SET EXPECTED TO 0
3015  012706  104002                          ERROR   2               ;*REPORT BIT NOT CLEAR
3016  012710  011305                          MOV     (R3),R5         ;RESTORE R5
3017  012712  104401                  4$:     SCOP1                   ;SW09 SET?
3018  012714  062703  000002                  ADD     #2,R3           ;POP POINTER TO NEXT TABLE ENTRY
3019  012720  005713                          TST     (R3)            ;END OF TABLE?
3020  012722  001412                          BEQ     6$              ;IF YES JUMP OVER TABLE
3021  012724  000747                          BR      1$              ;START TESTING NEXT BIT
3022  012726  000001                  5$:     #TCR0                   ;TCR BIT FOR LINE 0
3023  012730  000002                          #TCR1                   ;TCR BIT FOR LINE 1
3024  012732  000004                          #TCR2                   ;TCR BIT FOR LINE 2
3025  012734  000010                          #TCR3                   ;TCR BIT FOR LINE 3
3026  012736  000400                          #DTR0                   ;DTR BIT FOR LINE 0
3027  012740  001000                          #DTR1                   ;DTR BIT FOR LINE 1
3028  012742  002000                          #DTR2                   ;DTR BIT FOR LINE 2
3029  012744  004000                          #DTR3                   ;DTR BIT FOR LINE 3
3030  012746  000000                          #0                      ;END OF TABLE
3031  012750  005037  001364          6$:     CLR     LOCK            ;CLEAR TIGHT SCOPE LOOP INDIC.
3032  012754  012710  177777                  MOV     #-1,(R0)        ;SET ALL BITS IN TCR REGISTER
3033  012760  012705  007400                  MOV     #007400,R5      ;SET EXPECTED
3034  012764  104413                          DEVICE.CLR              ;SET DCLR BIT IN CSR
3035  012766  011004                          MOV     (R0),R4         ;READ REGISTER
3036  012770  020504                          CMP     R5,R4           ;TCR BITS CLEARED?
3037  012772  001401                          BEQ     7$              ;IF YES BRANCH
3038  012774  104002                          ERROR   2               ;TCR BITS NOT CLEARED!
3039  012776  005005                  7$:     CLR     R5              ;SET FXPECTED TO ZERO
3040  013000  005227  000000          8$:     INC     #0              ;DELAY FOR ACT
3041  013004  001375                          BNE     8$              ;
3042  013006  012710  177777                  MOV     #-1,(R0)        ;SET ALL POSSIBLE BITS
3043  013012  000005                          RESET                   ;DO BUS INIT
```

```
3044  013014  011004                         MOV     (R0),R4         ;DID REGISTER CLEAR?
3045  013016  001401                         BEQ     9$              ;IF YES GET OUT
3046  013020  104002                         ERROR   2               ;REGISTER DID NOT CLEAR!
3047  013022                          9$:
3048                                          ;************************ TEST 5 ****************************
3049                                          ;*THIS TEST VERIFIES THAT
3050                                          ;*BITS 'RDONE,TRDY, BIT9, BIT8,
3051                                          ;*AND SILOAL'' ARE READ ONLY AND THAT TRDY IS
3052                                          ;*ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.
3053                                          ;*
3054                                  ;::*  TEST 5
3055                                  ;;*************************************************************
3056  013022  000004                 TST5:   SCOPE
3057  013024  012737  000005  001246         MOV     #5,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
3058  013032  012737  013124  001362         MOV     #TST6,NEXT      ;POINT TO THE START OF THE NEXT TEST
3059  013040  013700  002010                 MOV     DZVCSR,R0       ;SET ADDRESS TO R0
3060  013044  104413                         DEVICE.CLR              ;DO A DEVICE CLEAR
3061  013046  005005                         CLR     R5              ;SET EXPECTED TO 0
3062  013050  012710  121600                 MOV     #RDONE+TRDY+BIT9+BIT8+SILOAL,(R0)
3063                                                                  ;WRITE THE BITS
3064  013054  011004                         MOV     (R0),R4         ;READ BACK THE BITS
3065  013056  001401                         BEQ     2$              ;BR IF NONE ARE SET.
3066  013060  104002                         ERROR   2               ;*BITS WERE SET.
3067  013062  012705  100040          2$:    MOV     #TRDY+MSENAB,R5 ;SET EXPECTED BIT
3068  013066  052777  000017  166730         BIS     #17,@DZVTCR     ;SET TCR BITS FOR ALL LINES
3069  013074  052710  000040                 BIS     #MSENAB,(R0)    ;SET SCAN ENABLE
3070  013100  005002                         CLR     R2              ;SET COUNTER TO ZERO
3071  013102  011004                  3$:    MOV     (R0),R4         ;READ THE REGISTER
3072  013104  042704  001400                 BIC     #BIT9!BIT8,R4   ;MASK OUT LINE NO.
3073  013110  020504                         CMP     R5,R4           ;BIT SET?
3074  013112  001404                         BEQ     4$              ;BR IF YES
3075  013114  104414                         DELAY                   ;STALL TIME
3076  013116  005202                         INC     R2              ;UPDATE COUNTER
3077  013120  001370                         BNE     3$              ;BR IF COUNTER NOT DONE.
3078  013122  104002                         ERROR   2               ;*TRDY NOT SET!
3079  013124                          4$:
3080                                          ;************************ TEST 6 ****************************
3081                                          ;*THIS TEST VERIFIES THAT:
3082                                          ;*TIE,SILOEN,RIE,MSENAB,AND MAINT ARE THE
3083                                          ;*ONLY R/W BITS IN THE DZVCSR AND THAT
3084                                          ;*SETTING 'DCLR'' IN THE CSR WILL CLEAR THESE BITS.
3085                                  ;::*  TEST 6
3086                                  ;;*************************************************************
3087  013124  000004                 TST6:   SCOPE
3088  013126  012737  000006  001246         MOV     #6,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
3089  013134  012737  013254  001362         MOV     #TST7,NEXT      ;POINT TO THE START OF THE NEXT TEST
3090  013142  104413                         DEVICE.CLR              ;SET DCLR IN CSR
3091  013144  013700  002010                 MOV     DZVCSR,R0       ;SET UP FOR ERROR MESSAGE
3092  013150  012710  177757                 MOV     #^C<DCLR>,(R0)  ;TRY TO SET ALL BITS EXCEPT DCLR
3093  013154  012705  050150                 MOV     #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
3094  013160  011004                         MOV     (R0),R4         ;ACTUAL
3095  013162  020405                         CMP     R4,R5           ;CMP EXPECTED VS ACTUAL
3096  013164  001401                         BEQ     1$              ;YES
3097  013166  104002                         ERROR   2               ;*NO
3098  013170  105010                  1$:    CLRB    (R0)            ;CLEAR LOW BYTE OF CSR
3099  013172  105005                         CLRB    R5              ;CLEAR LOW BYTE OF EXPECTED DATA
```

```
3100  013174  011004                        MOV     (R0),R4         ;READ CSR
3101  013176  020405                        CMP     R4,R5           ;DOES CSR COMPARE WITH EXPECTED?
3102  013200  001401                        BEQ     3$              ;BRANCH IF YES
3103  013202  104002                        ERROR   2               ;IF NOT PRINT ERROR
3104  013204  012710  177757          3$:   MOV     #^C<DCLR>,(R0)  ;SET ALL CSR BITS POSSIBLE
3105  013210  105077  166576                CLRB    @#DZVCSR        ;CLEAR HIGH BYTE OF CSR
3106  013214  012705  000150                MOV     #RIE!MSENAB!MAINT,R5    ;SET EXPECTED IN R5
3107  013220  011004                        MOV     (R0),R4         ;READ CSR REGISTER
3108  013222  020405                        CMP     R4,R5           ;DOES ACTUAL=EXPECTED
3109  013224  001401                        BEQ     4$              ;IF YES CONTINUE
3110  013226  104002                        ERROR   2               ;IF NO PRINT ERROR
3111  013230  012710  177757          4$:   MOV     #^C<DCLR>,(R0)  ;SET ALL POSSIBLE CSR BITS
3112  013234  005005                        CLR     R5              ;SET R5 TO EXPECTED RESULTS
3113  013236  052710  000020                BIS     #DCLR,(R0       ;DEVICE MASTER RESET
3114  013242  000240                        NOP
3115  013244  011004                        MOV     (R0),R4         ;ACTUAL
3116  013246  020405                        CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
3117  013250  001401                        BEQ     2$              ;YES
3118  013252  104002                        ERROR   2               ;*NO
3119  013254                          2$
3120                                        ;********************** TEST 7 ****************************
3121                                        ;*THIS TEST PERFORMS RESET TESTING AND
3122                                        ;*TESTING OF READ ONLY REGISTER DZVRBUF
3123                                        ;*AND TESTING OF WRITE ONLY REGISTER DZVLPR
3124                                  ;;*  TEST 7
3125                                  ;;********************** ****************************************
3126  013254  000004                  TST7: SCOPE
3127  013256  012737  000007  001246        MOV     #7,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
3128  013264  012737  013340  001362        MOV     #TST10,NEXT     ;POINT TO THE START OF THE NEXT TEST
3129  013272  104413                        DEVICE.CLR              ;CLEAR DZV11
3130  013274  013700  002014                MOV     DZVRBUF,R0      ;SET UP FOR ERROR MESSAGE
3131  013300  011005                        MOV     (R0),R5         ;COPY PRESENT CONTENTS
3132  013302  042705  106000                BIC     #DVALID'BIT11!BIT10,R5 ;CLEAR ILLEGAL BITS
3133  013306  012777  177777  166504        MOV     #-1,@DZVLPR     ;TRY TO WRITE ALL 1'S
3134  013314  011004                        MOV     (R0),R4         ;ACTUAL
3135  013316  020405                        CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
3136  013320  001401                        BEQ     1$              ;IF YES,GO CONTINUE PROCESSING
3137  013322  104002                        ERROR   2               ;*ERROR- BIT PATTERN NOT CORRECT
3138  013324  005077  166470          1$:   CLR     @DZVLPR ;TRY TO WRITE ALL ZEROES
3139  013330  011004                        MOV     (R0),R4         ;READ REGISTER
3140  013332  020405                        CMP     R4,R5           ;CMP ACTUAL VS. EXPECTED
3141  013334  001401                        BEQ     2$              ;BRANCH IF EQUAL
3142  013336  104002                        ERROR   2               ;VALUES DID NOT COMPARE
3143  013340                          2$:
3144                                        ;********************** TEST 10 ****************************
3145                                        ;*THIS TEST PERFORMS RESET TESTING AND
3146                                        ;*TESTING OF READ ONLY REGISTER DZVMSR
3147                                        ;*AND TESTING OF WRITE ONLY REGISTER DZVTDR
3148                                  ;;*  TEST 10
3149                                  ;;************************************************************
3150  013340  000004                  TST10: SCOPE
3151  013342  012737  000010  001246        MOV     #10,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3152  013350  012737  013424  001362        MOV     #TST11,NEXT     ;POINT TO THE START OF THE NEXT TEST
3153  013356  104413                        DEVICE.CLR              ;CLEAR DZV11
3154  013360  013700  002030                MOV     DZVMSR,R0       ;SET UP FOR ERROR MESSAGE
3155  013364  011005                        MOV     (R0),R5         ;COPY PRESENT CONTENTS
```

