DZV11

DZV11 DIAG PRT 1
CVDZACO

AH-A878C-MC
FICHE 1 OF 1

JUL 1982
COPYRIGHT© 77-82
MADE IN USA

```
        .REM    &
```

IDENTIFICATION

PRODUCT CODE: AC-A877C-MC

PRODUCT NAME: CVDZAC0 DZV11 DIAG PRT1

DATE RELEASED: 17-FEB-82

MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

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

COPYRIGHT (C) 1977,1982 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

THE FUNCTION OF THE DZV11 DIAGNOSTICS IS TO VERIFY THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS ALSO VERIFY THAT THE DZV11 OPERATES IN ITS ENVIRONMENT SUCH AS THE SYSTEM IN WHICH IT IS INSTALLED.

PARAMETERS MAY BE SUPPLIED TO THE PROGRAM BY EITHER 'AUTO SIZING' OR INPUT FROM THE USER ON THE CONSOLE BY HAVING SW00-1 AT START TIME. AUTO SIZING WILL BE DONE ONLY THE FIRST TIME THE PROGRAM IS STARTED AND SW07=0 AND SW00=0 AND SW03=0. THE AUTOSIZER IS DESIGNED TO DETECT DZV11 DEVICE ADDRESSES AND VECTORS ONLY. ALL REMAINING PARAMETERS WILL DEFAULT TO CERTAIN VALUES (SEE SEC.8.5). CONSOLE INPUT MAY BE CONTROLLED AT ANY START TIME THROUGH THE USE OF SW00,SW03, SW04, AND SW06 (SEE SEC. 4.1.1 FOR A DETAILED DESCRIPTION OF THESE SWITCHES).

CURRENTLY THERE ARE THREE STANDALONE DIAGNOSTICS (CVDZA,CVDZB,AND CVDZC) ONE SYSTEM MODULE FOR DEC X/11 (CXDZBA), AND AN OVERLAY FOR ITEP (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.

```
**********************************************************************
*                                                            ::GPA   *
*                                                            ::GPA   *
* NOTE: THIS DIAGNOSTIC HAS BEEN MODIFIED TO RUN IN KXT11 (SBC 11/21)*
* BASED SYSTEMS.  THE PROGRAM WILL AUTOMATICALLY ADJUST ITSELF TO RUN*
* IN THE APPROPRIATE ENVIRONMENT AS FOLLOWS:                         *
*                                                                    *
*              LSI-11, 11/2, AND 11/23        SBC 11/21              *
*              -----------------------        ---------              *
* CSR RANGE:        160010 TO 163770       174000 TO 177770         *
* VECTOR RANGE:        300 TO 770             300 TO 370            *
* AUTO-SIZING FOR...                                                 *
* ...CSR AND VECTOR:    ENABLED                DISABLED              *
*                                                            ::GPA   *
**********************************************************************
```

2. REQUIREMENTS

2.1 EQUIPMENT

AN LSI11 CPU WITH MINIMUM 4K OF MEMORY.
ASR 33 (OR EQUIVALENT FOR CONSOLE)
DZV11 INTERFACE MODULE
H329 STAGGERED TURNAROUND CONNECTOR.
H325 CABLE TURNAROUND CONNECTOR.

NOTE: A STAGGERED TURNAROUND CONNECTOR IS NEEDED IN ORDER TO TEST THE PARITY LOGIC.

2.2 STORAGE

PROGRAM WILL USE ALL 4K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDF. 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 PRESERVF 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 TEST THAT IS NOT IN THE ORDER OF SEQUENCE THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS.
NOTE: IF RUNNING MULTIPLE DZV11'S; THE DZV11 YOU DESIRE TO BE UNDER TEST MUST BE SELECTED BY THE USE OF SW06 BEFORE LOCKING ON THE TEST. IN OTHER WORDS; EACH TIME THE PROGRAM IS STARTED; THE FIRST DZV11 WILL BE SELECTED TO BE UNDER TEST UNLESS SW06 IS USED TO SELECT ONLY ONE.

SW 09 LOOP ON CURRENT DATA: THIS SWITCH WILL ONLY WORK IF CALL 'SCOP1' 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 START'NG 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 UP TO 16 (DECIMAL) DZV11S SEQUENTIALY. THEY CONTAIN THE CSR,VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DZV11.
DZVACTV(1406)	EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DZV11 WILL BE TESTED IN TURN. EXAMPLE: (DZVACTV) 1406/0000000000011111 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.
04	000017	THIS IS THE BINARY REPRESENTATION OF WHAT LINES ARE TO BE TESTED.
506	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 SEQUENCE IS REPEATED.
NOTE: IF THE PROGRAM DOES NOT FIND YOUR DZV11, SOMETHING IS WRONG AND AUTO SIZING SHOULD NOT BE DONE.

8.5.2 FINDING THE VECTOR

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '.+2' (NEXT ADDRESS). BIT14 AND BIT5 (TX 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:

$SWREG	-(1142)	USED AS THE SOFTWARE SWITCH REGISTER WHILE RUNNING UNDER APT.
$VECT1	-(1170)	USED TO SPECIFY THE FIRST VECTOR ADDRESS
$BASE	-(1174)	USED TO INDICATE BOTTOM ADDRESS OF DZV11 UNDER TEST
$DEVM	-(1176)	A BIT MAP REPRESENTING WHICH DZV11'S WILL BE TESTED
$CDW1	-(1200)	USED TO INDICATE WHICH LINES TO RUN ON ALL DZV11'S
$CDW2	-(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).
$DDW0	-(1204)	EACH OF THE $DDW 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 $BASE 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.

10.0 PROGRAM DESCRIPTION

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

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

MISCELLANEOUS DEFINITIONS

GENERAL PURPOSE REGISTER DEFINITIONS

PRIORITY LEVEL DEFINITIONS

''SWITCH REGISTER'' SWITCH DEFINITIONS

DATA BIT DEFINITIONS (BIT00 TO BIT15)

BASIC ''CPU'' TRAP VECTOR ADDRESSES

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

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

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

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

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

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

```
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($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
SW14=1  LOOP ON TEST
SW11=1  INHIBIT ITERATIONS
CALL
        SCOPE           ;;SCOPE=IOT

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

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


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
```

```
********** 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
```

```
********** TEST 13 **********
TEST TO TRANSMIT ONE CHAR AND
RECEIVE ONE CHAR ON ONE LINE
AT A TIME. THE CHAR IS "252" AND
ALL SELECTED LINES WILL BE TURNED ON .

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 14 **********
THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
TO ZERO FOR EACH LINE.
THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
EMPTIED BY ISSUING A DEVICE MASTER CLEAR.

********** TEST 15 **********
 THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
(ONE LINE AT A TIME   BASED UPON VALID LINES)
THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED

********** 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 17 **********
 THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
BUT WILL INTERRUPT IF THE PROCESSOR STATUS
ALLOWS INTERRUPTS.

********** TEST 20 **********
THIS TEST VERIFIES THAT THE RECEIVER WILL
INTERRUPT BEFORE THE TRANSMITTER EVEN
THOUGH THE TRANSMITTER WAS ENABLED
FIRST.  SET PS TO HIGH (MASK INTERRUPTS);
GET RDONE AND TRDY TO SET;
SET TX IE AND RX IE;
CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
```