```
3156  013366  062705  170360                BIC     #170360,@S      ;CLEAR ILLEGAL BITS
3157  013372  112777  177777  166434        MOVB    #-1,@DZV DR     ;TRY TO WRITE ALL 1'S
3158  013400  011004                       MOV     (R0),R4         ;ACTUAL
3159  013402  020405                       CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
3160  013404  001401                       BEQ     1$              ;IF YES,GO CONTINUE PROCESSING
3161  013406  104002                       ERROR   2               ;*ERROR- BIT PATTERN NOT CORRECT
3162  013410  005077  166420        1$:    CLR     @DZVTDR  TRY TO WRITE ALL ZEROES
3163  013414  011004                       MOV     (R0),R4         ;READ REGISTER
3164  013416  020405                       CMP     R4,R5           ;CMP ACTUAL VS. EXPECTED
3165  013420  001401                       BEQ     2$              ;BRANCH IF EQUAL
3166  013422  104002                       ERROR   2               ;VALUES DID NOT COMPARE
3167  013424                        2$:
3168
3169                                        ;********************** TEST 11 ****************************
3170                                        ;*VERIFY THAT SETTING 'DTR' FOR A LINE WILL
3171                                        ;*BRING UP 'CO' AND 'RING' FOR:
3172                                        ;*THE SAME LINE IF IN EXTERNAL MODE
3173                                        ;*THE STAGGERED LINE IF IN STAGGERED MODE.
3174                                        ;*LINES ARE STAGGERED AS FOLLOWS:
3175                                        ;*LINE0 WITH LINE1; LINE2 WITH LINE3
3176                                        ;*THIS TEST IS ONLY RUN IF AN H325 OR H329
3177                                        ;*IS CONNECTED ON THE DZV UNDER TEST.
3178
3179                                ;;*  TEST 11
3180                                ;;*************************************************************
3181  013424  000004                TST11:  SCOPE
3182  013426  012737  000011  001246       MOV     #11,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3183  013434  012737  013620  001362       MOV     #TST12,NEXT     ;POINT TO THE START OF THE NEXT TEST
3184  013442  005737  001372               TST     MODE            ;TEST TO SEE IF TESTING WITH
3185  013446  001001                       BNE     8$              ;CONNECTOR
3186  013450  104400                       ADVANCE                 ;IF NO, GO TO NEXT TEST
3187  013452  012737  013542  001364  8$:  MOV     #10$,LOCK       ;SET FOR TIGHT SCOPE LOOP
3188  013460  104413                       DEVICE.CLR              ;SET DCLR IN CSR TO ZERO DEVICE
3189  013462  013700  002030               MOV     DZVMSR,R0       ;SET REGISTER
3190  013466  005003                       CLR     R3              ;ZERO LINE NUMBER
3191  013470  012702  000001               MOV     #1,R2           ;SET POINTER
3192  013474  130237  001366        1$:    BITB    R2,LINE         ;TEST THIS LINE?
3193  013500  001003                       BNE     3$              ;YES
3194  013502  005203                2$:    INC     R3              ;LINE #
3195  013504  104420                       SHIFT                   ;GET NEXT LINE
3196  013506  000772                       BR      1$              ;TEST NEXT LINE
3197  013510  010204                3$:    MOV     R2,R4           ;SAVE BINARY BIT FOR LINE #
3198  013512  105737  001372               TSTB    MODE            ;RUNNING IN EXTERNAL MODE?
3199  013516  100406                       BMI     5$              ;IF YES SKIP STAGGERED SETUP
3200  013520  032703  000001               BIT     #BIT0,R3        ;IF EVEN LINE
3201  013524  001402                       BEQ     4$              ;GO GET ODD PARTNER
3202  013526  006204                       ASR     R4              ;OTHERWISE GET EVEN COMPANION
3203  013530  000401                       BR      5$              ;GO SETUP EXPECTED RESULTS
3204  013532  006304                4$:    ASL     R4              ;FIND ODD PARTNER
3205  013534  010405                5$:    MOV     R4,R5           ;LOAD R5 FOR EXPECTED
3206  013536  000305                       SWAB    R5              ;PLACE IN UPPER BYTE
3207  013540  150405                       BISB    R4,R5           ;SET FOR RING BITS
3208  013542  150277  166260        10$:   BISB    R2,@#DZVTCR     ;SET DTR BIT
3209  013546  104414                       DELAY                   ;DELAY FOR CABLE LAG
3210  013550  011004                       MOV     (R0),R4         ;MOVE RESULTS OF MSR REGISTER TO R4
3211  013552  020504                       CMP     R5,R4           ;RESULTS=EXPECTED?
```

```
3212  013554  001401                          BEQ     6$              ;IF YES CONTINUE
3213  013556  104002                          ERROR   2               ;IF NOT PRINT ERROR RESULTS
3214  013560  104401                  6$:     SCOP1                   ;IS SW09 SET?
3215  013562  012737  013570  001364          MOV     #11$,LOCK       ;SET UP FOR NEXT TIGHT SCOPE
3216  013570  140277  166232          11$:    BICB    R2,@#DZVTCR     ;CLEAR  DTR BIT FOR LINE UNDER TEST
3217  013574  104414                          DELAY                   ;DELAY FOR CABLE LAG
3218  013576  011004                          MOV     (R0),R4         ;LOAD MSR REGISTER INTO R4
3219  013600  001402                          BEQ     7$              ;IF CO AND RING CLEARED CONTINUE
3220  013602  005005                          CLR     R5              ;OTHERWISE SET EXPECTED FOR ERROR
3221  013604  104002                          ERROR   2               ;PRINTOUT
3222  013606  104401                  7$:     SCOP1                   ;IS SW09 SET?
3223  013610  012737  013542  001364          MOV     #10$,LOCK       ;RESET TIGHT SCOPE LOOP
3224  013616  000731                          BR      2$              ;GET NEXT LINE
3225
3226                                          ;********************** TEST 12 **********************
3227                                          ;* THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
3228                                          ;* IS READY TO BE LOADED, AND THAT THE LINE SPECI-
3229                                          ;* FIED IN BITS 8-9 OF DZVCSR CORRESPOND
3230                                          ;* TO THE LINE SELECTED IN DZVTCR
3231                                  ;;*  TEST 12
3232                                  ;;*****************************************************************
3233  013620  000004                  TST12:  SCOPE
3234  013622  012737  000012  001246          MOV     #12,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3235  013630  012737  013752  001362          MOV     #TST13,NEXT     ;POINT TO THE START OF THE NEXT TEST
3236  013636  104413                          DEVICE.CLR              ;ISSUE A 'DEVICE CLEAR' (RESET)
3237  013640  012737  013674  001364          MOV     #2$,LOCK        ;SET UP FOR TIGHT SCOPE LOOP
3238  013646  005037  001374                  CLR     SAVLIN          ;INITIALIZE FOR ERROR PRINTOUT
3239  013652  013700  002010                  MOV     DZVCSR,R0       ;SET POINTER
3240  013656  012705  100040                  MOV     #MSENAB!TRDY,R5 ;START THE EXPECTED LINE NUMBER AT 0
3241  013662  012702  000001                  MOV     #1,R2           ;USING R2 AS A BIT POINTER, POINT TO LINE 0
3242  013666  130237  001366          1$:     BITB    R2,LINE         ;IS THIS LINE SELECTED?
3243  013672  001421                          BEQ     6$              ;IF NO, SKIP THE STARTUP
3244  013674  050277  166124          2$:     BIS     R2,@#DZVTCR     ;SET THE GO BIT FOR THIS LINE
3245  013700  052710  000040                  BIS     #MSENAB,(R0)    ;START THE SCANNER
3246  013704  005004                          CLR     R4              ;SET FOR DELAY
3247  013706  005710                  3$:     TST     (R0)            ;TX READY?
3248  013710  100404                          BMI     4$              ;BR IF YES
3249  013712  104414                          DELAY                   ;DELAY
3250  013714  005204                          INC     R4              ;COUNTER
3251  013716  001373                          BNE     3$              ;BR IF <>0!
3252  013720  104003                          ERROR   3               ;*TX NOT READY!
3253  013722  011004                  4$:     MOV     (R0),R4         ;GET THE LINE POINTED TO BY THE SCANNER
3254  013724  020405                          CMP     R4,R5           ;IS THE LINE NUMBER WHAT IT SHOULD BE?
3255  013726  001401                          BEQ     5$              ;IF YES,GO WORK ON THE NEXT LINE
3256  013730  104002                          ERROR   2               ;*LINE NUMBER DID NOT MATCH TCR BIT
3257  013732  104401                  5$:     SCOP1                   ;IS SW09 SET?
3258  013734  104413                          DEVICE.CLR              ;SET DCLR IN CSR;SETUP FOR NEXT LINE
3259  013736  062705  000400          6$:     ADD     #400,R5         ;POINT TO THE NEXT EXPECTED LINE
3260  013742  104420                          SHIFT                   ;POINT TO THE NEXT LINE.ARE ALL LINES TESTED?
3261  013744  005237  001374                  INC     SAVLIN          ;ADJUST FOR ERROR PRINTOUT
3262  013750  000746                          BR      1$              ;IF NOT, GO DO THE NEXT LINE
3263                                          ;********************** TEST 13 **********************
3264                                          ;*TEST TO TRANSMIT ONE CHAR AND
3265                                          ;*RECEIVE ONE CHAR ON ONE LINE
3266                                          ;*AT A TIME. THE CHAR IS '252' AND
3267                                          ;*ALL SELECTED LINES WILL BE TURNED ON .
```

```
                                        ;*THIS IS THE FIRST TIME ANY
                                        ;*DATA IS CHECKED IN THE RECEIVER.
                                        ;*USING SWITCH NINE WITH THIS TEST CREATES A TIGHT SCOPE LOOP
                                        ;*WHICH TRANSMITS A STEADY STREAM OF CHARACTERS.
                                ;:*  TEST 13
                                ;*************************************************************
013752  000004                TST13:  SCOPE
013754  012737  000013  001246        MOV     #13,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
013762  012737  014242  001362        MOV     #TST14,NEXT     ;POINT TO THE START OF THE NEXT TEST
013770  012737  014224  001364        MOV     #16$,LOCK       ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
013776  104417                        DCLASM                  ;SET DCLR IN CSR AND SET MAINT MODE
014000  104421                        LPRSET                  ;LOAD LPR REGISTER FOR ALL LINES
014002  005037  001374                CLR     SAVLIN          ;INIT. FOR ERROR PRINTOUT
014006  105037  001425                CLRB    DONFLG          ;INIT FOR TCR BIT HANDLER
014012  012702  000001                MOV     #1,R2           ;LINE POINTER
014016  012701  000252                MOV     #252,R1         ;SAVE CHARACTER TO BE TRANSMITTED
014022  052777  000040  165760        BIS     #MSENAB,@DZVCSR ;START SCANNER
014030  030237  001306        3$:     BIT     R2,LINE         ;VALID LINE ?
014034  001467                        BEQ     15$             ;NO SET UP NEXT LINE
014036  010277  165762                MOV     R2,@DZVTCR      ;SET TCR BIT
014042  005005                5$:     CLR     R5              ;SET R5 FOR A DELAY LOOP
014044  105777  165740                TSTB    @DZVCSR         ;IS REC DONE = 0 ?
014050  100001                        BPL     6$              ;IF YES, ALLOW TIME FOR TRDY TO SET
014052  104020                        ERROR   20              ;*REC DONE SHOULD = 0
014054  005777  165730        6$:     TST     @DZVCSR         ;TRDY SET?
014060  100404                        BMI     7$              ;IF YES BRANCH
014062  104414                        DELAY                   ;IF NO THEN WAIT FOR IT
014064  005205                        INC     R5              ;DELAY LOOP
014066  001372                        BNE     6$              ;BRANCH BACK AND TEST AGAIN
014070  104003                        ERROR   3               ;*TRDY FAILED TO SET!
014072  105737  001425        7$:     TSTB    DONFLG          ;HAVE WE ALREADY SENT CHARAC.
014076  001041                        BNE     13$             ;IF YES GO CLEAR TCR BIT
014100  105237  001425                INCB    DONFLG          ;IF NOT INDICATE HAVING BEEN HERE
014104  110177  165724                MOVB    R1,@DZVTDR      ;LOAD CHARACTER
014110  013705  001374                MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
014114  005737  001372                TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
014120  100006                        BPL     10$             ;IF NOT, SKIP STAGGERED SETUP

                                        ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER

014122  006205                        ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
014124  103402                        BCS     8$              ;IF IT IS SET, GO CLEAR IT
014126  000261                        SEC                     ;IF IT IS CLEAR SET IT HERE
014130  000401                        BR      9$              ;SKIP THE CLEARING
014132  000241                8$:     CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
014134  006105                9$:     ROL     R5              ;GET THE NEW BIT BACK INTO R5
014136  000305                10$.    SWAB    R5              ;MOVE THE LINE NUMBER TO THE UPPER BYTE
014140  150105                        BISB    R1,R5           ;ADD CHARACTER
014142  052705  100000                BIS     #DVALID,R5      ;ADD DATA VALID
014146  005003                        CLR     R3
014150  105777  165634        11$:    TSTB    @DZVCSR         ;IS RDONE SET?
014154  100404                        BMI     12$             ;IF YES GO GET CHAR.
014156  104414                        DELAY                   ;IF NOT THEN WAIT
014160  005203                        INC     R3              ;DELAY LOOP
014162  001372                        BNE     11$             ;DELAY DONE?
014164  104004                        ERROR   4               ;*RDONE FAILED TO SET
```

```
3324  014166  017704  165622          12$:    MOV     @DZVRBUF,R4     ;LOAD THE VALUE ACTUALLY RECEIVED
3325  014172  020405                          CMP     R4,R5           ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
3326  014174  001722                          BEQ     5$              ;IF YES, GO DO THE NEXT LINE
3327  014176  104006                          ERROR   6               ;*NO DATA/CONTENTS DID NOT COMPARE
3328  014200  000720                          BR      5$              ;GO BACK AND WAIT TO CLEAR TCR BIT
3329  014202  104401                  13$:    SCOPE                   ;CHECK TO SEE IF SWITCH NINE IS SET
3330  014204  105037  001425                  CLRB    DONFLG          ;SET UP FOR NEXT LINE
3331  014210  005077  165610                  CLR     @DZVTCR         ;CLEAR PREVIOUS TCR BIT
3332  014214  005237  001374          15$:    INC     SAVLIN          ;SET LINE INDICATOR FOR NEXT LINE
3333  014220  104420                          SHIFT                   ;CALCULATE NEXT LINE
3334  014222  000702                          BR      3$              ;GET GET STARTED
3335
3336                                  ;TIGHT SCOPE LOOP FOR THIS TEST. LOOP TRANSMITS CHARACTERS ONLY
3337
3338  014224  005777  165560          16$:    TST     @DZVCSR         ;IS TRANSMITTER READY?
3339  014230  100375                          BPL     16$             ;IF NOT, WAIT FOR IT
3340  014232  110177  165576                  MOVB    R1,@DZVTDR      ;LOAD THE CHARACTER
3341  014236  104401                          SCOPE                   ;LOOP AGIN IF SW09=1
3342  014240  000760                          BR      13$             ;OTHERWISE, GO PICK UP THE TEST NORMALLY
3343
3344                                          ;********************** TEST 14 ****************************
3345                                          ;*THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
3346                                          ;*DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
3347                                          ;*TO ZERO FOR EACH LINE.
3348                                          ;*THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
3349                                          ;*EMPTIED BY ISSUING A DEVICE MASTER CLEAR.
3350                                  ;;*  TEST 14
3351                                  ;;*************************************************************
3352  014242  000004                  TST14:  SCOPE
3353  014244  012737  000014  001246          MOV     #14,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3354  014252  012737  014564  001362          MOV     #TST15,NEXT     ;POINT TO THE START OF THE NEXT TEST
3355  014260  105037  001425                  CLRB    DONFLG          ;CLEAR TEST CONTROL FLAG
3356  014264  005037  001374                  CLR     SAVLIN          ;CLEAR LINE INDICATOR
3357  014270  104417                          DCLASM                  ;ISSUE A DEVICE MASTER CLEAR
3358                                                                  ;AND SET MAINT BIT IF NECESSARY
3359  014272  013701  001370                  MOV     PAR,R1          ;SAVE DEFAULT PARAMETERS
3360  014276  042737  010000  001370          BIC     #RCVON,PAR      ;DISABLE RECEIVER IN DEFAULT PAR.
3361  014304  104421                  100$:   LPRSET                  ;LOAD PARAMETERS IN LPR REGISTER
3362  014306  010137  001370                  MOV     R1,PAR          ;RESTORE DEFAULT PARAMETERS
3363  014312  012701  000252                  MOV     #252,R1         ;LOAD A CHARAC. INTO R1
3364  014316  013702  001366                  MOV     LINE,R2         ;COPY AN IMAGE OF THE ACTIVE LINES
3365  014322  010277  165476                  MOV     R2,@DZVTCR      ;SET TCR BITS FOR ALL ACTIVE LINES
3366  014326  052777  000040  165454          BIS     #MSENAB,@DZVCSR ;SET MASTER SCAN ENABLE
3367  014334  005005                  1$:     CLR     R5              ;INIT DELAY COUNTER
3368  014336  005777  165446          2$:     TST     @DZVCSR         ;IS TRANS READY SET?
3369  014342  100404                          BMI     3$              ;BRANCH IF YES
3370  014344  104414                          DELAY                   ;WAIT FOR TRDY TO SET
3371  014346  005205                          INC     R5              ;INCREMENT DELAY COUNTER
3372  014350  001372                          BNE     2$              ;RETURN TO CHECK TRDY
3373  014352  104003                          ERROR   3               ;TRDY FAILED TO SET!
3374  014354  117705  165432          3$:     MOVB    @#DZVCSR,R5     ;MOVE LINE NO. TO R5
3375  014360  012703  000001                  MOV     #1,R3           ;INIT TCR POINTER
3376  014364  042705  177774                  BIC     #^C<3>,R5       ;ISOLATE LINE NO.
3377  014370  001403                          BEQ     31$             ;IF LINE 0 BRANCH
3378  014372  106303                  30$:    ASLB    R3              ;SHIFT R3 POINTER TO NEXT LINE
3379  014374  005305                          DEC     R5              ;DECREMENT LINE NO.
```

```
3380  014376  001375                  BNE     30$             ;WHEN R5=0, R3 POINTS TO LINE TCR
3381  014400  030302          31$     BIT     R3,R2           ;HAS CHARACTER BEEN SENT?
3382  014402  001007                  BNE     4$              ;BRANCH IF NO
3383  014404  140377  165414          BICB    R3,@DZVTCR      ;IF YES THEN CLEAR TCR BIT
3384  014410  001351                  BNE     1$              ;IF ALL CHARAC. SENT DROP THROUGH
3385  014412  105737  001425          TSTB    DONFLG          ;IF NO MORE ACTIVE IS THIS SECOND
3386                                                          ;TIME HERE?
3387  014416  001037                  BNE     10$             ;IF YES SKIP TO SECOND PART OF TEST
3388  014420  000404                  BR      5$              ;IF FIRST TIME HERE GO ZERO TCR BITS
3389  014422  110177  165406    4$    MOVB    R1,@DZVTDR      ;LOAD CHAR. INTO BUFFER
3390  014426  040302                  BIC     R3,R2           ;INDICATE CHARAC. SENT ON THIS LINE
3391  014430  000741                  BR      1$              ;GO BACK AND WAIT FOR TRDY TO SET
3392  014432  005077  165366    5$:   CLR     @DZVTCR         ;CLEAR OUT TCR BITS
3393  014436  005005                  CLR     R5              ;INIT DELAY COUNTER
3394  014440  105777  165344    6$:   TSTB    @DZVCSR         ;IS RECEIV. DONE SET?
3395  014444  100002                  BPL     7$              ;IF NOT THEN WAIT TO SEE IF IT WILL
3396  014446  104020                  ERROR   20              ;REC DONE SHOULD NOT SET!
3397  014450  000403                  BR      8$              ;GO FIND WHICH LINE RECEIVED
3398  014452  104414          7$:     DELAY                   ;STALL FOR RECEIVER
3399  014454  005205                  INC     R5              ;INCREMENT DELAY COUNTER
3400  014456  001370                  BNE     6$              ;IF NOT DONE GO RETEST REC DONE
3401  014460  017704  165330    8$.   MOV     @DZVRBUF,R4     .READ REC. BUFFER
3402  014464  100007                  BPL     9$               IS DVALID SET?
3403  014466  000304                  SWAB    R4              .IF YES GET LINE NO.
3404  014470  042704  177774          BIC     #^C<3>,R4       ;ISOLATE LINE NO.
3405  014474  010437  001374          MOV     R4,SAVLIN       ;SET UP LINE NO. FOR ERROR REPORT
3406  014500  104017                  ERROR   17              ;DVALID SHOULD NOT BE SET
3407  014502  000766                  BR      8$              ;GO CHECK FOR ANY OTHER CHAR. IN SILO
3408  014504  105237  001425    9$:   INCB    DONFLG          ;INDICATE THAT FIRST PART OF TEST IS DONE
3409  014510  013701  001370          MOV     PAR,R1          ;SAVE DEFAULT LINE PARAM.
3410  014514  000673                  BR      100$            ;NOW GO RELOAD LPR REGISTER TO
3411                                                          ;TURN RECEIVERS ON
3412  014516  005005          10$:    CLR     R5              ;ZERO DELAY COUNTER
3413  014520  104414          11$:    DELAY                   ;WAIT FOR ALL CHARAC. TO BE RECEIVED
3414  014522  005205                  INC     R5              ;INCREASE DELAY COUNT
3415  014524  001375                  BNE     11$             ;CONT. DELAY IF NOT FINISHED
3416  014526  104413                  DEVICE.CLR              ;ISSUE A MASTER CLEAR
3417  014530  000240                  NOP
3418  014532  000240                  NOP
3419  014534  105777  165250          TSTB    @DZVCSR         ;NOW IS RECEIV. DONE SET?
3420  014540  100003                  BPL     12$             ;BRANCH IF NO
3421  014542  005037  001374          CLR     SAVLIN          ;CLEAR LINE NO FOR ERROR REPORT
3422  014546  104020                  ERROR   20              ;REC. DONE SHOULD NOT BE SET!
3423  014550  017704  165240    12$:  MOV     @DZVRBUF,R4     ;READ REC. BUFFER
3424  014554  100003                  BPL     13$             ;IS DVALID SET? IT SHOULDN'T BE
3425  014556  005037  001374          CLR     SAVLIN          ;DEVICE. CLR DID NOT ZERO SILO
3426  014562  104017                  ERROR   17              ;PRINT OUT THE ERROR.(LINE NO. IS IRRELEVANT)
3427  014564                  13$:
3428
3429                                  ;*************************** TEST 15 ******************************
3430                                  ;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
3431                                  ;*CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
3432                                  ;*(ONE LINE AT A TIME   BASED UPON VALID LINES)
3433                                  ;*THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
3434                          .;*  TEST 15
3435                          .;*************************************************************
```