&

```
3026                            ;;GPA   PRGFRT  ^?CVDZAB/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2?,CVDZAB
3027                            ;;GPA   .HEADER <CVDZA-B>,<1977,1981>
 (2)                            .TITLE  CVDZA-C
 (2)                            ;*COPYRIGHT (C) 1977,1981
 (2)                            ;*DIGITAL EQUIPMENT CORP.
 (2)                            ;*MAYNARD, MASS. 01754
 (2)                            ;*
 (2)                            ;*
 (2)                            ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 (2)                            ;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
 (2)                            ;*
 (2)          000001            $TN=1
 (2)                                    ;STARTING PROCEDURE
 (2)                                    ;LOAD PROGRAM
 (2)                                    ;LOAD ADDRESS 000200
 (2)                                    ;PRESS START
 (2)                                    ;PROGRAM WILL TYPE
 (2)                                    ;''CVDZAC/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2''
 (2)                                    ;PROGRAM WILL TYPE 'RUNNING'' TO INDICATE THAT TESTING HAS STARTED
 (2)                                    ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
 (2)                                    ;AND THEN RESUME TESTING
 (2)
 (1)                            .REM    !
 (1)                            ;SWITCH REGISTER OPTIONS
 (1)                            ;----------------------
 (1)
 (1)
 (1)                            SW15=100000             ;=1,HALT ON ERROR
 (1)                            SW14=40000              ;=1,LOOP ON CURRENT TEST
 (1)                            SW13=20000              ;=1,INHIBIT ERROR TYPEOUT
 (1)                            SW12=10000              ;=1,DELETE TYPEOUT/BELL ON ERROR.
 (1)                            SW11=4000               ;=1,INHIBIT ITERATIONS
 (1)                            SW10=2000               ;=1,ESCAPE TO NEXT TEST ON ERROR
 (1)                            SW09=1000               ;=1,LOOP WITH CURRENT DATA
 (1)                            SW08=400                ;=1,LOOP ON ERROR
 (1)                            SW07=200                ;=1, DO ''AUTO SIZING'' ON INITAL START UP.
 (1)                            SW06=100                ;=1, DESELECT SPECIFIC DEVICES
 (1)                                                    ;NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT
 (1)                            SW05=40
 (1)                            SW04=20                 ;=1, SELECT DELAY PARAMETER
 (1)                            SW03=10                 ;=1, SELECT SPECIFIC PARAMETERS
 (1)                            SW02=4                  ;=1, LOCK ON TEST SELECT
 (1)                            SW01=2                  ;=1, RESTART PROGRAM AT SELECTED TEST
 (1)                            SW00=1                  ;=1, SELECT DEVICE ADDRESS, VECTOR, ETC.
 (1)                            !
 (2)                            .SBTTL  BASIC DEFINITIONS
 (2)
 (2)                            ;*INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***
 (2)          001120            STACK=  1120
 (2)                            .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
 (2)                            .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
 (2)
 (2)                            ;*MISCELLANEOUS DEFINITIONS
 (2)          000011            HT=     11              ;;CODE FOR HORIZONTAL TAB
 (2)          000012            LF=     12              ;;CODE FOR LINE FEED
 (2)          000015            CR=     15              ;;CODE FOR CARRIAGE RETURN
```

```
(2)          000200                CRLF=    200            ;;CODE FOR CARRIAGE RETURN-LINE FEED
(2)          177776                PS=      177776         ;;PROCESSOR STATUS WORD
(2)                                .EQUIV   PS,PSW
(2)          177774                STKLMT=  177774         ;;STACK LIMIT REGISTER
(2)          177772                PIRQ=    177772         ;;PROGRAM INTERRUPT REQUEST REGISTER
(2)          177570                DSWR=    177570         ;;HARDWARE SWITCH REGISTER
(2)          177570                DDISP=   177570         ;;HARDWARE DISPLAY REGISTER
(2)
(2)                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
(2)          000000                R0=      %0             ;;GENERAL REGISTER
(2)          000001                R1=      %1             ;;GENERAL REGISTER
(2)          000002                R2=      %2             ;;GENERAL REGISTER
(2)          000003                R3=      %3             ;;GENERAL REGISTER
(2)          000004                R4=      %4             ;;GENERAL REGISTER
(2)          000005                R5=      %5             ;;GENERAL REGISTER
(2)          000006                R6=      %6             ;;GENERAL REGISTER
(2)          000007                R7=      %7             ;;GENERAL REGISTER
(2)          000006                SP=      %6             ;;STACK POINTER
(2)          000007                PC=      %7             ;;PROGRAM COUNTER
(2)
(2)                                ;*PRIORITY LEVEL DEFINITIONS
(2)          000000                PR0=     0              ;;PRIORITY LEVEL 0
(2)          000040                PR1=     40             ;;PRIORITY LEVEL 1
(2)          000100                PR2=     100            ;;PRIORITY LEVEL 2
(2)          000140                PR3=     140            ;;PRIORITY LEVEL 3
(2)          000200                PR4=     200            ;;PRIORITY LEVEL 4
(2)          000240                PR5=     240            ;;PRIORITY LEVEL 5
(2)          000300                PR6=     300            ;;PRIORITY LEVEL 6
(2)          000340                PR7=     340            ;;PRIORITY LEVEL 7
(2)
(2)                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
(2)          100000                SW15=    100000
(2)          040000                SW14=    40000
(2)          020000                SW13=    20000
(2)          010000                SW12=    10000
(2)          004000                SW11=    4000
(2)          002000                SW10=    2000
(2)          001000                SW09=    1000
(2)          000400                SW08=    400
(2)          000200                SW07=    200
(2)          000100                SW06=    100
(2)          000040                SW05=    40
(2)          000020                SW04=    20
(2)          000010                SW03=    10
(2)          000004                SW02=    4
(2)          000002                SW01=    2
(2)          000001                SW00=    1
(2)                                .EQUIV   SW09,SW9
(2)                                .EQUIV   SW08,SW8
(2)                                .EQUIV   SW07,SW7
(2)                                .EQUIV   SW06,SW6
(2)                                .EQUIV   SW05,SW5
(2)                                .EQUIV   SW04,SW4
(2)                                .EQUIV   SW03,SW3
(2)                                .EQUIV   SW02,SW2
(2)                                .EQUIV   SW01,SW1
```

```
(2)                             .EQUIV  SW00,SW0
(2)
(2)                             ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(2)             100000          BIT15=  100000
(2)             040000          BIT14=  40000
(2)             020000          BIT13=  20000
(2)             010000          BIT12=  10000
(2)             004000          BIT11=  4000
(2)             002000          BIT10=  2000
(2)             001000          BIT09=  1000
(2)             000400          BIT08=  400
(2)             000200          BIT07=  200
(2)             000100          BIT06=  100
(2)             000040          BIT05=  40
(2)             000020          BIT04=  20
(2)             000010          BIT03=  10
(2)             000004          BIT02=  4
(2)             000002          BIT01=  2
(2)             000001          BIT00=  1
(2)                             .EQUIV  BIT09,BIT9
(2)                             .EQUIV  BIT08,BIT8
(2)                             .EQUIV  BIT07,BIT7
(2)                             .EQUIV  BIT06,BIT6
(2)                             .EQUIV  BIT05,BIT5
(2)                             .EQUIV  BIT04,BIT4
(2)                             .EQUIV  BIT03,BIT3
(2)                             .EQUIV  BIT02,BIT2
(2)                             .EQUIV  BIT01,BIT1
(2)                             .EQUIV  BIT00,BIT0
(2)
(2)                             ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(2)             000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(2)             000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(2)             000014          TBITVEC=14              ;;"T" BIT
(2)             000014          TRTVEC= 14              ;;TRACE TRAP
(2)             000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(2)             000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(2)             000024          PWRVEC= 24              ;;POWER FAIL
(2)             000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(2)             000034          TRAPVEC=34              ;;"TRAP" TRAP
(2)             000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
(2)             000064          TPVEC=  64              ;;TTY PRINTER VECTOR
(2)             000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
(1)
(1)
(1)                             ;INSTRUCTION DEFINITIONS
(1)                             ;----------------------
(1)
(1)             005746          PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
(1)             005726          POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
(1)             010046          PUSHR0=10046    ;SAVE R0 ON STACK
(1)             012600          POPR0=12600     ;RESTORE R0 FROM STACK
(1)             024646          PUSH2SP=24646   ;DECREMENT STACK TWICE
(1)             022626          POP2SP=22626    ;INCREMENT STACK TWICE
(1)             000200          MASK=BIT7       ;SET INTERRUPT MASK (INHIBIT FURTHER INTERRUPTS)
(1)             000000          CLEAR=0         ;ALLOW INTERRUPTS (CLEAR PROCESSOR STATUS)
```

```
(1)
(1)
(1)                                     ;DZV11 CONTROL AND STATUS REGISTER DEFINITIONS
(1)                                     ;(DZVCSR)       BIT DEFINITIONS
(1)                                     ;-----------------------------
(1)
(1)          000010                     MAINT = BIT3     ;MAINTENANCE MODE ENABLE
(1)          000020                     DCLR=BIT4        ;DEVICE CLEAR
(1)          000040                     MSENAB=BIT5      ;MASTER SCAN ENABLE
(1)          000100                     RIE=BIT6         ;RECEIVER INTERRUPT ENABLE
(1)          000200                     RDONE=BIT7       ;RECEIVER DONE
(1)          010000                     SILOEN= BIT12    ;SILO ALARM ENABLE
(1)          020000                     SILOAL = BIT13   ;SILO ALARM
(1)          040000                     TIE=BIT14        ;TRANSMITTER INTERRUPT ENABLE
(1)          100000                     TRDY=BIT15       ;TRANSMITTER READY
(1)
(1)                                     ;DZVCSR WORD DEFINITIONS
(1)                                     ;----------------------
(1)          000000                     TL0=0            ;TRANSMIT LINE 0
(1)          000400                     TL1=BIT8         ;TRANSMIT LINE 1
(1)          001000                     TL2=BIT9         ;TRANSMIT LINE 2
(1)          001400                     TL3=BIT9!BIT8    ;TRANSMIT LINE 3
(1)
(1)
(1)                                     ;DZVRBUF BIT DEFINITIONS
(1)                                     ;----------------------
(1)
(1)          010000                     PARER=BIT12      ;PARITY ERROR
(1)          020000                     FRMERR=BIT13     ;FRAME ERROR
(1)          040000                     OVRRUN=BIT14     ;OVERRUN ERROR
(1)          100000                     DVALID=BIT15     ;DATA VALID
(1)
(1)                                     ;DZVRBUF WORD DEFINITIONS
(1)                                     ;-----------------------
(1)
(1)          000000                     RL0=0            ;RECEIVER LINE 0
(1)          000400                     RL1=BIT8         ;RECEIVER LINE 1
(1)          001000                     RL2=BIT9         ;RECEIVER LINE 2
(1)          001400                     RL3=BIT9!BIT8    ;RECEIVER LINE 3
(1)
(1)                                     ;DZVLPR WORD DEFINITIONS
(1)                                     ;----------------------
(1)
(1)          000000                     LP0=0            ;LINE PARAMETER 0
(1)          000001                     LP1=BIT0         ;LINE PARAMETER 1
(1)          000002                     LP2=BIT1         ;LINE PARAMETER 2
(1)          000003                     LP3=BIT1!BIT0    ;LINE PARAMETER 3
(1)
(1)          000000                     FIVE=0           ;FIVE BITS/CHAR,1 STOP BIT
(1)          000010                     SIX=BIT3         ;SIX BITS/CHAR,1 STOP BIT
(1)          000020                     SEVEN=BIT4       ;SEVEN BITS/CHAR,1 STOP BIT
(1)          000030                     EIGHT=BIT4!BIT3  ;EIGHT BITS/CHAR,1 STOP BIT
(1)          000040                     FIVES=BIT5       ;FIVE BITS/CHAR,2 STOP BITS
(1)          000050                     SIXS=BIT5!BIT3   ;SIX BITS/CHAR,2 STOP BITS
(1)          000060                     SEVENS=BIT5!BIT4         ;SEVEN BITS/CHAR, 2 STOP BITS
(1)          000070                     EIGHTS=BIT5!BIT4!BIT3    ;EIGHT BITS/CHAR, 2 STOP BITS
```

```
(1)
(1)          000100                  PARITY=BIT6             ;PARITY ENABLED
(1)          000200                  ODDPAR=BIT7             ;ODD PARITY ENABLED
(1)          000000                  ONESTOP=0               ;ONE STOP BIT ENABLED
(1)          000040                  TWOSTOP=BIT5            ;TWO STOP BITS ENABLED
(1)          000000                  EVEPAR=0                ;EVEN PARITY ENABLED
(1)          010000                  RCVON=BIT12             ;ENABLE RECEIVER (RECEIVER ON)
(1)
(1)          000000                  S50=0                   ;SPEED 50 BAUD
(1)          000400                  S75=BIT8                ;SPEED 75 BAUD
(1)          001000                  S110=BIT9               ;SPEED 110 BAUD
(1)          001400                  S134=BIT9!BIT8          ;SPEED 134.5 BAUD
(1)          002000                  S150=BIT10              ;SPEED 150 BAUD
(1)          002400                  S300=BIT10!BIT8         ;SPEED 300 BAUD
(1)          003000                  S600=BIT10!BIT9         ;SPEED 600 BAUD
(1)          003400                  S1200=BIT10!BIT9!BIT8   ;SPEED 1200 BAUD
(1)          004000                  S1800=BIT11             ;SPEED 1800 BAUD
(1)          004400                  S2000=BIT11!BIT8        ;SPEED 2000 BAUD
(1)          005000                  S2400=BIT11!BIT9        ;SPEED 2400 BAUD
(1)          005400                  S3600=BIT11!BIT9!BIT8   ;SPEED 3600 BAUD
(1)          006000                  S4800=BIT11!BIT10       ;SPEED 4800 BAUD
(1)          006400                  S7200=BIT11!BIT10!BIT8  ;SPEED 7200 BAUD
(1)          007000                  S9600=BIT11!BIT10!BIT9  ;SPEED 9600 BAUD
(1)          007400                  S19200=BIT11!BIT10!BIT9!BIT8    ;SPEED 19200 BAUD
(1)
(1)                                  ;DZVTCR BIT DEFINITIONS
(1)                                  ;----------------------
(1)          000001                  TCR0=BIT0               ;ENABLE TRANSMISSION ON LINE 0
(1)          000002                  TCR1=BIT1               ;ENABLE TRANSMISSION ON LINE 1
(1)          000004                  TCR2=BIT2               ;ENABLE TRANSMISSION ON LINE 2
(1)          000010                  TCR3=BIT3               ;ENABLE TRANSMISSION ON LINE 3
(1)          000400                  DTR0=BIT8               ;DATA TERMINAL READY FOR LINE 0
(1)          001000                  DTR1=BIT9               ;DATA TERMINAL READY FOR LINE 1
(1)          002000                  DTR2=BIT10              ;DATA TERMINAL READY FOR LINE 2
(1)          004000                  DTR3=BIT11              ;DATA TERMINAL READY FOR LINE 3
(1)
(1)                                  ;DZVMSR BIT DEFINITIONS
(1)                                  ;----------------------
(1)          000001                  RING0=BIT0              ;RING INDICATED ON LINE 0
(1)          000002                  RING1=BIT1              ;RING INDICATED ON LINE 1