```
3436  014564  000004                    TST15:  SCOPE
3437  014566  012737  000015  001246            MOV     #15,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3438  014574  012737  015054  001362            MOV     #TST16,NEXT     ;POINT TO THE START OF THE NEXT TEST
3439  014602  012737  014670  001364            MOV     #5$,LOCK        ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
3440  014610  104417                            DCLASM                  ;SET DCLR AND SET PNTFLG
3441  014612  104421                            LPRSET                  ;LOAD LPR REGISTER FOR ALL LINES
3442  014614  005037  001374                    CLR     SAVLIN          ;INIT FOR FIRST LINE
3443  014620  104422                            BUFSET                  ;ZERO BUFFER AREA
3444  014622  105037  001425                    CLRB    DONFLG          ;ZERO  CR BIT HANDLER FLAG
3445  014626  012702  000001                    MOV     #1,R2           ;LINE POINTER
3446  014632  052777  000040  165150            BIS     #MSENAB,@DZVCSR ;START SCANNER
3447  014640  030237  001366          3$:       BIT     R2,LINE         ;VALID LINE ?
3448  014644  001477                            BEQ     15$             ;NO SET UP NEXT LINE
3449  014646  010277  165152                    MOV     R2,@DZVTCR      ;SET TCR BIT
3450  014652  013700  001374                    MOV     SAVLIN,R0       ;ADJUST BUFFER POINTER
3451  014656  006300                            ASL     R0              ;OFFSET
3452  014660  105777  165124          4$.       TSTB    @DZVCSR         ;IS REC DONE = 0 ?
3453  014664  100001                            BPL     5$              ;IF YES, ALLOW TIME FOR TRDY TO SET
3454  014666  104020                            ERROR   20              ;*REC DONE SHOULD = 0
3455  014670  005005                    5$      CLR     R5              ;USE R5 AS TIMER WAITING FOR TRDY TO SET
3456  014672  005777  165112          6$:       TST     @DZVCSR         ;IS THE TRANSMITTER READY?
3457  014676  100404                            BMI     7$              ;IF SO, GO TRANSMIT A CHARACTER
3458  014700  104414                            DELAY                   ;WAIT A LITTLE BIT
3459  014702  005205                            INC     R5              ;UP THE LOCAL COUNTER.TIME EXCEEDED?
3460  014704  001372                            BNE     6$              ;IF NOT, GO TRY AGAIN
3461  014706  104003                            ERROR   3               ;*TRDY FAILED TO SET!
3462  014710  105737  001425          7$.       TSTB    DONFLG          ;ALL CHARAC. TRANS.?
3463  014714  001047                            BNE     14$             ;IF YES GO ZERO TCR BIT
3464  014716  116077  001426  165110            MOVB    TDO(R0),@DZVTDR ;LOAD CHARACTER
3465  014724  013705  001374                    MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
3466  014730  005737  001372                    TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
3467  014734  100006                            BPL     10$             ;IF NOT, SKIP STAGGERED SETUP
3468
3469                                            ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3470
3471  014736  006205                            ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
3472  014740  103402                            BCS     8$              ;IF IT IS SET, GO CLEAR IT
3473  014742  000261                            SEC                     ;IF IT IS CLEAR SET IT HERE
3474  014744  000401                            BR      9$              ;SKIP THE CLEARING
3475  014746  000241                    8$:     CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3476  014750  006105                    9$:     ROL     R5              ;GET THE NEW BIT BACK INTO R5
3477  014752  000305                    10$:    SWAB    R5              ;MOVE THE LINE NUMBER TO THE UPPER BYTE
3478  014754  156005  001426                    BISB    TDO(R0),R5      ;ADD CHARACTER
3479  014760  052705  100000                    BIS     #DVALID,R5      ;ADD DATA VALID
3480  014764  005003                            CLR     R3
3481  014766  105777  165016          11$:      TSTB    @DZVCSR         ;REC DONE?
3482  014772  100404                            BMI     12$             ;IF YES GO CHECK CHAR.
3483  014774  104414                            DELAY                   ;IF NOT WAIT FOR REC.
3484  014776  005203                            INC     R3              ;DELAY LOOP TIMER
3485  015000  001372                            BNE     11$             ;DELAY FINISHED?
3486  015002  104004                            ERROR   4               ;*RDONE FAILED TO SET!
3487  015004  017704  165004          12$:      MOV     @DZVRBUF,R4     ;LOAD THE VALUE ACTUALLY RECEIVED
3488  015010  020405                            CMP     R4,R5           ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
3489  015012  001401                            BEQ     13$             ;IF YES, GO DO THE NEXT LINE
3490  015014  104006                            ERROR   6               ;*NO DATA/CONTENTS DID NOT COMPARE
3491  015016  104401                    13$:    SCOP1                   ;CHECK TO SEE IF SWITCH NINE IS SET
```

```
015020  105260  001426                   INCB    TDO(R0)            ;INCREMENT BINARY PATTERN FOR THIS LINE
015024  001315                           BNE     4$                 ;GO 'ROUND AGAIN FOR NEXT CHARACTER
015026  105237  001425                   INCB    DONFLG             ;INDICATE ALL CHAR. SENT
015032  000712                           BR      4$                 ;BRANCH TO CLEAR TCR BIT
015034  005077  164764           14$:    CLR     @DZVTCR            ;CLEAR TCR REGISTER
015040  105037  001425                   CLRB    DONFLG             ;INIT FOR NEXT LINE
015044  005237  001374           15$:    INC     SAVLIN             ;INC EXPECTED LINE
015050  104420                           SHIFT                      ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
015052  000672                           BR      3$                 ;IF NO, GO AROUND AGAIN FOR NEXT LINE


                                         ;************************ TEST 16 ******************************
                                         ;*THIS TEST WILL PROVE THAT:
                                         ;* 1) THE TRANSMITTER 'BREAK BIT' WORKS
                                         ;* 2) THE RECEIVER CAN FLAG 'FRAMING ERRORS'
                                         ;* 3) THE RECEIVER CAN FLAG 'PARITY ERRORS'
                                         ;*ONLY ONE LINE AT A TIME WILL BE EXERCISED.
                                 ;:*  TEST 16
                                 ;:*************************************************************
015054  000004                   TST16:  SCOPE
015056  012737  000016  001246           MOV     #16,$TSTNM         ;LOAD THE NUMBER OF THIS TEST
015064  012737  015256  001362           MOV     #TST17,NEXT        ;POINT TO THE START OF THE NEXT TEST
015072  012737  015202  001364           MOV     #5$,LOCK           ;SET FOR LOOP
015100  005037  001374                   CLR     SAVLIN             ;INIT LINE INDIC. FOR ERROR PRINTOUT
015104  012702  000001                   MOV     #1,R2              ;LINE POINTER
015110  030237  001366           1$:     BIT     R2,LINE            ;VALID LINE?
015114  001454                           BEQ     9$                 ;IF NOT SET FOR NEXT LINE
015116  104417                           DCLASM                     ;SET DCLR IN CSR AND SET MNTFLG
015120  013701  001370                   MOV     PAR,R1             ;PICK UP PARAMETERS
015124  052737  000300  001370           BIS     #ODDPAR!PARITY,PAR ;FORCE ODD PARITY
015132  104421                           LPRSET                     ;LOAD LPR REGISTER
015134  010137  001370                   MOV     R1,PAR             ;RESET PAR TO ORIGINAL VALUE
015140  052777  000040  164642           BIS     #MSENAB,@DZVCSR    ;START SCANNER
015146  013705  001374                   MOV     SAVLIN,R5          ;MAKE EXPECTED DATA
015152  005737  001372                   TST     MODE               ;IS THIS TEST IN STAGGERED MODE?
015156  100006                           BPL     4$                 ;IF NOT, SKIP STAGGERED SETUP

                                         ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER

015160  006205                           ASR     R5                 ;GET THE LAST BIT INTO THE CARRY BIT
015162  103402                           BCS     2$                 ;IF IT IS SET, GO CLEAR IT
015164  000261                           SEC                        ;IF IT IS CLEAR SET IT HERE
015166  000401                           BR      3$                 ;SKIP THE CLEARING
015170  000241                   2$:     CLC                        ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
015172  006105                   3$:     ROL     R5                 ;GET THE NEW BIT BACK INTO R5
015174  000305                   4$:     SWAB    R5                 ;PUT LINE NUMBER IN UPPER BYTE
015176  052705  130000                   BIS     #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
015202  005003                   5$:     CLR     R3                 ;INIT DELAY ACCUMULATOR
015204  110277  164626                   MOVB    R2,@HDZVTDR        ;SET BREAK BIT
015210  105777  164574           6$:     TSTB    @DZVCSR            ;RECEIVER DONE?
015214  100404                           BMI     7$                 ;BRANCH IF YES
015216  104414                           DELAY                      ;WAIT FOR REC DONE TO SET
015220  005203                           INC     R3                 ;INC DELAY LOOP
015222  001372                           BNE     6$                 ;DELAY FINISHED?
015224  104004                           ERROR   4                  ;*RDONE FAILED TO SET!
015226  017704  164562           7$:     MOV     @DZVRBUF,R4        ;ACTUAL
```

```
3548  015232  020405                          CMP     R4,R5           ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
3549  015234  001401                          BEQ     8$              ;IF YES, GO CLEAN UP
3550  015236  104006                          ERROR   6               ;*DATA/CONTENTS FAILED TO COMPARE
3551  015240  105077  164572          8$:     CLRB    @#DZVTDR        ;CLEAR BREAK BITS
3552  015244  104401                          SCOP1                   ;LOOP?
3553  015246  005237  001374          9$:     INC     SAVLIN          ;INC LINE #
3554  015252  104420                          SHIFT                   ;SET R2 TO NEXT LINE
3555  015254  000715                          BR      1$              ;GO BACK AND TEST NEXT LINE
3556                                          ;********************** TEST 17 ****************************
3557                                          ;* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
3558                                          ;*WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
3559                                          ;*BUT WILL INTERRUPT IF THE PROCESSOR STATUS
3560                                          ;*ALLOWS INTERRUPTS.
3561                                  ;:*  TEST 17
3562                                  ;;*************************************************************
3563  015256  000004                  TST17:  SCOPE
3564  015260  012737  000017  001246          MOV     #17,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3565  015266  012737  015652  001362          MOV     #TST20,NEXT     ;POINT TO THE START OF THE NEXT TEST
3566  015274  104417                          DCLASM                  ;SET DCLR IN CSR AND SET MAINT BIT
3567                                          ;IF NECESSARY (INTERNAL MODE)
3568  015276  104421                          LPRSET                  ;SET UP LPR REGISTER
3569  015300  005037  001374                  CLR     SAVLIN          ;INIT LINE INDIC. FOR ERROR
3570  015304  105037  001425                  CLRB    DONFLG          ;INIT TCR BIT HANDLER FLAG
3571  015310  113777  001366  164506          MOVB    LINE,@DZVTCR    ;SET ALL VALID TCR BITS
3572  015316  106427  000200                  MTPS    #MASK           ;SET CPU STATUS TO DZV11 PRIO.
3573  015322  012777  000200  164512          MOV     #MASK,@DZVRIS   ;SET RECEIVER STATUS
3574  015330  012777  000200  164510          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER STATUS
3575  015336                          1$:
3576  015336  012777  015424  164500          MOV     #6$,@DZVTIV     ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3577  015344  012777  015446  164466          MOV     #7$,@DZVRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
3578  015352  012777  000200  164462          MOV     #MASK,@DZVRIS   ;SET THE INTERRUPT VECTOR STATUS
3579  015360  012777  000200  164460          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
3580  015366  052777  040040  164414          BIS     #TIE!MSENAB,@DZVCSR ;ENABLE THE DEVICE
3581  015374  005005                          CLR     R5              ;INIT DELAY COUNTER
3582  015376  005777  164406          4$:     TST     @DZVCSR         ;TRDY SET?
3583  015402  100003                          BPL     5$              ;IF NOT GO DO DELAY
3584  015404  000240                          NOP                     ;WAIT FOR INTERRUPT
3585  015406  000240                          NOP
3586  015410  000420                          BR      8$              ;GO CLEAR TIE BIT
3587  015412  104414                  5$:     DELAY                   ;DELAY ROUTINE CALL
3588  015414  005205                          INC     R5              ;INC DELAY COUNTER
3589  015416  001367                          BNE     4$              ;DELAY FINISHED?
3590  015420  104003                          ERROR   3               ;*TRDY NOT SET!
3591  015422  000413                          BR      8$              ;GO CLEAR TIE
3592  015424  022626                  6$:     POP2SP                  ;REMOVE THE INTERRUPT FROM THE STACK
3593  015426  042777  040000  164354          BIC     #TIE,@DZVCSR    ;DON'T LET ANY MORE INTERRUPTS OCCUR
3594  015434  105737  001425                  TSTB    DONFLG          ;PROCESSOR ALLOWING INTER?
3595  015440  001013                          BNE     10$             ;IF YES NO ERROR
3596  015442  104010                          ERROR   10              ;IF NOT PRINT ERROR
3597  015444  000413                          BR      9$              ;RETURN TO THE NORMAL FLOW
3598  015446  104012                  7$:     ERROR   12              ;*RECEIVER SHOULD NOT INTERRUPT
3599  015450  022626                          POP2SP                  ;POP FOR FAKE RTI
3600  015452  042777  040000  164330  8$:     BIC     #TIE,@DZVCSR    ;RESET TRANSMITTER INTERRUPT ENABLE
3601  015460  105737  001425                  TSTB    DONFLG          ;INTERRUPTS ENABLED?
3602  015464  001403                          BEQ     9$              ;IF NOT GET OUT
3603  015466  104007                          ERROR   7               ;IF YES TRANS FAILED TO INTER.
```

```
3604  015470  106427  000000          10$:    MTPS    #CLEAR          ;ALLOW INTERRUPTS
3605  015474                          9$:
3606  015474  012777  015600  164342          MOV     #11$,@DZVTIV    ;SET UP THE TRANSMITTER INTERRUPT VECTOR
3607  015502  012777  015604  164330          MOV     #12$,@DZVRIV    ;SET UP THE RECEIVER INTERRUPT VECTOR
3608  015510  012777  000200  164324          MOV     #MASK,@DZVRIS   ;SET THE INTERRUPT VECTOR STATUS
3609  015516  012777  000200  164322          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
3610  015524  052777  000140  164256          BIS     #RIE!MSENAB,@DZVCSR ;ENABLE THE DEVICE
3611  015532  113777  001426  164274          MOVB    TDO,@DZVTDR     ;LOAD BUFFER WITH ANY CHAR.
3612  015540  005005                          CLR     R5              ;INIT DELAY ACCUMULATOR
3613  015542  105777  164242          13$:    TSTB    @DZVCSR         ;REC. DONE?
3614  015546  100003                          BPL     14$             ;IF NOT DELAY
3615  015550  000240                          NOP                     ;WAIT FOR INTERRUPT
3616  015552  000240                          NOP
3617  015554  000404                          BR      18$
3618  015556  104414                  14$:    DELAY                   ;DELAY FOR INTERRUPT
3619  015560  005205                          INC     R5              ;INCREMENT DELAY COUNTER
3620  015562  001367                          BNE     13$             ;DELAY FINISHED?
3621  015564  104004                          ERROR   4               ;*NO RX DONE! (NOT SET)
3622  015566  105737  001425          18$:    TSTB    DONFLG          ;PROCESSOR ALLOWING INTERRUPTS?
3623  015572  001411                          BEQ     15$             ;IF NOT DON'T PRINT ERROR
3624  015574  104011                          ERROR   11              ;RECEIVER FAILED TO INTERRUPT
3625  015576  000407                          BR      15$             ;CONTINUE TEST
3626  015600  104010                  11$:    ERROR   10              ;TRANSMITTER SHOULD NOT INTER.
3627  015602  000404                          BR      16$             ;CONT TEST
3628  015604  105737  001425          12$:    TSTB    DONFLG          ;PROCESSOR ALLOWING INTERRUPTS?
3629  015610  001001                          BNE     16$             ;IF YES DON'T PRINT ERROR
3630  015612  104012                          ERROR   12              ;*RECEIVER SHOULD NOT INTERRUPT
3631  015614  022626                  16$:    POP2SP                  ;POP FOR FAKE RTI
3632  015616  042777  040100  164164  15$:    BIC     #RIE!TIE,@DZVCSR        ;CLEAR INTERRUPTS
3633  015624  105737  001425                  TSTB    DONFLG          ;SECOND TIME THROUGH?
3634  015630  001005                          BNE     17$             ;IF YES LEAVE TEST
3635  015632  105237  001425                  INCB    DONFLG          ;IF NO INDICATE SECOND TEST PASS
3636  015636  106427  000000                  MTPS    #CLEAR          ;ALLOW INTERRUPTS
3637  015642  000635                          BR      1$              ;RESTART TEST
3638  015644  106427  000200          17$:    MTPS    #MASK           ;DON'T ALLOW INTERRUPTS
3639  015650  104413                          DEVICE.CLR              ;CLEAR DEVICE, LEAVE TEST
3640
3641                                          ;************************ TEST 20 *****************************
3642                                          ;*THIS TEST VERIFIES THAT THE RECEIVER WILL
3643                                          ;*INTERRUPT BEFORE THE TRANSMITTER EVEN
3644                                          ;*THOUGH THE TRANSMITTER WAS ENABLED
3645                                          ;*FIRST.  SET PS TO HIGH (MASK INTERRUPTS);
3646                                          ;*GET RDONE AND TRDY TO SET;
3647                                          ;*SET TX IE AND RX IE;
3648                                          ;*CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
3649                                  ;;*  TEST 20
3650                                  ;;*************************************************************
3651  015652  000004                  TST20:  SCOPE
3652  015654  012737  000020  001246          MOV     #20,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
3653  015662  012737  004100  001362          MOV     #$EOP,NEXT      ;POINT TO THE END-OF-PASS HANDLER
3654  015670  104417                          DCLASM                  ;SET DCLR IN CSR AND MNTFLG
3655  015672  104421                          LPRSET                  ;LOAD PAR REGISTER FOR ALL LINES
3656  015674  005037  001374                  CLR     SAVLIN          ;INIT. ERROR LINE INDIC.
3657  015700  012777  016110  164132          MOV     #8$,@DZVRIV     ;SETUP INTERRUPT STUFF
3658  015706  012777  000200  164126          MOV     #MASK,@DZVRIS   ;
3659  015714  012777  016176  164122          MOV     #12$,@DZVTIV    ;
```

```
3660  015726  012777  000200  164116          MOV     #MASK,@DZVTIS   ;
3661  015730  052777  000040  164052          BIS     #MSENAB,@DZVCSR
3662  015736  012702  000001                  MOV     #1,R2           ;LINE POINTER
3663  015742  030237  001366          3$:     BIT     R2,LINE         ;VALID LINE ?
3664  015746  001515                          BEQ     14$             ;IF NOT GO TO NEXT LINE
3665  015750  106427  000200          4$:     MTPS    #MASK
3666  015754  110277  164044                  MOVB    R2,@DZVTCR      ;SET TCR BIT
3667  015760  005777  164030                  TST     @DZVRBUF        ;VALID DATA?
3668  015764  100001                          BPL     .+4             ;IT BETTER NOT BE SET
3669  015766  104017                          ERROR   17              ;DATA VALID SHOULD NOT BE SET
3670  015770  105777  164014          5$:     TSTB    @DZVCSR         ;RECEIVER DONE ?
3671  015774  100001                          BPL     .+4
3672  015776  104020                          ERROR   20              ;RECEIVER DONE BIT SHOULD NOT BE SET
3673  016000  005005                          CLR     R5
3674  016002  005004                          CLR     R4
3675  016004  005777  164000          99$:    TST     @DZVCSR         ;WAIT FOR TRDY
3676  016010  100404                          BMI     100$            ;BR IF READY
3677  016012  104414                          DELAY                   ;STALL TIME
3678  016014  005204                          INC     R4              ;
3679  016016  001372                          BNE     99$
3680  016020  104003                          ERROR   3               ;TRDY FAILED TO SET
3681  016022  105077  164006          100$:   CLRB    @DZVTDR         ;SEND A ZERO CHARACTER
3682  016026  005004                          CLR     R4
3683  016030  105777  163754          6$:     TSTB    @DZVCSR         ;IS RDONE SET?
3684  016034  100404                          BMI     7$
3685  016036  104414                          DELAY
3686  016040  005204                          INC     R4
3687  016042  001372                          BNE     6$
3688  016044  104004                          ERROR   4               ;*RDONE FAILED TO SET!
3689  016046  005777  163736          7$:     TST     @DZVCSR         ;TRANS DONE BIT = 1 ?
3690  016052  100401                          BMI     .+4             ;YES
3691  016054  104003                          ERROR   3               ;*NO   TRANS DONE FAILED TO SET
3692                                          ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
3693                                          ;SET INTERRUPT ENABLES
3694  016056  052777  040000  163724          BIS     #TIE,@DZVCSR
3695  016064  052777  000100  163716          BIS     #RIE,@DZVCSR
3696  016072  106427  000000                  MTPS    #CLEAR          ;ALLOW THE INTERRUPTS
3697  016076  000240                          NOP
3698  016100  000240                          NOP
3699  016102  104007                          ERROR   7               ;*TRANSMITTER FAILED TO INTERRUPT
3700  016104  104011                          ERROR   11              ;*RECEIVER FAILED TO INTERRUPT
3701  016106  000435                          BR      14$             ;GET OUT
3702
3703                                          ;RECEIVER INTERRUPT ROUTINE
3704  016110  017704  163700          8$:     MOV     @DZVRBUF,R4     ;ACTUAL
3705  016114  010403                          MOV     R4,R3
3706  016116  000303                          SWAB    R3
3707  016120  042703  177770                  BIC     #^C<7>,R3       ;STRIP JUNK
3708  016124  005737  001372                  TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
3709  016130  100006                          BPL     11$             ;IF NOT, SKIP STAGGERED SETUP
3710
3711                                          ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
3712
3713  016132  006203                          ASR     R3              ;GET THE LAST BIT INTO THE CARRY BIT
3714  016134  103402                          BCS     9$              ;IF IT IS SET, GO CLEAR IT
3715  016136  000261                          SEC                     ;IF IT IS CLEAR SET IT HERE
```