(1)          000004                  RING2=BIT2              ;RING INDICATED ON LINE 2
(1)          000010                  RING3=BIT3              ;RING INDICATED ON LINE 3
(1)          000400                  CO0=BIT8                ;CARRIER PRESENT ON LINE 0
(1)          001000                  CO1=BIT9                ;CARRIER PRESENT ON LINE 1
(1)          002000                  CO2=BIT10               ;CARRIER PRESENT ON LINE 2
(1)          004000                  CO3=BIT11               ;CARRIER PRESENT ON LINE 3
(1)
(1)                                  ;DZVTDR BIT DEFINITIONS
(1)                                  ;----------------------
(1)
(1)          000400                  BRK0=BIT8               ;BREAK FOR LINE 0
(1)          001000                  BRK1=BIT9               ;BREAK FOR LINE 1
(1)          002000                  BRK2=BIT10              ;BREAK FOR LINE 2
(1)          004000                  BRK3=BIT11              ;BREAK FOR LINE 3
```

```
(1)
(1)                             ;TABLE OF LOOP AROUND FUNCTIONS (H325)
(1)                             ;
(1)                             ;       ----------------
(1)                             ;       I               ^
(1)                             ;       V               ^
(1)                             ;       REC             TRANS
(1)                             ;       DATA            DATA
(1)                             ;
(1)                             ;       ----------------
(1)                             ;       I               ^
(1)                             ;       V               ^
(1)                             ;       CO              RTS
(1)                             ;
(1)                             ;       ----------------
(1)                             ;       I               ^
(1)                             ;       V               ^
(1)                             ;       RING            DTR
```

```
(1)                             ;:*****************************************************************************
(1)                             ;-------------------------------------------------------------------
(1)                                     ;TRAPCATCHER FOR ILLEGAL INTERRUPTS
(1)                                     ;THE STANDARD "TRAP CATCHER" IS PLACED
(1)                                     ;BETWEEN ADDRESS 0 TO ADDRESS 776.
(1)                                     ;IT LOOKS LIKE "PC+2 HALT".
(1)                             ; ------------------------------------------------------------------------
(1)                             ;:*****************************************************************************
(1)
(1)             000000          .=0
(1)                                     ;STANDARD INTERRUPT VECTORS
(1)                                     ;-------------------------
(1)
(1)             000020          .=20
(1)     000020  004404                  .SCOPE                  ;SCOPE LOOP HANDLER
(1)     000022  000200                  MASK                    ;HANDLE AT PRIORITY 7
(1)     000024  007414                  $PWRDN                  ;POWER FAIL HANDLER
(1)     000026  000340                  340                     ;SERVICE AT PRIORITY LEVEL 7
(1)     000030  006522                  $ERROR                  ;ERROR HANDLER
(1)     000032  000340                  340                     ;SERVICE AT PRIORITY LEVEL 7
(1)     000034  006314                  .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
(1)     000036  000340                  340                     ;SERVICE AT PRIORITY LEVEL 7
(2)                             .SBTTL  ACT11 HOOKS
(2)
(3)                             ;;*****************************************************************************
(2)                             ;HOOKS REQUIRED BY ACT11
(2)             000040                  $SVPC=.                 ;SAVE PC
(2)             000046                  .=46
(2)     000046  004340                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(2)             000052                  .=52
(2)     000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
(2)             000040                  .=$SVPC                 ;; RESTORE PC
(1)
(1)             000174          .=174
(1)     000174  000000          DISPREG:0                       ;SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 11S
(1)     000176  000000          SWREG:  0                       ;SOFTARE SWITCH REGISTER FOR SWITCHLESS 11S
(1)             000200          .=200
(1)     000200  000137  002116          JMP     .START          ;GO TO START OF PROGRAM
(1)
(2)
(2)             001000          .=1000
(2)     001000  005200  053103  055104  MTITLE: .ASCIZ  <200><12>/CVDZAC/<200>/FOUR LINE ASYNC MUX TESTS, PART 1 OF 2/<200>
(2)
```

```
(3)          001120                          .=1120
(4)                                          ;;*****************************************************************
(4)                                          .SBTTL  APT MAILBOX-ETABLE
(4)
(5)                                          ;;*****************************************************************
(4)                                          .EVEN
(4)  001120                                  $MAIL:                          ;;APT MAILBOX
(4)  001120  000000                          $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
(4)  001122  000000                          $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
(4)  001124  000000                          $TESTN: .WORD   ATESTN   ;;TEST NUMBER
(4)  001126  000000                          $PASS:  .WORD   APASS    ;;PASS COUNT
(4)  001130  000000                          $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
(4)  001132  000000                          $UNIT:  .WORD   AUNIT    ;;I/O UNIT NUMBER
(4)  001134  000000                          $MSGAD: .WORD   AMSGAD   ;;MESSAGE ADDRESS
(4)  001136  000000                          $MSGLG: .WORD   AMSGLG   ;;MESSAGE LENGTH
(4)  001140                                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
(4)  001140      000                         $ENV:   .BYTE   AENV     ;;ENVIRONMENT BYTE
(4)  001141      000                         $ENVM:  .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
(4)  001142  000000                          $SWREG: .WORD   ASWREG   ;;APT SWITCH REGISTER
(4)  001144  000000                          $USWR:  .WORD   AUSWR    ;;USER SWITCHES
(4)  001146  000000                          $CPUOP: .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
(4)                                          ;*                              BITS 15-11=CPU TYPE
(4)                                          ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(4)                                          ;*                                      11/70=06,PDQ=07,Q=10
(4)                                          ;*                              BIT 10=REAL TIME CLOCK
(4)                                          ;*                              BIT  9=FLOATING POINT PROCESSOR
(4)                                          ;*                              BIT  8=MEMORY MANAGEMENT
(4)  001150      000                         $MAMS1: .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
(4)  001151      000                         $MTYP1: .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
(4)                                          ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
(4)                                          ;*                                      900 NSEC CORE=001
(4)                                          ;*                                      300 NSEC BIPOLAR=002
(4)                                          ;*                                      500 NSEC MOS=003
(4)  001152  000000                          $MADR1: .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
(4)                                          ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(4)  001154      000                         $MAMS2. .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
(4)  001155      000                         $MTYP2. .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
(4)  001156  000000                          $MADR2: .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
(4)  001160      000                         $MAMS3: .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
(4)  001161      000                         $MTYP3: .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
(4)  001162  000000                          $MADR3: .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
(4)  001164      000                         $MAMS4: .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
(4)  001165      000                         $MTYP4: .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
(4)  001166  000000                          $MADR4: .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
(4)  001170  000300                          $VECT1: .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(4)  001172  000000                          $VECT2: .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(4)  001174  160010                          $BASE:  .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(4)  001176  000001                          $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
(4)  001200  000017                          $CDW1:  .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
(4)  001202  000000                          $CDW2:  .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
(4)  001204  017470                          $DDW0:  .WORD   ADDW0    ;;DEVICE DESCRIPTOR WORD#0
(4)  001206  017470                          $DDW1:  .WORD   ADDW1    ;;DEVICE DESCRIPTOR WORD#1
(4)  001210  017470                          $DDW2:  .WORD   ADDW2    ;;DEVICE DESCRIPTOR WORD#2
(4)  001212  017470                          $DDW3:  .WORD   ADDW3    ;;DEVICE DESCRIPTOR WORD#3
(4)  001214  017470                          $DDW4:  .WORD   ADDW4    ;;DEVICE DESCRIPTOR WORD#4
(4)  001216  017470                          $DDW5:  .WORD   ADDW5    ;;DEVICE DESCRIPTOR WORD#5
```

```
   (4)  001220  017470                 $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
   (4)  001222  017470                 $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
   (4)  001224  017470                 $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
   (4)  001226  017470                 $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
   (4)  001230  017470                 $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
   (4)  001232  017470                 $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
   (4)  001234  017470                 $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
   (4)  001236  017470                 $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
   (4)  001240  017470                 $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
   (4)  001242  017470                 $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
   (4)
   (4)
   (4)  001244                         $ETEND:
   (4)
```

```
    (3)                                 .SBTTL  COMMON TAGS
    (3)
    (4)                                 ;;*****************************************************************
    (3)                                 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
    (3)                                 ;*USED IN THE PROGRAM.
    (3)
    (3)  001244                         $CMTAG:                         ;;START OF COMMON TAGS
    (3)  001244  000000                         .WORD   0
    (3)  001246     000                 $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
    (3)  001247     000                 $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
    (3)  001250  000000                 $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
    (3)  001252  000000                 $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
    (3)  001254  000000                 $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
    (3)  001256  000000                 $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
    (3)  001260     000                 $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
    (3)  001261     001                 $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
    (3)  001262  000000                 $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
    (3)  001264  000000                 $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
    (3)  001266  000000                 $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
    (3)  001270  000000                 $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
    (3)  001272  000000                 $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
    (3)  001274  000000                         .WORD   0               ;;RESERVED--NOT TO BE USED
    (3)  001276  000000                         .WORD   0
    (3)  001300     000                 $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
    (3)  001301     000                 $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
    (3)  001302  000000                         .WORD   0
    (3)  001304  177570                 SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
    (3)  001306  177570                 DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
    (3)  001310  177560                 $TKS:   177560                  ;;TTY KBD STATUS
    (3)  001312  177562                 $TKB:   177562                  ;;TTY KBD BUFFER
    (3)  001314  177564                 $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
    (3)  001316  177566                 $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
    (3)  001320     000                 $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
    (3)  001321     002                 $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
    (3)  001322     012                 $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
    (3)  001323     000                 $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
    (3)  001324  000000                 $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
    (3)                                                                 ;;WHICH  ($REG0) WAS OBTAINED
    (5)  001326  000000                 $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
    (5)  001330  000000                 $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
    (5)  001332  000000                 $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
    (5)  001334  000000                 $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
    (5)  001336  000000                 $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
    (5)  001340  000000                 $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
    (5)  001342  000000                 $TMP0:  .WORD   0               ;;USER DEFINED
    (5)  001344  000000                 $TMP1:  .WORD   0               ;;USER DEFINED
    (5)  001346  000000                 $TMP2:  .WORD   0               ;;USER DEFINED
    (5)  001350  000000                 $TMP3:  .WORD   0               ;;USER DEFINED
    (5)  001352  000000                 $TMP4:  .WORD   0               ;;USER DEFINED
    (3)  001354  000000                 $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
    (3)  001356     077                 $QUES:  .ASCII  /?/             ;;QUESTION MARK
    (3)  001357     015                 $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
    (3)  001360  000012                 $LF:    .ASCIZ  <12>            ;;LINE FEED
```

```
(3)                                 .SBTTL  ERROR POINTER TABLE
(3)
(3)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(3)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(3)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(3)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(3)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(3)
(3)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(3)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
(3)                                 ;*      DT              ;;POINTS TO THE DATA
(3)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
(3)
(3)
(3)  001362                         $ERRTB:
(2)
(2)                                         ;PROGRAM CONTROL PARAMETERS
(2)                                         ;-------------------------
(2)
(2)  001362  000000                 NEXT:   0                       ;ADDRESS OF NEXT TEST TO BE EXECUTED
(2)  001364  000000                 LOCK:   0                       ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP
(2)
(2)                                         ;PROGRAM VARIABLES
(2)                                         ;-----------------
(2)
(2)  001366  000017                 LINE:   17                      ;DEFAULT ALL FOUR LINES RUNNING
(2)  001370  017470                 PAR:    17470                   ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NO PARIT
(2)  001372  000000                 MODE:   0                       ;DEFAULT MAINTENANCE MODE
(2)  001374  000000                 SAVLIN: 0                       ;LINE NUMBER
(2)  001376  000000                 XMTLIN: 0                       ;TRANSMISSION LINE NUMBER
(2)  001400  000000                 XMTCNT: 0                       ;COUNT OF WORDS IN A TRANSMISSION PATTERN
(2)  001402  000000                 REGIST: 0                       ;DEVICE ADDRESS STORAGE LOCATION
(2)  001404  000000                 SAVPC:  0                       ;PROGRAM COUNTER STORAGE
(2)  001406  000001                 DZVACTV:.BLKW   1               ;*DZV11'S SELECTED ACTIVE.
(2)  001410  000001                 SAVACTV:.BLKW   1               ;*A BIT MAP OF DZV11'S IN THE SYSTEM
(2)  001412  000001                 RUN:    1                       ;*POINTER ONE PAST RUNNING DEVICE.
(2)  001414  000001                 DZVNUM: .BLKB 1                 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM
(2)  001415     001                 SAVNUM: .BYTE 1                 ;*WORKABLE NUMBER.
(2)  001416  000001                 SAVNO:  .BLKB   1               ;*OCTAL NO. OF DZV11'S BEING TESTED
(2)          001420                 .EVEN
(2)  001420  001500                 ACTIVE: DZV.MAP                 ;TABLE POINTER.
```

```
(2)
(2)                                             ;PROGRAM CONTROL FLAGS
(2)                                             ;--------------------
(2)
(2)  001422      000                    INIFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG
(2)  001423      000                    HDRFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
(2)  001424      000                    MNTFLG: .BYTE   0               ;MAINTENANCE BIT SET FLAG
(2)  001425      000                    DONFLG: .BYTE   0               ;TRANSMISSION COMPLETION FLAG
(2)                                     .EVEN
(2)                                     ;DATA VARIABLES
(2)  001426  000000                    TD0:    .WORD   0
(2)  001430  000000                    TD1:    .WORD   0
(2)  001432  000000                    TD2:    .WORD   0
(2)  001434  000000                    TD3:    .WORD   0
(2)  001436  000000                    TR0:    .WORD   0
(2)  001440  000000                    TR1:    .WORD   0
(2)  001442  000000                    TR2:    .WORD   0
(2)  001444  000000                    TR3:    .WORD   0
(2)  001446                             STOP:
(2)                                     .SBTTL  APT PARAMETER BLOCK
(2)
(3)                                     ;;******************************************************************
(2)                                     ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(3)                                     ;;******************************************************************
(2)          001446                             .$X=.   ;;SAVE CURRENT LOCATION
(2)          000024                             .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(2)  000024  000200                             200     ;;FOR APT START UP
(2)          000044                             .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
(2)  000044  001446                             $APTHDR ;;POINT TO APT HEADER BLOCK
(2)          001443                             .=.$X   ;;RESET LOCATION COUNTER
(3)                                     ;;******************************************************************
(2)                                     ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(2)                                     ;INTERFACE SPEC.
(2)
(2)  001446                             $APTHD:
(2)  001446  000000                    $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(2)  001450  001120                    $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(2)  001452  000120                    $TSTM:  .WORD   80.     ;;RUN TIM OF LONGEST TEST
(2)  001454  000024                    $PASTM: .WORD   20.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(2)  001456  000000                    $UNITM: .WORD   0.      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(2)  001460  000052                            .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
(1)                                             ;DZV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
(1)                                             ;-----------------------------------------
(1)
(1)          001500                     .=1500
(1)  001500                             DZV.MAP:
(3)
(3)  001500  000001                    DZCR0:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 0
(3)  001502  000001                    DZVC0:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 0
(3)  001504  000001                    LINE0:  .BLKW   1       ;ALL LINES SELECTED
(3)  001506  000001                    PAR0:   .BLKW   1       ;PARAMETERS
(3)  001510  000001                    MANT0:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
(3)
(3)  001512  000001                    DZCR1:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 1
(3)  001514  000001                    DZVC1:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 1
(3)  001516  000001                    LINE1:  .BLKW   1       ;ALL LINES SELECTED
```

```
    (3)  001520  000001                PAR1:    .BLKW   1       ;PARAMETERS
    (3)  001522  000001                MANT1:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001524  000001                DZCR2:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 2
    (3)  001526  000001                DZVC2:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 2
    (3)  001530  000001                LINE2:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001532  000001                PAR2:    .BLKW   1       ;PARAMETERS
    (3)  001534  000001                MANT2:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001536  000001                DZCR3:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 3
    (3)  001540  000001                DZVC3:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 3
    (3)  001542  000001                LINE3:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001544  000001                PAR3:    .BLKW   1       ;PARAMETERS
    (3)  001546  000001                MANT3:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001550  000001                DZCR4:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 4
    (3)  001552  000001                DZVC4:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 4
    (3)  001554  000001                LINE4:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001556  000001                PAR4:    .BLKW   1       ;PARAMETERS
    (3)  001560  000001                MANT4:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001562  000001                DZCR5:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 5
    (3)  001564  000001                DZVC5:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 5
    (3)  001566  000001                LINE5:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001570  000001                PAR5:    .BLKW   1       ;PARAMETERS
    (3)  001572  000001                MANT5:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001574  000001                DZCR6:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 6
    (3)  001576  000001                DZVC6:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 6
    (3)  001600  000001                LINE6:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001602  000001                PAR6:    .BLKW   1       ;PARAMETERS
    (3)  001604  000001                MANT6:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001606  000001                DZCR7:   .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 7
    (3)  001610  000001                DZVC7:   .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 7
    (3)  001612  000001                LINE7:   .BLKW   1       ;ALL LINES SELECTED
    (3)  001614  000001                PAR7:    .BLKW   1       ;PARAMETERS
    (3)  001616  000001                MANT7:   .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001620  000001                DZCR10:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 10
    (3)  001622  000001                DZVC10:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 10
    (3)  001624  000001                LINE10:  .BLKW   1       ;ALL LINES SELECTED
    (3)  001626  000001                PAR10:   .BLKW   1       ;PARAMETERS
    (3)  001630  000001                MANT10:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001632  000001                DZCR11:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 11
    (3)  001634  000001                DZVC11:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 11
    (3)  001636  000001                LINE11:  .BLKW   1       ;ALL LINES SELECTED
    (3)  001640  000001                PAR11:   .BLKW   1       ;PARAMETERS
    (3)  001642  000001                MANT11:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001644  000001                DZCR12:  .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 12
    (3)  001646  000001                DZVC12:  .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 12
    (3)  001650  000001                LINE12:  .BLKW   1       ;ALL LINES SELECTED
    (3)  001652  000001                PAR12:   .BLKW   1       ;PARAMETERS
    (3)  001654  000001                MANT12:  .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
```

```
    (3)
    (3)  001656  000001                DZCR13: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 13
    (3)  001660  000001                DZVC13: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 13
    (3)  001662  000001                LINE13: .BLKW   1       ;ALL LINES SELECTED
    (3)  001664  000001                PAR13:  .BLKW   1       ;PARAMETERS
    (3)  001666  000001                MANT13: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001670  000001                DZCR14: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 14
    (3)  001672  000001                DZVC14: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 14
    (3)  001674  000001                LINE14: .BLKW   1       ;ALL LINES SELECTED
    (3)  001676  000001                PAR14:  .BLKW   1       ;PARAMETERS
    (3)  001700  000001                MANT14: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001702  000001                DZCR15: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 15
    (3)  001704  000001                DZVC15: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 15
    (3)  001706  000001                LINE15: .BLKW   1       ;ALL LINES SELECTED
    (3)  001710  000001                PAR15:  .BLKW   1       ;PARAMETERS
    (3)  001712  000001                MANT15: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001714  000001                DZCR16: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 16
    (3)  001716  000001                DZVC16: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 16
    (3)  001720  000001                LINE16: .BLKW   1       ;ALL LINES SELECTED
    (3)  001722  000001                PAR16:  .BLKW   1       ;PARAMETERS
    (3)  001724  000001                MANT16: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (3)
    (3)  001726  000001                DZCR17: .BLKW   1       ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 17
    (3)  001730  000001                DZVC17: .BLKW   1       ;RECEIVER AND BASE VECTOR  FOR DZV11 NUMBER 17
    (3)  001732  000001                LINE17: .BLKW   1       ;ALL LINES SELECTED
    (3)  001734  000001                PAR17:  .BLKW   1       ;PARAMETERS
    (3)  001736  000001                MANT17: .BLKW   1       ;MAINTENANCE MODE FOR THIS DEVICE
    (1)
    (1)  001740  177777                DZV.END:        177777
```

```
(1)                                     ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
(1)                                     ;POINTERS TO SUBROUTINES CAN BE FOUND
(1)                                     ;IN THE TABLE IMMEDIATELY FOLLOWING THE DEFINITIONS
(1)
(1)                                     ;;*************************************************************************
(1)                                     ;;-------------------------------------------------------------------------
(1)  001742                             .TRPTAB:
(3)          104400                     ADVANCE=TRAP+0          ;CALL TO ADVANCE TO NEXT TEST( OR SCOPE THIS ONE)
(2)  001742  006410                             .ADVANCE
(3)          104401                     SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
(2)  001744  004650                             .SCOP1
(3)          104402                     TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
(2)  001746  004674                             .TYPE
(3)          104403                     INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
(2)  001750  005514                             .INSTR
(3)          104404                     INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
(2)  001752  005620                             .INSTER
(3)          104405                     PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
(2)  001754  005640                             .PARAM
(3)          104406                     SETFLG=TRAP+6           ;CALL TO  SET FLAG ROUTINE
(2)  001756  010252                             .SETFLG
(3)          104407                     SAV05=TRAP+7            ;CALL TO REGISTER SAVE ROUTINE
(2)  001760  006040                             .SAV05
(3)          104410                     RES05=TRAP+10           ;CALL TO REGISTER RESTORE ROUTINE
(2)  001762  006100                             .RES05
(3)          104411                     CONVRT=TRAP+11          ;CALL TO DATA OUTPUT ROUTINE
(2)  001764  006132                             .CONVRT
(3)          104412                     CNVRT=TRAP+12           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
(2)  001766  006136                             .CNVRT
(3)          104413                     DEVICE.CLR=TRAP+13              ;CALL TO ISSUE A DEVICE CLEAR
(2)  001770  006336                             .DEVICE.CLR
(3)          104414                     DELAY=TRAP+14           ;CALL TO DELAY FOR FAST CPU'S
(2)  001772  006370                             .DELAY
(3)          104415                     PARMD=TRAP+15           ;CONVERT DECIMAL STRING TO OCTAL
(2)  001774  011256                             .PARMD
(3)          104416                     PAWCH=TRAP+16           ;SET FLAG    ECHO OR  CABLE
(2)  001776  010372                             .PAWCH
(3)          104417                     DCLASM=TRAP+17          ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2)  002000  006356                             .DCLASM
(3)          104420                     SHIFT=TRAP+20           ;CALL TO ROTATE LINE POINTER
(2)  002002  006422                             .SHIFT
(3)          104421                     LPRSET=TRAP+21          ;CALL TO SET UP LPR DEVICE REGISTER
(2)  002004  006440                             .LPRSET
(3)          104422                     BUFSET=TRAP+22          ;CALL TO ZERO BUFFER AREA
(2)  002006  006500                             .BUFSET
(1)
(1)                                     ;-------------------------------------------------------------------------
(1)                                     ;;*************************************************************************
```

```
(1)                                     ;DZV11 VECTOR AND REGISTER INDIRECT POINTERS
(1)                                     ;WORKING AREA
(1)
(1)  002010  160040                     DZVCSR: 160040  ;R/W
(1)  002012  160041                     HDZVCSR:160041  ;R/W
(1)  002014  160042                     DZVRBUF:160042  ;READ ONLY
(1)  002016  160043                     HDZVRBUF:160043 ;READ ONLY
(1)  002020  160042                     DZVLPR: 160042  ;WRITE ONLY
(1)  002022  160043                     HDZVLPR:160043  ;WRITE ONLY
(1)  002024  160044                     DZVTCR: 160044  ;R/W
(1)  002026  160045                     HDZVTCR:160045  ;R/W
(1)  002030  160046                     DZVMSR: 160046  ;READ ONLY
(1)  002032  160047                     HDZVMSR:160047  ;READ ONLY
(1)  002034  160046                     DZVTDR: 160046  ;WRITE ONLY
(1)  002036  160047                     HDZVTDR:160047  ;WRITE ONLY
(1)
(1)                                     ;DEFAULT DZV VECTORS
(1)
(1)  002040  000300                     DZVRIV: 300     ;REC INTR VECTOR
(1)  002042  000302                     DZVRIS: 302     ;REC INTR STATUS
(1)  002044  000304                     DZVTIV: 304     ;XMIT INTR VECTOR
(1)  002046  000306                     DZVTIS: 306     ;XMIT INTR STATUS
(1)
(1)
```

```
    (1)
    (1)                                     ;TIME TABLE FOR RELATIVE TIMING TESTS
    (1)                                     ;-----------------------------------
    (1)
    (1)  002050                             TMTBL:
    (1)  002050  000000                     T50:    0
    (1)  002052  000000                     T75:    0
    (1)  002054  000000                     T110:   0
    (1)  002056  000000                     T134:   0
    (1)  002060  000000                     T150:   0
    (1)  002062  000000                     T300:   0
    (1)  002064  000000                     T600:   0
    (1)  002066  000000                     T1200:  0
    (1)  002070  000000                     T1800:  0
    (1)  002072  000000                     T2000:  0
    (1)  002074  000000                     T2400:  0
    (1)  002076  000000                     T3600:  0
    (1)  002100  000000                     T4800:  0
    (1)  002102  000000                     T7200:  0
    (1)  002104  000000                     T9600:  0
    (1)  002106  000000                     TEIGHT:0
    (1)  002110  000000                     TSEVEN: 0
    (1)  002112  000000                     TSIX:   0
    (1)  002114  000000                     TFIVE:  0
```

```
(1)
(1)                                             ;PROGRAM INITIALIZATION
(1)                                             ;LOCK OUT INTERRUPTS
(1)                                             ;SET UP PROCESSOR STACK
(1)                                             ;SET UP POWER FAIL VECTOR
(1)                                             ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
(1)                                             ;TYPE TITLE MESSAGE
(1)
(1)  002116                             .START:
(1)  002116  000005                             RESET                   ;CLEAR THE WORLD. START NEW ENVIRONMENT
(1)  002120  012706  001120                      MOV     #STACK,SP       ;SET UP STACK
(1)  002124  106427  000200                      MTPS    #MASK           ;LOCK OUT INTERRUPTS
(1)  002130  012737  007414  000024              MOV     #$PWRDN,@#24    ;SET UP POWER FAIL VECTOR
(1)  002136  012737  006522  000030              MOV     #$ERROR,EMTVEC  ;SET UP ERROR VECTOR
(1)  002144  012737  000340  00C032              MOV     #340,EMTVEC+2
(1)  002152  005037  001126                      CLR     $PASS           ;CLEAR PASS COUNT
(1)  002156  105037  001247                      CLRB    $ERFLG          ;CLEAR ERROR FLAG
(1)  002162  012737  001500  001420              MOV     #DZV.MAP,ACTIVE ;GET MAP POINTER.
(1)  002170  012737  000001  001412              MOV     #1,RUN          ;POINT POINTER TO FIRST DEVICE.
(1)  002176  005037  001256                      CLR     $ERTTL          ;CLEAR ERROR COUNT
(1)  002202  005037  001262                      CLR     $ERRPC          ;CLEAR LAST ERROR POINTER
(1)  002206  005037  001246                      CLR     $TSTNM          ;SET UP FOR TEST 1
(1)  002212  012737  002116  001252              MOV     #.START,$LPADR  ;SET UP FOR POWER FAIL BEFORE
(1)                                                                      ;TESTING STARTS
(1)                                             ;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
(1)  002220  012737  000176  001304              MOV     #SWREG,SWR      ;POINT TO SOFTWARE SWR
(1)  002226  012737  000174  001306              MOV     #DISPREG,DISPLAY ;POINT TO SOFTWARE DISPLAY REGISTER
(1)  002234  004737  017342                      CALL    FALCON          ; CHECK FOR FALCON (KXT11)      ;;GPA
(1)  002240  001405                              BEQ     1000$           ; BR IF NOT                     ;;GPA
(1)  002242  004737  000570                      CALL    FALCINI         ; YES, INIT FOR FALCON          ;;GPA
(1)  002246  042737  000040  000032              BIC     #40,EMTVEC+2    ; LOWER EMT TO 6.               ;;GPA
(1)  002254                             1000$:                           ;                               ;;GPA
(1)  002254  105737  001422                      TSTB    INIFLG          ;HAVE WE ALREADY BEEN HERE TODAY?
(1)  002260  001010                              BNE     10$             ;IF SO, SKIP PRINTING THE TITLE
(1)  002262  023727  000042  004340              CMP     @#42,#$ENDAD    ;IF RUNNING UNDER ACT
(1)  002270  001402                              BEQ     1$              ;DON'T PRINT TITLE
(1)  002272  104402  001000                      TYPE    ,MTITLE         ;PRINT THE DIAGNOSTIC'S TITLE
(1)  002276  105337  001422             1$:      DECB    INIFLG          ;SET THE ONCE ONLY FLAG
(1)  002302  105737  001141             10$:     TSTB    $ENVM           ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
(1)  002306  100004                              BPL     15$             ;IF NOT, GO CHECK FOR AUTO-SIZING
(1)  002310  004737  011260                      JSR     PC,SETAPT       ;OTHERWISE, GO DO APT SIZING FROM ETABLE
(1)  002314  000137  003630                      JMP     105$            ;GO PRINT DZV STATUS TABLE
(1)  002320  005737  000042             15$:     TST     @#42            ; CHAINED UNDER XXDP ??         ;;GPA
(1)  002324  001404                              BEQ     16$             ; BR IF NOT                     ;;GPA
(1)  002326  004737  011260                      CALL    SETAPT          ; YES, SET-UP FROM ETABLE       ;;GPA
(1)  002332  000137  003630                      JMP     105$            ; AND PROCEED                   ;;GPA
(1)  002336  004737  007176             16$:     CALL    GETSWR          ; GET INITIAL SWITCH SETTING.   ;;GPA
(1)  002342  032777  000001  176734              BIT     #SW00,@SWR      ;RESELECT ?
(1)  002350  001002                              BNE     20$             ;IF YES, GO SET UP THE INFORMATION
(1)  002352  000137  002654                      JMP     55$             ;IF NO, SKIP THE INTERROGATION
(1)  002356  012700  001500             20$:     MOV     #DZV.MAP,R0     ;POINT TO THE BEGINNING OF THE MAP TABLE
(1)  002362  105037  001423                      CLRB    HDRFLG          ;MAKE SURE A MAP GETS PRINTED
(1)  002366  005020                      25$:     CLR     (R0)+           ;CLEAR A TABLE LOCATION
(1)  002370  020027  001740                      CMP     R0,#DZV.END     ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  002374  001374                              BNE     25$             ;IF NOT ,CLEAR THE NEXT LOCATION IN THE TABLE
(1)  002376  105337  001422                      DECB    INIFLG          ;INSURE NO AUTO SIZING IF QUESTIONS ANSWERED!
```

```
(1)
(1)                                     ;THE FOLLOWING ARE PARAMETERS USED TO FILL IN THE MAP
(1)                                     ;TABLE AND SET UP THE DIAGNOSTIC.
(1)
(1)                                             ;GET THE BASE ADDRESS OF THE DZV11'S
(1)             002402                          GETCSR= .               ; POINTER FOR FALCON TWEAKER    ;;GPA
(2)   002402    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002404    003074                          91$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002406    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002410    160000                          160000                  ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002412    163770                          163770                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002414    001500                          DZCR0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002416       007                          .BYTE   7               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002417       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)   002420    013737  001500  001174          MOV     DZCR0,$BASE     ;COPY BASE ADDRESS TO ETABLE
(1)
(1)                                             ;GET THE BASE VECTOR ADDRESS
(1)             002426                          GETVEC= .               ; POINTER FOR FALCON TWEAKER    ;;GPA
(2)   002426    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002430    003140                          92$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002432    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002434    000300                          300                     ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002436    000776                          776                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002440    001502                          DZVC0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002442       003                          .BYTE   3               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002443       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)   002444    013737  001502  001170          MOV     DZVC0,$VECT1    ;COPY VECTOR TO ETABLE
(1)
(1)                                             ;GET THE MODE OF OPERATION (E,I,S)
(2)   002452    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002454    003367                          96$                     ;POINTER TO THE MESSAGE TO BE PRINTED
(2)   002456    104406                          SETFLG                  ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
(2)   002460    001510                          MANT0                   ;THIS IS THE FLAG BEING SETUP
(1)
(1)                                             ;GET THE NUMBER OF DZV11'S RUNNING
(2)   002462    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002464    003324                          95$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002466    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002470    000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002472    000020                          16.                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002474    001344                          $TMP1                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002476       000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002477       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)
(1)   002500    012737  000017  001504          MOV     #17,LINE0       ;SET UP DEFAULT LINES
(1)   002506    012737  017470  001506          MOV     #17470,PAR0     ;SET UP DEFAULT LPR PARAMETER
(1)                                             ;RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
(1)   002514    032777  000010  176562          BIT     #SW03,@SWR      ;DO YOU WANT PARAMETERS?
(1)   002522    001402                          BEQ     30$             ;IF NO, SKIP THE PARAMETER CALL
(1)   002524    004737  002704                  JSR     PC,65$          ;GET PARAMETERS
(1)   002530    012737  000001  001410  30$:    MOV     #1,SAVACTV      ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
(1)   002536    113737  001344  001414          MOVB    $TMP1,DZVNUM    ;COPY THE NUMBER OF DEVICES
(1)   002544    005337  001344          35$:    DEC     $TMP1           ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
(1)   002550    001404                          BEQ     40$             ; SELECTED DEVICES
(1)   002552    000261                          SEC                     ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
(1)   002554    006137  001410                  ROL     SAVACTV         ;POINT TO THE NEXT DEVICE
```

```
(1)  002560  000771                          BR      35$             ;GO DO THIS PROCEDURE AGAIN
(1)  002562  013737  001410  001346  40$:    MOV     SAVACTV,$TMP2   ;# OF TIMES
(1)  002570  012700  001500                  MOV     #DZCR0,R0       ;SET A POINTER TO THE SPECIFIED INFORMATION
(1)  002574  012701  001512                  MOV     #DZCR1,R1       ;POINT R1 TO THE REST OF THE MAP TABLE
(1)  002600  012702  001204                  MOV     #$DDW0,R2       ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
(1)  002604  000241                          CLC                     ;INITIALIZE THE "C" BIT FOR A ROTATION
(1)  002606  006037  001346                  ROR     $TMP2           ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
(1)  002612  006237  001346          45$:    ASR     $TMP2           ;ISOLATE A SELECTION FLAG IN THE "C" BIT
(1)  002616  103404                          BCS     50$             ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
(1)  002620  012711  177777                  MOV     #-1,(R1)        ;TERMINATE THE LIST
(1)  002624  000137  003572                  JMP     100$            ;GO TO THE NEXT BLOCK
(1)  002630  012011                  50$:    MOV     (R0)+,(R1)      ;ADDRESS
(1)  002632  062721  000010                  ADD     #10,(R1)+       ;POINT TO THE NEXT DZV11 ADDRESS VALUE
(1)  002636  012011                          MOV     (R0)+,(R1)      ;VECTOR
(1)  002640  062721  000010                  ADD     #10,(R1)+       ;POINT TO THE NEXT VECTOR VALUE
(1)  002644  012021                          MOV     (R0)+,(R1)+     ;LINES
(1)  002646  012021                          MOV     (R0)+,(R1)+     ;PARAMETERS
(1)  002650  012021                          MOV     (R0)+,(R1)+     ;MAINTENANCE MODE
(1)  002652  000757                          BR      45$
(1)  002654  032777  000010  176422  55$:    BIT     #SW03,@SWR      ;ASK PARAMETERS ?
(1)  002662  001002                          BNE     60$             ;IF NO, GO DO AUTO SIZING
(1)  002664  000137  003572                  JMP     100$            ;GO SET UP FOR AUTO SIZING
(1)  002670  004737  002704          60$:    JSR     PC,65$          ;GO ASK PARAMETERS
(1)  002674  105337  001422                  DECB    INIFLG          ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
(1)  002700  000137  003630                  JMP     105$            ;GO TO THE NEXT BLOCK
(1)
(1)                                          ;GET THE ACTIVE LINES PARAMETER
(1)
(1)  002704                          65$:
(2)  002704  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002706  003201                          93$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002710  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002712  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002714  000017                          17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002716  001504                          LINE0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002720     360                          .BYTE   360             ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002721     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002722  105037  001423                  CLRB    HDRFLG          ;MAKE SURE THE CHANGES ARE PRINTED
(1)
(1)                                  ;THIS SEGMENT CHECKS TO MAKE SURE THE LINE PARAMETER JUST ENTERED
(1)                                  ;IS LEGITIMATE IN STAGGERED MODE OPERATION IF THAT MODE WAS SELECTED
(1)
(1)  002726  005737  001510                  TST     MANT0           ;IS STAGGERED THE MODE OF OPERATION?
(1)  002732  100021                          BPL     85$             ;IF NOT, SKIP THIS SEGMENT
(1)  002734  013703  001504                  MOV     LINE0,R3        ;GET A SCRATCH COPY OF THE ACTIVE LINES