```
3716  016140  000401                  BR      10$             ;SKIP THE CLEARING
3717  016142  000241          9$:     CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
3718  016144  006103          10$:    ROL     R3              ;GET THE NEW BIT BACK INTO R3
3719  016146  020337  001374  11$:    CMP     R3,SAVLIN       ;IS THIS A VALID LINE
3720  016152  001401                  BEQ     .+4             ;YES
3721  016154  104015                  ERROR   15              ;*INVALID LINE
3722  016156  042704  177400          BIC     #^C<377>,R4     ;STRIP JUNK
3723  016162  120504                  CMPB    R5,R4           ;DATA COMPARE ?
3724  016164  001401                  BEQ     .+4             ;YES
3725  016166  104005                  ERROR   5               ;*DATA DOES NOT COMPARE
3726  016170  040277  163630          BIC     R2,@DZVTCR      ;CLEAR TCR BIT
3727  016174  000401                  BR      13$             ;GO GET OUT OF INTERRUPT MODE
3728                                  ;TRANSMITTER INTE RUPT SVC ROUTINE
3729  016176  104011          12$:    ERROR   11              ;THE RECEIVER INTERRUPT FAILED
3730                                                          ;TO OVERRIDE THE TRANSMITTER
3731  016200  022626          13$:    POP2SP                  ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
3732  016202  005237  001374  14$:    INC     SAVLIN          ;ADJUST FOR NEXT LINE
3733  016206  104420                  SHIFT                   ;GET THE NEXT POINTER. IF DONE, ADVANCE
3734  016210  000137  015742          JMP     3$              ;OTHERWISE GO DO THE NEXT LINE
```

```
3735                                    ;ERROR TABLE
3736  016214  000000    .ERRTAB:        0       ;ERROR 0
3737  016216  000000                    0
3738  016220  000000                    0
3739
3740  016222  016362                    EM1     ;ERROR
3741  016224  017200                    DH1
3742  016226  017320                    DT1
3743
3744  016230  016435                    EM2     ;ERROR 2
3745  016232  017224                    DH2
3746  016234  017332                    DT2
3747
3748  016236  016463                    EM3     ;ERROR 3
3749  016240  017257                    DH3
3750  016242  017350                    DT3
3751
3752  016244  016522                    EM4     ;ERROR 4
3753  016246  017257                    DH3
3754  016250  017350                    DT3
3755
3756  016252  016551                    EM5     ;ERROR 5
3757  016254  017271                    DH4
3758  016256  017356                    DT4
3759
3760  016260  016600                    EM6     ;ERROR 6
3761  016262  017271                    DH4
3762  016264  017356                    DT4
3763
3764  016266  016637                    EM7     ;ERROR 7
3765  016270  017257                    DH3
3766  016272  017350                    DT3
3767
3768  016274  016700                    EM10    ;ERROR 10
3769  016276  017257                    DH3
3770  016300  017350                    DT3
3771
3772  016302  016742                    EM11    ;ERROR 11
3773  016304  017257                    DH3
3774  016306  017350                    DT3
3775
3776  016310  017000                    EM12    ;ERROR 12
3777  016312  017257                    DH3
3778  016314  017350                    DT3
3779
3780  016316  000000                    0
3781  016320  000000                    0
3782  016322  000000                    0
3783
3784  016324  000000                    0
3785  016326  000000                    0
3786  016330  000000                    0
3787
3788  016332  017037                    EM15    ;ERROR 15
3789  016334  000000                    0
3790  016336  000000                    0
```

```
3791
3792  016340  000000                  0
3793  016342  000000                  0
3794  016344  000000                  0
3795
3796  016346  017101                  EM17    ;ERROR 17
3797  016350  017257                  DH3
3798  016352  017350                  DT3
3799
3800  016354  017137                  EM20
3801  016356  017257                  DH3
3802  016360  017350                  DT3
```

```
3803                                                    ;ERROR MESSAGES
3804  016362 047200 020117 052502 EM1:    .ASCIZ  <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
      016435    200 042522 044507 EM2:    .ASCIZ  <200>?REGISTER R/W FAILURE?
      016463    200 051124 047101 EM3:    .ASCIZ  <200>/TRANSMIT READY (TRDY) NOT SET/
      016522 051200 041505 044505 EM4:    .ASCIZ  <200>/RECEIVER DONE NOT SET/
      016551    200 040504 040524 EM5.    .ASCIZ  <200>/DATA COMPARISON ERROR/
      016600 042200 053132 030461 EM6:    .ASCIZ  <200>/DZV11 *RECEIVER BUFFER* ERROR/
      016637    200 051124 047101 EM7:    .ASCIZ  <200>/TRANSMITTER FAILED TO INTERRUPT/
      016700 052600 042516 050130 EM10:   .ASCIZ  <200>/UNEXPECTED TRANSMITTER INTERRUPT/
      016742 051200 041505 044505 EM11:   .ASCIZ  <200>/RECEIVER FAILED TO INTERRUPT/
      017000 052600 042516 050130 EM12:   .ASCIZ  <200>/UNEXPECTED RECEIVER INTERRUPT/
      017037    200 041501 044524 EM15:   .ASCIZ  <200>/ACTION DETECTED ON INVALID LINE./
      017101    200 040504 040524 EM17:   .ASCIZ  <200>/DATA VALID SHOULD NOT BE SET/
      017137    200 042522 042503 EM20:   .ASCIZ  <200>/RECEIVER DONE SHOULD NOT BE SET/

      017200 052200 040522 020120 DH1:    .ASCIZ  <200>/TRAP PC  DZV11 REG/
      017224 042600 050130 041505 DH2:    .ASCIZ  <200>/EXPECTED  FOUND  REGISTER/
      017257    200 044514 042516 DH3:    .ASCIZ  <200>/LINE NO./
      017271    200 054105 042520 DH4:    .ASCIZ  <200>/EXPECTED  FOUND  LINE/

                                  .EVEN
                                          ;DATA TABLES FOR ERROR MESSAGES
3805  017300 000002               DT1:    2
3806  017322    006     003               .BYTE   6,3
3807  017324 001330                       $REG1
3808  017326    006     001               .BYTE   6,1
3809  017330 001326                       $REG0
3810
3811  017332 000003               DT2:    3
3812  017334    006     004               .BYTE   6,4
3813  017336 001340                       $REG5
3814  017340    006     001               .BYTE   6,1
3815  017342 001336                       $REG4
3816  017344    006     001               .BYTE   6,1
3817  017346 001326                       $REG0
3818
3819  017350 000001               DT3:    1
3820  017352    003     001               .BYTE   3,1
3821  017354 001374                       SAVLIN
3822
3823  017356 000003               DT4:    3
3824  017360    006     004               .BYTE   6,4
3825  017362 001340                       $REG5
3826  017364    006     001               .BYTE   6,1
3827  017366 001336                       $REG4
3828  017370    003     001               .BYTE   3,1
3829  017372 001374                       SAVLIN
3830
3831                                      ;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES
3832                                      ;----------------------------------------------
3833
3834  017374 002450               DLYTBL: 2450                    ;TIME FOR   50 BAUD
3835  017376 001560                       1560                    ;TIME FOR   75 BAUD
3836  017400 001120                       1120                    ;TIME FOR  110 BAUD
3837  017402 000750                        750                    ;TIME FOR  134 BAUD
3838  017404 000660                        660                    ;TIME FOR  150 BAUD
```

```
3839  017406  000330                    330                     ;TIME FOR  300 BAUD
3840  017410  000150                    150                     ;TIME FOR  600 BAUD
3841  017412  000060                     60                     ;TIME FOR 1200 BAUD
3842  017414  000040                     40                     ;TIME FOR 1800 BAUD
3843  017416  000030                     30                     ;TIME FOR 2000 BAUD
3844  017420  000020                     20                     ;TIME FOR 2400 BAUD
3845  017422  000010                     10                     ;TIME FOR 3600 BAUD
3846  017424  000001                      1                     ;TIME FOR 4800 BAUD
3847  017426  000001                      1                     ;TIME FOR 7200 BAUD
3848  017430  000001                      1                     ;TIME FOR 9600 BAUD
3849  017432  000001                      1                     ;TIME OF DELAY FOR 19200 BAUD
3850
3851                                                    ;DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
3852                                                    ;FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.
3853
3854  017434                          CORMAX.
3855              000001              .END
```