(1)  002740  006003                  70$:    ROR     R3              ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
(1)  002742  103410                          BCS     80$             ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
(1)  002744  001414                          BEQ     85$             ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
(1)  002746  006203                          ASR     R3              ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
(1)  002750  103373                          BCC     70$             ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
(1)  002752  104402  001356          75$:    TYPE    ,$QUES          ;THIS IS AN INCORRECT PARAMETER
(1)  002756  104402  010176                  TYPE    ,MBADLN         ;LET THE USER KNOW ABOUT IT
(1)  002762  000750                          BR      65$             ;GO GET THE CORRECT PARAMETER
(1)  002764  001772                  80$:    BEQ     75$             ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
(1)  002766  006203                          ASR     R3              ;GET THE NEXT FLAG
(1)  002770  103370                          BCC     75$             ;IF IT ISN'T SET, THERE'S AN ERROR
```

```
(1)  002772  000241                        CLC                     ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
(1)  002774  000761                        BR      70$             ;GO TEST THE NEXT PAIR OF FLAGS
(1)
(1)                                         ;GET THE LINE PARAMETER REGISTER ARGUMENT
(1)
(1)  002776                           85$:
(2)  002776  104403                        INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003000  003254                        94$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  003002  104405                        PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  003004  000000                        0                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003006  000017                        17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003010  001506                        PAR0                    ;POINTER TO MAP LOCATION TO BE FILLED
(2)  003012     000                        .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  003013     001                        .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  003014  012702  001504                MOV     #LINE0,R2       ;POINT TO THE LINE SELECTION PARAMETER
(1)  003020  012703  001506                MOV     #PAR0,R3        ;POINT TO THE CHOSEN PARAMETERS
(1)  003024  011304                        MOV     (R3),R4         ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
(1)  003026  006304                        ASL     R4              ;ALIGN INDEX ON WORD BOUNDARY
(1)  003030  016437  017302  006406        MOV     DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
(1)  003036  000313                        SWAB    (R3)            ;PLACE IN HIGH BYTE
(1)  003040  052713  010070                BIS     #10070,(R3)     ;PLACE EXTRA PARAMETERS INTO LOC
(1)  003044  011262  000012           90$: MOV     (R2),12(R2)     ;LOAD THE LINES
(1)  003050  011363  000012                MOV     (R3),12(R3)     ;LOAD THE PARAMETERS
(1)  003054  062702  000012                ADD     #12,R2          ;POINT TO THE NEXT SET
(1)  003060  062703  000012                ADD     #12,R3          ;  .. OF BOTH PARAMETERS
(1)  003064  020327  001734                CMP     R3,#PAR17       ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  003070  001365                        BNE     90$             ;IF NOT, GO LOAD SOME MORE PARAMETERS
(1)  003072  000207                        RTS     PC              ;RETURN TO CALLING BLOCK
(1)  003074  030600  052123  041440  91$:  .ASCIZ  <200>/1ST CSR ADDRESS (160000:163770):  /
(1)  003140  030600  052123  053040  92$:  .ASCIZ  <200>/1ST VECTOR ADDRESS (300:770).  /
(1)  003201     200  044514  042516  93$:  .ASCIZ  <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:17):  /
(1)  003254  042200  043105  052501  94$:  .ASCIZ  <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17):  /
(1)  003324  021600  047440  020106  95$:  .ASCIZ  <200>/# OF DZV11'S <IN OCTAL> (1:20):  /
(1)  003367     200  040515  047111  96$:  .ASCII  <200>/MAINTENANCE MODE/
(1)  003410  020200  042533  052130        .ASCII  <200>/ [EXTERNAL  <H325>     (E)]/
(1)  003444  020200  044533  052116        .ASCII  <200>/ [INTERNAL  <DZVCSR03=1>(I)]/
(1)  003501     200  055440  052123        .ASCIZ  <200>/ [STAGGERED <H329>     (S)]: /
(1)  003540  042600  052116  051105  97$:  .ASCIZ  <200>/ENTER DELAY PARAMETER: /
(1)          003572                        .EVEN
(1)  003572                           100$:
(1)  003572  122737  000377  001422        CMPB    #377,INIFLG     ;ONLY DO AUTO SIZE ON 1ST START
(1)  003600  001013                        BNE     105$            ;
(1)  003602  032777  000200  175474        BIT     #BIT7,@SWR      ;BIT7=1??
(1)  003610  001007                        BNE     105$            ;BR IF NO AUTO SIZE
(1)  003612  005737  017356                TST     KXTFLAG         ; FALCON ??                     ;;GPA
(1)  003616  001402                        BEQ     1002$           ; SKIP NEXT IF NOT.             ;;GPA
(1)  003620  000137  002356                JMP     20$             ; YES, DON'T AUTO-SIZE.         ;;GPA
(1)  003624                           1002$:                                                      ;;GPA
(1)  003624  004737  011406                JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
(1)  003630  105737  001423           105$: TSTB    HDRFLG          ;HAS THE TABLE BEEN TYPED YET?
(1)  003634  001021                        BNE     120$            ;IF SO, DON'T TYPE IT AGAIN
(1)  003636  105337  001423                DECB    HDRFLG          ;INDICATE THAT THE TABLE WILL BE TYPED
(1)  003642  104402  010150                TYPE    ,XHEAD          ;TYPE MAP HEADER
(1)  003646  012700  001500                MOV     #DZV.MAP,R0     ;SET POINTER
(1)  003652  010037  001344           110$: MOV     R0,$TMP1        ;POINT TO THE MAP LOCATION
(1)  003656  012037  001346                MOV     (R0)+,$TMP2     ;SET DATA
```

```
(1)  003662  022737  177777  001346          CMP     #-1,$TMP2       ;END OF LIST?
(1)  003670  001403                          BEQ     120$            ;BR IF YES
(1)  003672  104411                  115$:   CONVRT                  ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
(1)  003674  010240                          XSTATQ                  ;CONVERT THE DATA AT THIS ADDRESS
(1)  003676  000765                          BR      110$            ;GO PRINT THE NEXT PARAMETER
(1)  003700  013737  001410  001406  120$:   MOV     SAVACTV,DZVACTV ;COPY BIT MAP OF SYSTEM DEVICES ACTIVE
(1)  003706  113737  001414  001416          MOVB    DZVNUM,SAVNO    ;COPY NO. OF SYSTEM DEVICES ACTIVE
(1)  003714  032777  000100  175362          BIT     #SW06,@SWR      ;DESELECT SPECIFIC DEVICES??
(1)  003722  001431                          BEQ     135$            ;BR IF NO.
(1)  003724                          121$:
(2)  003724  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003726  010066                          MNEW                    ;POINTER TO MESSAGE TO BE PRINTED
(2)  003730  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  003732  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003734  177777                          177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003736  001406                          DZVACTV                 ;POINTER TO MAP LOCATION TO BE FILLED
(2)  003740     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  003741     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  003742  023737  001406  001410          CMP     DZVACTV,SAVACTV ;IS THE VALUE VALID?
(1)  003750  101403                          BLOS    122$            ;BRANCH IF YES
(1)  003752  104402  007740                  TYPE    ,MERR3          ;IF NOT THEN TYPE ERROR
(1)  003756  000762                          BR      121$            ;GO REASK QUESTION
(1)  003760  105037  001416          122$:   CLRB    SAVNO           ;CLEAR NO. OF DEVICES BEING TESTED
(1)  003764  013737  001406  001344          MOV     DZVACTV,$TMP1   ;COPY BIT MAP OF ACTIVE DEVICES BEING TESTED
(1)  003772  006237  001344          126$:   ASR     $TMP1           ;SHIFT OUT AN ACTIVE BIT
(1)  003776  103002                          BCC     127$            ;IF NOT ACTIVE SKIP INCREMENT
(1)  004000  105237  001416                  INCB    SAVNO           ;IF ACTIVE RECORD IT
(1)  004004  001372                  127$:   BNE     126$            ;IF ALL ACTIVE BITS RECORDED DON'T BRANCH
(1)  004006  032777  000020  175270  135$:   BIT     #SW04,@SWR      ;CHECK TO SEE IF DELAY COUNT CHANGES
(1)  004014  001407                          BEQ     140$            ;IF NOT, GO CLEAR VECTOR AREA
(2)  004016  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  004020  003540                          97$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  004022  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  004024  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  004026  177777                          177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  004030  006406                          DLYCNT                  ;POINTER TO MAP LOCATION TO BE FILLED
(2)  004032     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  004033     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  004034  012700  000300          140$:   MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
(1)  004040  012701  000302                  MOV     #302,R1         ;VECTOR AREA. 300-776
(1)  004044  010120                  145$:   MOV     R1,(R0)+        ;START PUTTING 'PC+2 - HALT''
(1)  004046  005021                          CLR     (R1)+           ;IN VECTOR AREA.
(1)  004050  022021                          CMP     (R0)+,(R1)+     ;POP POINTERS
(1)  004052  005737  017356                  TST     KXTFLAG         ; IF FALCON...                  ;;GPA
(1)  004056  001403                          BEQ     1001$           ;                               ;;GPA
(1)  004060  020027  000400                  CMP     R0,#400         ;...STOP AT 400.                ;;GPA
(1)  004064  000402                          402                     ; SKIP NEXT                     ;;GPA
(1)  004066                          1001$:                          ;                               ;;GPA
(1)  004066  022700  001000                  CMP     #1000,R0        ;ALL DONE??
(1)  004072  001364                          BNE     145$            ;BR IF NO.
(1)
(1)                                  ;TEST START AND RESTART
(1)                                  ;----------------------
(1)
(1)  004074  012706  001120          .BEGIN: MOV     #STACK,SP       ;SET UP STACK
(1)  004100  106427  000200                  MTPS    #MASK           ;LOCK OUT INTERRUPTS
```

```
CVDZA-C MACY11 30G(1063)  10-AUG-81  11:08  PAGE 25-22
CVDZAC.P11    10-AUG-81 10:55          PROGRAM INITIALIZATION AND START UP.                                    SEQ 0041

   (1)  004104  005737  000042                    TST     @#42            ;IS PROGRAM UNDER MONITOR CONTROL
   (1)  004110  001015                            BNE     2$              ;BR IF YES
   (1)  004112  032777  000004  175164            BIT     #BIT2,@SWR      ;CHECK FOR LOCK ON TEST
   (1)  004120  001406                            BEQ     1$              ;BR IF NO LOCK DESIRED.
   (1)  004122  104402  007764                    TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
   (1)  004126  012737  000240  004416            MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
   (1)  004134  000403                            BR      2$              ;CONTINUE ALONG.
   (1)  004136  013737  004644  004416  1$:       MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
   (1)  004144  012737  010552  001252  2$:       MOV     #CYCLE,$LPADR   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
   (1)  004152  113737  001416  001415            MOVB    SAVNO,SAVNUM    ;COPY ACTIVE DEVICES BEING TESTED
   (1)  004160  104402  007655                    TYPE    ,MR             ;TYPE "RUNNING"
   (1)  004164  000177  175062                    JMP     @$LPADR         ;START TESTING
  3028                             ;;GPA   PRGEND  DZV11,<END PASS CVDZA-B  >,10.
```

```
  3029                                          ;EN` OF PASS
   (2)                                          ;TYPE NAME OF TEST
   (2)                                          ;UPDATE PASS COUNT
   (2)                                          ;CHECK FOR EXIT TO ACT-11
   (2)                                          ;RESTART TEST
   (3)                                  .SBTTL  END OF PASS ROUTINE
   (3)
   (4)                                  ;;*****************************************************************
   (3)                                  ;*INCREMENT THE PASS NUMBER ($PASS)
   (3)                                  ;*IF THERES A MONITOR GO TO IT
   (3)                                  ;*IF THERE ISN'T JUMP TO CYCLE
   (3)
   (3)  004170                          $EOP:
   (5)  004170  000004                          SCOPE
   (5)  004172  005037  001262                  CLR     $ERRPC          ;CLEAR LAST ERROR PC
   (5)  004176  105037  001247                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
   (5)  004202  104402  007631                  TYPE    ,MEPASS         ;TYPE END PASS
   (5)  004206  104402  010013                  TYPE    ,MCSRX          ;TYPE CSR
   (5)  004212  104412  004354                  CNVRT   ,XCSR           ;SHOW IT
   (5)  004216  104402  010021                  TYPE    ,MVECX          ;TYPE VECTOR
   (5)  004222  104412  004362                  CNVRT   ,XVEC           ;SHOW IT
   (5)  004226  005237  001126                  INC     $PASS           ;RAISE PASS COUNT
   (5)  004232  104402  010027                  TYPE    ,MPASSX         ;TYPE PASSES
   (5)  004236  104412  004370                  CNVRT   ,XPASS          ;SHOW IT
   (5)  004242  005337  001126                  DEC     $PASS           ;RESTORE PASS COUNT
   (5)  004246  104402  010040                  TYPE    ,MERRX          ;TYPE ERRORS
   (5)  004252  104412  004376                  CNVRT   ,XERR           ;SHOW IT
   (5)  004256  005237  001130                  INC     $DEVCT          ;INC $DEVCNT FOR APT
   (5)  004262  105337  001415                  DECB    SAVNUM          ;ARE ALL DEVICES TESTED?
   (5)  004266  001030                          BNE     $DOAGN          ;BR IF NO.
   (5)  004270  113737  001416  001415          MOVB    SAVNO,SAVNUM    ;RESTORE THE COUNT
   (3)  004276  005037  001354                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
   (3)  004302  005237  001126                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
   (3)  004306  042737  100000  001126          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
   (3)  004314  005327                          DEC     (PC)+           ;;LOOP?
   (3)  004316  000001          $EOPCT: .WORD   1
   (3)  004320  003013                          BGT     $DOAGN          ;;YES
   (3)  004322  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
   (3)  004324  000001          $ENDCT: .WORD   1
   (3)  004326  004316                          $EOPCT
   (3)  004330  013700  000042  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
   (3)  004334  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
   (3)  004336  000005                          RESET                   ;;CLEAR THE WORLD
   (3)  004340  004710          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
   (3)  004342  000240                          NOP                     ;;SAVE ROOM
   (3)  004344  000240                          NOP                     ;;FOR
   (3)  004346  000240                          NOP                     ;;ACT11
   (3)  004350                          $DOAGN:
   (3)  004350  000137                          JMP     @(PC)+          ;;RETURN
   (3)  004352  010552          $RTNAD: .WORD   CYCLE
   (2)
   (2)  004354  000001          XCSR:   1
   (2)  004356     006     002                  .BYTE   6,2
   (2)  004360  002010                          DZVCSR
   (2)  004362  000001          XVEC:   1
   (2)  004364     003     002                  .BYTE   3,2
```

SEQ 0043

```
                                        END OF PASS ROUTINE

(2)  004366  002040                             DZVRIV
(2)  004370  000001                     XPASS:  1
(2)  004372     006     002                     .BYTE   6,2
(2)  004374  001126                             $PASS
(2)  004376  000001                     XERR:   1
(2)  004400     006     002                     .BYTE   6,2
(2)  004402  001256                             $ERTTL
(2)
(2)                                             ;SCOPE LOOP AND ITERATION HANDLER
(2)                                             ;-------------------------------
(2)
(3)                                     .SBTTL  SCOPE HANDLER ROUTINE
(3)
(4)                                     ;;**************************************************************************
(3)                                     ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(3)                                     ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(3)                                     ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(3)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(3)                                     ;*SW14=1        LOOP ON TEST
(3)                                     ;*SW11=1        INHIBIT ITERATIONS
(3)                                     ;*CALL
(3)                                     ;*      SCOPE           ;;SCOPE=IOT
(3)
(3)  004404                             $SCOPE:
(5)  004404  005037  001262             .SCOPE: CLR     $ERRPC          ;CLEAR LAST ERROR PC.
(5)  004410  022716  012114                     CMP     #TST1+2,(SP)    ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
(5)  004414  001413                             BEQ     $XTSTR          ;IF SO, DON'T LOOP ON IT
(5)  004416  000?06                     TTST:   BR      1$              ;GOTO 1$    (IF LOCK SW02=1; THIS LOC =240)
(5)  004420  105777  174664                     TSTB    @$TKS           ;KEYBOARD DONE?
(5)  004424  100067                             BPL     $OVER           ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
(5)  004426  017766  174660  177776             MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
(3)  004434  032777  040000  174642  1$:        BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(3)  004442  001060                             BNE     $OVER           ;;YES IF SW14=1
(3)                                     ;#####START OF CODE FOR THE XOR TESTER#####
(3)  004444  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3)  004446  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3)  004452  012737  004472  000004             MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(3)  004460  005737  177060                     TST     @#177060        ;;TIME OUT ON XOR?
(3)  004464  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004470  000436                             BR      $SVLAD          ;;GO TO THE NEXT TEST
(3)  004472  022626                     5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(3)  004474  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004500  000441                             BR      $OVER           ;;LOOP ON THE PRESENT TEST
(3)  004502                             6$:;#####END OF CODE FOR THE XOR TESTER#####
(3)  004502  105737  001247             2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(3)  004506  001404                             BEQ     3$              ;;BR IF NO
(3)  004510  105037  001247             4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(3)  004514  005037  001354                     CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3)  004520  032777  004000  174556  3$:        BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(3)  004526  001011                             BNE     1$              ;;BR IF YES
(3)  004530  005737  001126                     TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(3)  004534  001406                             BEQ     1$              ;;      INHIBIT ITERATIONS
(3)  004536  005237  001250                     INC     $ICNT           ;;INCREMENT ITERATION COUNT
(3)  004542  023737  001354  001250             CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(3)  004550  002015                             BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
```

```
(3)  004552  012737  000001  001250  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(3)  004560  013737  004646  001354          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(3)  004566  105237  001246          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(3)  004572  113737  001246  001124          MOVB    $TSTNM,$TFSTN   ;;SET TEST NUMBER IN APT MAILBOX
(3)  004600  011637  001252                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(3)  004604  013777  001246  174474  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(3)  004612  013716  001252                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(5)  004616  004737  007150                  JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
(5)  004622  105037  001424                  CLRB    MNTFLG          ;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
(5)  004626  005737  001372                  TST     MODE            ;HAS THE MODE BEEN CHANGED?
(5)  004632  001003                          BNE     4$              ;IF NOT INTERNAL , GO DO A TEST
(5)  004634  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
(5)  004642  000002          4$:     RTI                             ;GO DO THE TEST
(5)  004644  000406          BRW:    406
(3)  004646  000005          $MXCNT: 5                               ;;MAX. NUMBER OF ITERATIONS
(1)
(1)                                  ;CHECK FOR FREEZE ON CURRENT DATA
(1)                                  ;-------------------------------
(1)
(1)  004650  032777  001000  174426  .SCOP1: BIT     #SW09,@SWR      ;IS SW09=1(SET)?
(1)  004656  001405                          BEQ     1$              ;BR IF NOT SET.
(1)  004660  005737  001364                  TST     LOCK            ;IS THERE A TIGHT LOOP SPECIFIED?
(1)  004664  001402                          BEQ     1$              ;IF NO, RETURN
(1)  004666  013716  001364                  MOV     LOCK,(SP)       ;IF YES, GOTO THE ADDRESS IN LOCK.
(1)  004672  000002          1$:     RTI                             ;GO BACK.
(1)
(1)  004674  032777  010000  174402  .TYPE:  BIT     #SW12,@SWR      ;INHIBIT ALL PRINTOUT??
(1)  004702  001403                          BEQ     $TYPE           ;IF NOT, GO TYPE
(1)  004704  062716  000002                  ADD     #2,(SP)         ;SKIP OVER MESSAGE POINTER
(1)  004710  000002                          RTI                     ;RETURN TO WHERE PROCEDURE WAS INVOKED
(2)                                  .SBTTL  TYPE ROUTINE
(2)
(3)                                  ;;**************************************************************************
(2)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(2)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(2)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(2)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(2)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(2)                                  ;*
(2)                                  ;*CALL:
(2)                                  ;*1) USING A TRAP INSTRUCTION
(2)                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(2)                                  ;*OR
(2)                                  ;*      TYPE
(2)                                  ;*      MESADR
(2)                                  ;*
(2)
(2)  004712  105737  001323          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(2)  004716  100002                          BPL     1$              ;;BR IF YES
(2)  004720  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
(2)  004722  000430                          BR      3$              ;;LEAVE
(2)  004724  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
(2)  004726  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(2)  004732  122737  000001  001140          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  004740  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(2)  004742  132737  000100  001141          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
(2)  004750  001405                    BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(2)  004752  010037  004762            MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(2)  004756  004737  005254            JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(2)  004762  000000            61$:    .WORD   0               ;;MESSAGE ADDRESS
(2)  004764  132737  000040  001141  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(2)  004772  001003                    BNE     60$             ;;YES,SKIP TYPE OUT
(2)  004774  112046            2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2)  004776  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(2)  005000  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(2)  005002  012600            60$:    MOV     (SP)+,R0        ;;RESTORE R0
(2)  005004  062716  000002    3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(2)  005010  000002                    RTI                     ;;RETURN
(2)  005012  122716  000011    4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(2)  005016  001430                    BEQ     8$
(2)  005020  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(2)  005024  001006                    BNE     5$
(2)  005026  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
(2)  005030  104402                    TYPE                    ;;TYPE A CR AND LF
(2)  005032  001357                    $CRLF
(2)  005034  105037  005242            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(2)  005040  000755                    BR      2$              ;;GET NEXT CHARACTER
(2)  005042  004737  005124    5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(2)  005046  123726  001322    6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(2)  005052  001350                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
(2)  005054  013746  001320            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(2)                                                            ;;AND THE NULL CHAR.
(2)  005060  105366  000001    7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(2)  005064  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2)  005066  004737  005124            JSR     PC,$TYPEC       ;;GO TYPE A NULL
(2)  005072  105337  005242            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(2)  005076  000770                    BR      7$              ;;LOOP
(2)
(2)                                    ;HORIZONTAL TAB PROCESSOR
(2)
(2)  005100  112716  000040    8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(2)  005104  004737  005124    9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(2)  005110  132737  000007  005242    BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(2)  005116  001372                    BNE     9$              ;;TAB STOP
(2)  005120  005726                    TST     (SP)+           ;;POP SPACE OFF STACK
(2)  005122  000724                    BR      2$              ;;GET NEXT CHARACTER
(2)  005124                    $TYPEC:
(2)  005124  105777  174160            TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
(2)  005130  100022                    BPL     10$             ;;BR IF NOT                     ;MJD001
(2)  005132  017746  174154            MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
(2)  005136  042716  177600            BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
(2)  005142  122716  000023            CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
(2)  005146  001012                    BNE     102$            ;;BR IF NOT                     ;MJD001
(2)  005150                    101$:                           ;                               ;MJD001
(2)  005150  105777  174134            TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
(2)  005154  100375                    BPL     101$            ;                               ;MJD001
(2)  005156  117716  174130            MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
(2)  005162  042716  177600            BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
(2)  005166  122716  000021            CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
(2)  005172  001366                    BNE     101$            ;;BR IF NOT                     ;MJD001
(2)  005174                    102$:                           ;                               ;MJD001
(2)  005174  005726                    TST     (SP)+           ;;FIX STACK                     ;MJD001
```

```
(2) 005176                          10$:                                                    ;MJD001
(2) 005176 105777 174112                    TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(2) 005202 100375                           BPL     10$                                     ;MJD001
(2) 005204 116677 000002 174104             MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2) 005212 122766 000015 000002             CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(2) 005220 001003                           BNE     1$              ;;BRANCH IF NO