```
ABASE = 160010          804#    1145    1186
ACDW1 = 000017          804#    1145    1188
ACDW2 = 000000         1145     1189
ACPUOP= 000000         1145     1160
ACTIVE  001420         1303#    1561*   2594*   2595    2597*   2601    2602*   2603    2606*
ADDW0 = 017470          804#    1145    1190
ADDW1 = 017470          804#    1145    1191
ADDW10= 017470          804#    1145    1200
ADDW11= 017470          804#    1145    1201
ADDW12= 017470          804#    1145    1202
ADDW13= 017470          804#    1145    1203
ADDW14= 017470          804#    1145    1204
ADDW15= 017470          804#    1145    1205
ADDW2 = 017470          804#    1145    1192
ADDW3 = 017470          804#    1145    1193
ADDW4 = 017470          804#    1145    1194
ADDW5 = 017470          804#    1145    1195
ADDW6 = 017470          804#    1145    1196
ADDW7 = 017470          804#    1145    1197
ADDW8 = 017470          804#    1145    1198
ADDW9 = 017470          804#    1145    1199
ADEVCT= 000000         1145     1151
ADEVM1 = 000001         804#    1145    1187
ADRCNT  005675         2128*    2165*   2175#
ADVANC= 104400         1455#    2305    2914    3186
AENV  = 000000         1145     1156
AENVM1 = 000000        1145     1157
AFATAL= 000000         1145     1148
AMADR1= 000000         1145     1173
AMADR2= 000000         1145     1177
AMADR3= 000000         1145     1180
AMADR4= 000000         1145     1183
AMAMS1= 000000         1145     1167
AMAMS2= 000000         1145     1175
AMAMS3= 000000         1145     1178
AMAMS4= 000000         1145     1181
AMSGAD= 000000         1145     1153
AMSGLG= 000000         1145     1154
AMSGTY= 000000         1145     1147
AMTYP1= 000000         1145     1168
AMTYP2= 000000         1145     1176
AMTYP3= 000000         1145     1179
AMTYP4= 000000         1145     1182
APASS = 000000         1145     1150
APRIOR= 000000         1145
APTCSU= 000040         1979     2084#
APTENV= 000001         1972     2040    2082#   2392
APTSIZ= 000200         2081#
APTSPO= 000100         1974     2042    2083#
ASUREG= 000000         1145     1158
ATESTM= 000000         1145     1149
AUNIT = 000000         1145     1152
AUSER = 000000         1145     1159
AUTO.S  011500         1736     2777#
AVECT1= 000300          804#    1145    1184
AVECT2= 000000         1145     1185
```

```
BIMSRO  006150          2251#
BIT0    = 000001         942#  1014  1016  1053  1064  3200
BIT00   = 000001         932#   942
BIT01   = 000002         931#   941
BIT02   = 000004         930#   940
BIT03   = 000010         929#   939
BIT04   = 000020         928#   938
BIT05   = 000040         927#   937
BIT06   = 000100         926#   936
BIT07   = 000200         925#   935
BIT08   = 000400         924#   934
BIT09   = 001000         923#   933
BIT1    = 000002         941#  1015  1016  1054  1065
BIT10   = 002000         922#  1038  1039  1040  1041  1046  1047  1048  1049  1059  1070  1078  3132
BIT11   = 004000         921#  1042  1043  1044  1045  1046  1047  1048  1049  1060  1071  1079  1912
                        3132
BIT12   = 010000         920#   981   997  1032
BIT13   = 020000         919#   982   998
BIT14   = 040000         918#   983   999  1894  2856
BIT15   = 100000         917#   984  1000
BIT2    = 000004         940#  1055  1066  1797
BIT3    = 000010         939#   976  1019  1021  1023  1025  1056  1067
BIT4    = 000020         938#   977  1020  1021  1024  1025  2302  2317
BIT5    = 000040         937#   978  1022  1023  1024  1025  1030  2789  2799  2856
BIT6    = 000100         936#   979  1027
BIT7    = 000200         935#   968   980  1028  1734  2421  2435
BIT8    = 000400         934#   989   991  1006  1008  1035  1037  1039  1041  1043  1045  1047  1049
                        1057  1068  1076  3062  3072
BIT9    = 001000         933#   990   991  1007  1008  1036  1037  1040  1041  1044  1045  1048  1049
                        1058  1069  1077  3062  3072
BPTVEC= 000014           949#
BRK0    = 000400        1076#
BRK1    = 001000        1077#
BRK2    = 002000        1078#
BRK3    = 004000        1079#
BRW     004554          1802  1932#
BUFSET= 104422          1491#  3443
CHRCNT  006146          2217*  2230*  2248#
CLEAR   = 000000         969#  3604*  3636*  3596*
CNVRT   = 104412        1475#  1825   1827  1830  1833  2371  2373  2375  2428
CONVRT= 104411          1473#  1746   2389
CORMAX  017434          3854#  3855
CO0     = 000400        1068#
CO1     = 001000        1069#
CO2     = 002000        1070#
CO3     = 004000        1071#
CR      = 000015         857#  2018   2028
CRLF    = 000200         858#  1989   2028
CSRMAP  011506          2780#
CYCLE   010452          1803  1856   2584#
DATABP  006642          2360*  2363   2387  2390#
DATAHD  006630          2359*  2383   2386#
DCLASM= 104417          1485#  3278   3357  3440  3519  3566  3654
DCLR    = 000020         977#  2271   2272  2930  3092  3104  3111  3113
DDISP   = 177570         864#  1237
DELAY   = 104414        1479#  3075   3209  3217  3249  3294  3320  3370  3398  3413  3458  3483  3543
```

```
DZVC4   001552          1376#
DZVC5   001564          1382#
DZVC6   001576          1388#
DZVC7   001610          1394#
DZVLEV  011016          2616    2652#
DZVLPR  002060          1503#   2314*   2668*   3133*   3138*
DZVMSR  002030          1507#   2677*   2907    3154    3189
DZVNUM  001414          1299#   1642*   1750    2747*   2757*   2785*   2814*   2815    2821    2823
DZVRBU  002014          1501#   2667*   2895    3130    3324    3401    3423    3487    3547    3667    3704
DZVRIS  002042          1515#   2654*   3573*   3578*   3608*   3658*
DZVRIV  002040          1514#   1863    2608*   2652    3577*   3607*   3657*
DZVTCR  002024          1505#   2673*   2901    3000    3068*   3244*   3287*   3331*   3365*   3383*   3392*   3449*   3496*
                        3571*   3666*   3726*
DZVTDR  002034          1509#   2678*   3157*   3162*   3301*   3340*   3389*   3464*   3611*   3681*
DZVTIS  002046          1517#   2658*   3574*   3579*   3609*   3660*
DZVTIV  002044          1516#   2656*   3576*   3606*   3659*
DZV.EN  001740          1447#   1587    2595    2603    2783
DZV.MA  001500          1303    1349#   1561    1584    1741    2597    2606    2740    2780    2786    2844
EIGHT = 000030          1021#
EIGHTS= 000070          1025#
EMTVEC= 000030           952#   1557*   1558*
EM1     016362          3740    3804#
EM10    016700          3768    3804#
EM11    016742          3772    3804#
EM12    017000          3776    3804#
EM15    017037          3788    3804#
EM17    017101          3796    3804#
EM2     016435          3744    3804#
EM20    017137          3800    3804#
EM3     016463          3748    3804#
EM4     016522          3752    3804#
EM5     016551          3756    3804#
EM6     016600          3760    3804#
EM7     016637          3764    3804#
ERRMSG  006616          2358*   2378    2381#
ERRVEC= 000004           945#   1899    1900*   1902*   1905*
ERTABO  006772          2373    2414#
EVEPAR= 000000          1031#
EXITER  006722          2401    2404#
FIVE  = 000000          1018#
FIVES = 000040          1022#
FRMERR= 020000           998#   3538
HALTS   006646          2343    2392#
HDRFLG  001423          1309#   1585*   1684*   1737    1739*
HDZVCS  002012          1500#   2665*   3105*   3374
HDZVLP  002022          1504#   2671*
HDZVMS  002032          1508#   2680*
HDZVRB  002016          1502#   2670*
HDZVTC  002026          1506#   2675*   3208*   3216*
HDZVTD  002036          1510#   2681*   3540*   3551*
HILIM   005670          2125*   2153    2172#
HT    = 000011           855#   1987    2028
INBUF   010304          2095    2131    2524*   2525    2530    2535    2553*   2554    2558    2568#   2692
INIFLG  001422          1308#   1571    1576*   1589*   1670*   1732    2779*
INSTER= 104404          1463#   2147    2541    2563    2711
INSTR   104403          1461#   1596    1608    1619    1626    1676    1709    1754    1774    2622
```

```
MERR2    007527        2512#  2586   2835
MERR3    007576        1764   2512#
MLOCK    007622        1779   2512#
MNEW     007724        1755   2512#
MNTFLG   001424        1310#  1927*  1930*  2279   2528*  2533*  2538*  2612*  2615*
MODE     001372        1290#  1928   2611*  2613   3184   3198   3303   3466   3526   3708
MPASSX   007665        1829   2512#
MPFAIL   007424        2505   2512#
MR       007513        1805   2512#
MSENAB=  000040         978#  2984   3067   3069   3093   3106   3240   3245   3284   3366   3446   3524   3580
                       3610   3661
MTITLE   001000        1139#  1575
MTSIN    007707        2370   2512#  2623
MVECX    007657        1826   2512#
NEXT     001362        1282#  2294   2410   2885*  2928*  2951*  2999*  3058*  3089*  3128*  3152*  3183*  3235*
                       3276*  3354*  3438*  3513*  3565*  3653*
NOLIST=  ****** U       80#
ODDPAR=  000200        1028#  3521
ONESTO=  000000        1029#
OVRRUN=  040000         999#
PAR      001370        1289#  2312   2610*  3359   3360*  3362*  3409   3520   3521*  3523*
PARAM  = 104405        1465#  1598   1610   1628   1678   1711   1756   1776   2624
PARAM1   005536        2130#  2148
PARER  = 010000         997#  3538
PARERR   005612        2133   2135   2137   2147#  2154   2156   2158
PARITY=  000100        1027#  3521
PARMD  = 104415        1481#
PAR0     001506        1354#  1636*  1714   1718
PAR1     001520        1360#
PAR10    001626        1402#
PAR11    001640        1408#
PAR12    001652        1414#
PAR13    001664        1420#
PAR14    001676        1426#
PAR15    001710        1432#
PAR16    001722        1438#
PAR17    001734        1444#  1728
PAR2     001532        1366#
PAR3     001544        1372#
PAR4     001556        1378#
PAR5     001570        1384#
PAR6     001602        1390#
PAR7     001614        1396#
PAWCH  = 104416        1483#
PIRQ   = 177772         862#
PIRQVE=  000240         956#
POPR0  = 012600         965#
POP1SP=  005726         963#
POP2SP=  022626         967#  2304   2869   2916   3592   3599   3631   3731
PR0    = 000000         879#
PR1    = 000040         880#
PR2    = 000100         881#
PR3    = 000140         882#
PR4    = 000200         883#
PR5    = 000240         884#
PR6    = 000300         885#
```

```
SW03   = 000010          901#    911     1638    1666
SW04   = 000020          900#    910     1772
SW05   = 000040          899#    909
SW06   = 000100          898#    908     1751
SW07   = 000200          897#    907
SW08   = 000400          896#    906     2406
SW09   = 001000          895#    905     1938
SW1    = 000002          913#
SW10   = 002000          894#   2408
SW11   = 004000          893#
SW12   = 010000          892#   1945     2337
SW13   = 020000          891#   2342
SW14   = 040000          890#
SW15   = 100000          889#
SW2    = 000004          912#
SW3    = 000010          911#
SW4    = 000020          910#
SW5    = 000040          909#
SW6    = 000100          908#
SW7    = 000200          907#
SW8    = 000400          906#
SW9    = 001000          905#
S110   = 001000         1036#
S1200  = 003400         1041#
S134   = 001400         1037#
S150   = 002000         1038#
S1800  = 004000         1042#
S19200= 007400          1049#
S2000  = 004400         1043#
S2400  = 005000         1044#
S300   = 002400         1039#
S3600  = 005400         1045#
S4800  = 006000         1046#
S50    = 000000         1034#
S600   = 003000         1040#
S7200  = 006400         1047#
S75    = 000400         1035#
S9600  = 007000         1048#
TBITVE= 000014           947#
TCR0   = 000001         1053#   3022
TCR1   = 000002         1054#   3023
TCR2   = 000004         1055#   3024
TCR3   = 000010         1056#   3025
TD0      001426         1314#   2326    3464    3478    3492*   3611
TD1      001430         1315#
TD2      001432         1316#
TD3      001434         1317#
TEIGHT   002106         1540#
TFMP     010346         2570#
TFIVE    002114         1543#
TIE    = 040000          983#   2987    3093    3580    3593    3600    3632    3694
TKVEC  = 000060          954#
TLAST  = 015652         2642    3831#
TL0    = 000000          988#                                            !
TL1    = 000400          989#
TL2    = 001000          990#
```

```
TL3    = 001400            991#
TMTBL    002050           1524#
TPVEC  = 000064            955#
TRAPVE= 000034             953#
TRDY   = 100000            984#  3062    3067    3240
TRTVEC= 000014             948#
TR0      001436           1318#
TR1      001440           1319#
TR2      001442           1320#
TR3      001444           1321#
TSEVEN   002110           1541#
TSIX     002112           1542#
TST1     012204           1888    2630    2646    2883#
TST10    013340           3128    3150#
TST11    013424           3152    3181#
TST12    013620           3183    3233#
TST13    013752           3235    3274#
TST14    014242           3276    3352#
TST15    014564           3354    3436#
TST16    015054           3438    3511#
TST17    015256           3513    3563#
TST2     012374           2885    2926#
TST20    015652           3565    3651#  3831
TST21  = ****** U          3653
TST3     012440           2928    2949#
TST4     012616           2951    2997#
TST5     013022           2999    3056#
TST6     013124           3058    3087#
TST7     013254           3089    3126#
TTST     004326           1800*   1802*   1890#
TWOSTO= 000040            1030#
TYPDAT   006632           2364    2384    2387#
TYPE   = 104402           1459#   1575    1697    1698    1740    1764    1799    1805    1823    1824    1826    1829    1832
                          1992    2093    2112    2205    2238    2365    2366    2369    2370    2372    2374    2376    2380
                          2385    2427    2429    2457    2504    2586    2621    2639    2644    2835
TYPMSG   006522           2362    2365#
T110     002054           1527#
T1200    002066           1532#
T134     002056           1528#
T150     002060           1529#
T1800    002070           1533#
T2000    002072           1534#
T2400    002074           1535#
T300     002062           1530#
T3600    002076           1536#
T4800    002100           1537#
T50      002050           1525#
T600     002064           1531#
T7200    002102           1538#
T75      002052           1526#
T9600    002104           1539#
VECMAP   012014           2834    2842#
WRDCNT   006144           2213*   2239*   2247#
WTBS.F   006620           2379    2382#
XBX      006410           2338    2340    2342#
XCSR     004264           1825    1858#   2375
```

```
XERR     004306          1833    1867#
XHEAD    010006          1740    2512#
XMTCNT   001400          1293#
XMTLIN   001376          1292#
XPASS    004300          1830    1864#
XSTATQ   010076          1747    2512#
XTSTN    007000          2371    2417#
XVEC     004272          1827    1861#
XX     = 160210          1350#   1356#   1362#   1368#   1374#   1380#   1386#   1392#   1398#   1404#   1410#   1416#   1422#
                         1428#   1434#   1440#   1446#
YY     = 000500          1350#   1356#   1362#   1368#   1374#   1380#   1386#   1392#   1398#   1404#   1410#   1416#   1422#
                         1428#   1434#   1440#   1446#
ZZ     = 000020          1350#   1356#   1362#   1368#   1374#   1380#   1386#   1392#   1398#   1404#   1410#   1416#   1422#
                         1428#   1434#   1440#   1446#
$APTHD   001446          1332    1338#
$ASTAT= ****** U         2062    2077
$ATYC    005130          2033    2035#
$ATY1    005104          2031#
$ATY3    005112          1977    2032#
$ATY4    005122          2034#   2395
$AUTOB   001300          1233#
$BASE    001174          1186#   1604*   2607*   2662    2741    2830*
$BDADR   001266          1228#
$BDDAT   001272          1230#
$CDW1    001200          1188#   2760
$CDW2    001202          1189#   2762    2831*
$CHARC   005100          1994*   2004*   2011    2020*   2025#
$CMTAG   001244          1216#
$CM1   = 000006          1248#   1249#   1250#   1251#   1252#   1253#   1254#
$CM2   = 000014          1248#   1249#   1250#   1251#   1252#   1253#   1254#
$CM3   = 000006          1246#   1248
$CM4   = 000005          1254#   1255#   1256#   1257#   1258#   1259#
$CPUOP   001146          1160#
$CRLF    001357          1261#   1993    2028    2205    2365    2366    2376    2457    2621    2639
$DDW0    001204          1190#   1651    2745    2781
$DDW1    001206          1191#
$DDW10   001230          1200#
$DDW11   001232          1201#
$DDW12   001234          1202#
$DDW13   001236          1203#
$DDW14   001240          1204#
$DDW15   001242          1205#
$DDW2    001210          1192#
$DDW3    001212          1193#
$DDW4    001214          1194#
$DDW5    001216          1195#
$DDW6    001220          1196#
$DDW7    001222          1197#
$DDW8    001224          1198#
$DDW9    001226          1199#
$DEVCT   001130          1151#   1834*
$DEVM    001176          1187#   2746    2833*
$DOAGN   004260          1836    1843    1848    1854#
$E     = 000022           804#   2885    2886#   2928    2929#   2951    2952#   2999    3000#   3058    3059#   3089    3090#
                         3128    3129#   3152    3153#   3183    3184#   3235    3236#   3276    3278#   3354    3355#   3438
                         3440#   3513    3514#   3565    3566#   3653    3654#
```

```
$XTSTR  004354          1889    1897#
$Y      = 000023        1448#   1455    1457#   1459#   1461#   1463#   1465#   1467#   1469#   1471#   1473#   1475#   1477#
                        1479#   1481#   1483#   1485#   1487#   1489#   1491#   1493#
$$GET4= 000000          1849#
$$LOCAT= ****** U       1894
.       = 017434        1107#   1108    1111#   1124    1125#   1127#   1129#   1131#   1134#   1138#   1140#   1263    1296#
                        1297#   1299#   1301#   1302#   1328    1329#   1331#   1333#   1348#   1351#   1352#   1353#   1354#
                        1355#   1357#   1358#   1359#   1360#   1361#   1363#   1364#   1365#   1366#   1367#   1369#   1370#
                        1371#   1372#   1373#   1375#   1376#   1377#   1378#   1379#   1381#   1382#   1383#   1384#   1385#
                        1387#   1388#   1389#   1390#   1391#   1393#   1394#   1395#   1396#   1397#   1399#   1400#   1401#
                        1402#   1403#   1405#   1406#   1407#   1408#   1409#   1411#   1412#   1413#   1414#   1415#   1417#
                        1418#   1419#   1420#   1421#   1423#   1424#   1425#   1426#   1427#   1429#   1430#   1431#   1432#
                        1433#   1435#   1436#   1437#   1438#   1439#   1441#   1442#   1443#   1444#   1445#   1731#   1857
                        1933    1934    2028    2080#   2469#   2486    2510    2569#   2571#   2573#   2588    2838    2861
                        3668    3671    3690    3720    3724
.ADVAN  006246          1456    2294#
.BEGIN  004004          1793#
.BUFSE  006336          1492    2325#
.CNVRT  005774          1476    2206#
.CONVR  005770          1474    2205#
.DCLAS  006214          1486    2278#
.DELAY  006226          1480    2282#
.DEVIC  006174          1478    2270#
.ERRTA  016214          2357    3736#
.INSTE  005456          1464    2110#
.INSTR  005352          1462    2089#
.INST1  005372          2093#   2113
.LPRSE  006276          1490    2310#
.MSG    005374          2091*   2094#
.PARAM  005476          1466    2121#
.PARMD  011156          1482    2684#
.PAUCH  010230          1484    2552#   2564
.RESOS  005736          1472    2194#
.SAVOS  005676          1470    2180#
.SCOPE  004314          1112    1887#   2873
.SCOP1  004560          1458    1938#
.SETFL  010110          1468    2523#   2542
.SHIFT  006260          1488    2301#
.START  002116          1135    1552#   1566
.TRPSR  006152          1118    2259#
.TRPTA  001742          1454#   2264
.TYPE   004604          1460    1945#
.$ASTA= ****** U        2032    2035
.$X     = 001446        1328#   1333
```

$STAG	804#	3303	3466	3526	3708										
$STAGF	804#														
$TCR	804#	2990													
$TLINE	804#	3226													
$TRPDE	804#	1455	1457	1459	1461	1463	1465	1467	1469	1471	1473	1475	1477	1479	1481
	1483	1485	1487	1489	1491										
$TSTN	804#	2881	2924	2947	2995	3054	3085	3124	3148	3179	3231	3272	3350	3434	3509
	3561	3649													
$UNIBU	804#	28?7													
$VARIA	804#	1137													
$XZ	804#	2377	2881	2920	2924	2939	2947	2990	2995	3048	3054	3080	3085	3120	3124
	3144	3148	3169	3178	3226	3231	3263	3272	3344	3350	3429	3434	3503	3509	3556
	3561	3641	3649												
$$CMPRE	1140#	1248	1249	1250	1251	1252	1?53								
$$CMTM	1140#	1254	1255	1256	1257	1258									
$$ESCA	1#	957#													
$$NEWT	1#	957#	2882	2925	2948	2996	3055	3086	3125	3149	3180	3232	32?3	3351	3435
	3510	3562	3650												
$$SKIP	1#	957#													
.EQUAT	1#	847													
.HEADE	1#	804													
.KT11	1#														
.SETUP	1#														
.SWJMI	1#														
.$A T1	1#	1120													
.$*PTB	1#	1142#													
.$APTP	1#	1323													
.$APTY	1#	2028													
.$ASTA	1#														
.$CATC	1#														
.$CMTA	1#	1140#													
.$DB2D	1#														
.$DB2O	1#														
.$DIV	1#														
.$EOP	1#	1812													
.$ERRO	1#														
.$ERRT	1#														
.$FULT	1#														
.$FOWE	1#	2470													
.$RAND	1#														
.$RDDE	1#														
.$RDOC	1#														
.$READ	1#														
.$R2AZ	1#														
.$SAVE	1#														
.$SB2D	1#														
.$SB2O	1#														
.$SCOP	1#	1874													
.$SIZE	1#														
.$SUPR	1#														
.$TRAP	1#														
.$TYPB	1#														
.$TYPD	1#														
.$TYPE	1#	1949													
.$TYPO	1#														
.$40CA	1#														

```
.1170       1#

. ABS.  017434      000

 ERRORS DETECTED:  0

 RUN-TIME: 14 18 1 SECONDS
 RUN-TIME RATIO: 48/34=1.4
 CORE USED:  50K  (99 PAGES)
```