(2) 005222 105037 005242                    CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(2) 005226 000406                           BR      $TYPEX          ;;EXIT
(2) 005230 122766 000012 000002 1$:         CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(2) 005236 001402                           BEQ     $TYPEX          ;;BRANCH IF YES
(2) 005240 105227                           INCB    (PC)+           ;;COUNT THE CHARACTER
(2) 005242 000000                   $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(2) 005244 000207                   $TYPEX: RTS     PC
(2)
(2)                                 .SBTTL  APT COMMUNICATIONS ROUTINE
(2)
(3)                                 ;;**************************************************************************
(2) 005246 112737 000001 005512 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(2) 005254 112737 000001 005510 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(2) 005262 000403                       BR      $ATYC
(2) 005264 112737 000001 005512 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(2) 005272                      $ATYC:
(4) 005272 010046                       MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(4) 005274 010146                       MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(2) 005276 105737 005510                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(2) 005302 001450                       BEQ     5$              ;;IF NOT:  BR
(2) 005304 122737 000001 001140         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(2) 005312 001031                       BNE     3$              ;;IF NOT:  BR
(2) 005314 132737 000100 001141         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(2) 005322 001425                       BEQ     3$              ;;IF NOT:  BR
(2) 005324 017600 000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(2) 005330 062766 000002 000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(2) 005336 005737 001120            1$: TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(2) 005342 001375                       BNE     1$              ;;IF NOT:  WAIT
(2) 005344 010037 001134                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(2) 005350 105720                   2$: TSTB    (R0)+           ;;FIND END OF MESSAGE
(2) 005352 001376                       BNE     2$
(2) 005354 163700 001134                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(2) 005360 006200                       ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(2) 005362 010037 001136                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(2) 005366 012737 000004 001120         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(2) 005374 000413                       BR      5$
(2) 005376 017637 000004 005422 3$: MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(2) 005404 062766 000002 000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(4) 005412 013746 177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(2) 005416 004737 004712                JSR     PC,$TYPE        ;;CALL TYPE MACRO
(2) 005422 000000                   4$: .WORD   0
(2) 005424                          5$:
(2) 005424 105737 005512            10$:TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(2) 005430 001416                       BEQ     12$             ;;IF NOT:  BR
(2) 005432 005737 001140                TST     $ENV            ;;RUNNING UNDER APT?
(2) 005436 001413                       BEQ     12$             ;;IF NOT:  BR
(2) 005440 005737 001120            11$:TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(2) 005444 001375                       BNE     11$             ;;IF NOT:  WAIT
(2) 005446 017637 000004 001122         MOV     @4(SP),$FATAL   ;;GET ERROR #
(2) 005454 062766 000002 000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
```

```
(2)  005462  005237  001120                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(2)  005466  105037  005512       12$:      CLRB    $FFLG           ;;CLEAR FATAL FLAG
(2)  005472  105037  005511                 CLRB    $LFLG           ;;CLEAR LOG FLAG
(2)  005476  105037  005510                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(4)  005502  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(4)  005504  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(2)  005506  000207                         RTS     PC              ;;RETURN
(2)  005510     000         $MFLG:  .BYTE   0               ;;MESSG. FLAG
(2)  005511     000         $LFLG:  .BYTE   0               ;;LOG FLAG
(2)  005512     000         $FFLG:  .BYTE   0               ;;FATAL FLAG
(2)          005514                 .EVEN
(2)          000200         APTSIZE=200
(2)          000001         APTENV=001
(2)          000100         APTSPOOL=100
(2)          000040         APTCSUP=040
(1)
(1)                                 ;STRING INPUT ROUTINE
(1)                                 ;---------------------
(1)
(1)  005514  010346         .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
(1)  005516  010446                 MOV     R4,-(SP)        ;SAVE R4 ON STACK
(1)  005520  017637  000004  005536 MOV     @4(SP),.MSG     ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1)  005526  062766  000002  000004 ADD     #2,4(SP)        ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1)  005534  104402         .INST1: TYPE                    ;PRINT THE MESSAGE
(1)  005536  000000         .MSG:   0                       ;MESSAGE IS POINTED TO FROM HERE
(1)  005540  012704  010446         MOV     #INBUF,R4       ;POINT R4 TO THE INPUT BUFFER
(1)  005544  012703  000007         MOV     #7,R3           ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1)  005550  105777  173534  1$:    TSTB    @$TKS           ;HAS A CHARACTER BEEN RECEIVED?
(1)  005554  100375                 BPL     1$              ;IF NO, KEEP WAITING FOR IT
(1)  005556  117714  173530         MOVB    @$TKB,(R4)      ;IF YES, SAVE IT IN THE INPUT BUFFER
(1)  005562  142714  000200         BICB    #200,(R4)       ;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1)  005566  122427  000015         CMPB    (R4)+,#15       ;IS THIS CHARACTER A LINE FEED?
(1)  005572  001417                 BEQ     INSTR2          ;IF SO, TERMINATE THE INPUT SEQUENCE
(1)  005574  105777  173514  2$:    TSTB    @$TPS           ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1)  005600  100375                 BPL     2$              ;IF WE CAN'T, WAIT UNTIL WE CAN
(1)  005602  017777  173504  173506 MOV     @$TKB,@$TPB     ;ECHO THE CHARACTER BACK
(1)  005610  005303                 DEC     R3              ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
(1)  005612  001356                 BNE     1$              ;IF WE DON'T HAVE 7, GO GET SOME MORE
(1)  005614  012604                 MOV     (SP)+,R4        ;IF WE HAVE 7, RESTORE R4
(1)  005616  012603                 MOV     (SP)+,R3        ;RESTORE R3
(1)  005620  010346         .INSTE: MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
(1)  005622  010446                 MOV     P.,-(SP)        ;SAVE R4 ON THE STACK
(1)  005624  104402  001356         TYPE    ,$QUES          ;PRINT A QUESTION MARK... WHAT'S GOING ON?
(1)  005630  000741                 BR      .INST1          ;GO PRINT THE MESSAGE AGAIN
(1)  005632  012604         INSTR2: MOV     (SP)+,R4        ;RESTORE R4
(1)  005634  012603                 MOV     (SP)+,R3        ;RESTORE R3
(1)  005636  000002                 RTI                     ;RETURN TO THE MAIN PROCEDURE
(1)
(1)                                 ;CONVERT ASCII STRING TO OCTAL
(1)                                 ;-----------------------------
(1)
(1)  005640  010546         .PARAM: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
(1)  005642  010446                 MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
(1)  005644  016605  000004         MOV     4(SP),R5        ;GET THE SETUP INFORMATION POINTER
(1)  005650  012537  006030         MOV     (R5)+,LOLIM     ;SET THE LOW LIMIT FOR THE INPUT
(1)  005654  012537  006032         MOV     (R5)+,HILIM     ;SET THE HIGH LIMIT FOR THE INPUT
```

```
(1)  005660  012537  006034            MOV     (R5)+,DEVADR    ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
(1)  005664  112537  006036            MOVB    (R5)+,LOBITS    ;GET THE MASK OF THE INCORRECT BITS
(1)  005670  112537  006037            MOVB    (R5)+,ADRCNT    ;GET THE COUNT OF ITEMS TO BE STORED
(1)  005674  010566  000004            MOV     R5,4(SP)        ;POINT TO WHERE MAIN LINE PROGRAM WILL R. ME
(1)  005700  005005            PARAM1: CLR     R5              ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
(1)  005702  012704  010446            MOV     #INBUF,R4       ;POINT TO THE INPUT BUFFER
(1)  005706  122714  000015            CMPB    #15,(R4)        ;IS THIS CHARACTER A CARRIAGE RETURN?
(1)  005712  001420                    BEQ     PARERR          ;IF SO, PRINT THE MESSAGE AGAIN
(1)  005714  121427  000060    1$:     CMPB    (R4),#60        ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
(1)  005720  002415                    BLT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)  005722  121427  000067            CMPB    (R4),#67        ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
(1)  005726  003012                    BGT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)  005730  142714  000060            BICB    #60,(R4)        ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
(1)  005734  152405                    BISB    (R4)+,R5        ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
(1)  005736  122714  000015            CMPB    #15,(R4)        ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
(1)  005742  001406                    BEQ     LIMITS          ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
(1)  005744  006305                    ASL     R5              ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
(1)  005746  006305                    ASL     R5              ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
(1)  005750  006305                    ASL     R5              ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
(1)                                                            ;NEXT THREE BITS
(1)  005752  000760                    BR      1$              ;GO GET THE NEXT CHARACTER
(1)  005754  104404            PARERR: INSTER                  ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
(1)  005756  000750                    BR      PARAM1          ;TRY GETTING THE PARAMETERS AGAIN
(1)
(1)                                    ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(1)                                    ;--------------------------------------
(1)
(1)  005760  020537  006032    LIMITS: CMP     R5,HILIM        ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
(1)  005764  101373                    BHI     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005766  020537  006030            CMP     R5,LOLIM        ;IS THE RESULT LOWER THAN ALLOWED?
(1)  005772  103770                    BLO     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005774  133705  006036            BITB    LOBITS,R5       ;ARE ANY INCORRECT BITS SET IN THE RESULT?
(1)  006000  001365                    BNE     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)
(1)                                    ;STORE NUMBER AT SPECIFIED ADDRESS
(1)
(1)  006002  013704  006034            MOV     DEVADR,R4       ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1)  006006  010524            1$:     MOV     R5,(R4)+        ;STORE THE RESULT
(1)  006010  062705  000002            ADD     #2,R5           ;CALCULATE THE NEXT DATUM
(1)  006014  105337  006037            DECB    ADRCNT          ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1)  006020  001372                    BNE     1$              ;IF NOT, GO STORE THE NEXT DATUM
(1)  006022  012604                    MOV     (SP)+,R4        ;RESTORE R4
(1)  006024  012605                    MOV     (SP)+,R5        ;RESTORE R5
(1)  006026  000002                    RTI                     ;RETURN TO THE MAIN PROGRAM
(1)
(1)  006030  000000            LOLIM:  0                       ;LOWEST ACCEPTABLE VALUE
(1)  006032  000000            HILIM:  0                       ;HIGHEST ACCEPTABLE
(1)  006034  000000            DEVADR: 0                       ;LOCATION WHERE RESULT WILL BE STORED
(1)  006036     000            LOBITS: .BYTE   0               ;INCORRECT BITS MASK
(1)  006037     000            ADRCNT: .BYTE   0               ;COUNT OF ITEMS TO BE STORED
(1)
(1)                                    ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1)                                    ;--------------------------------------
(1)
(1)  006040  016637  000004  001404  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
(1)
```

```
(1)                                             ;SAVE R0-R5
(1)
(1)  006046  010537  001340          SV05:      MOV     R5,$REG5        ;SAVE R5
(1)  006052  010437  001336                     MOV     R4,$REG4        ;SAVE R4
(1)  006056  010337  001334                     MOV     R3,$REG3        ;SAVE R3
(1)  006062  010237  001332                     MOV     R2,$REG2        ;SAVE R2
(1)  006066  010137  001330                     MOV     R1,$REG1        ;SAVE R1
(1)  006072  010037  001326                     MOV     R0,$REG0        ;SAVE R0
(1)  006076  000002                             RTI                     ;LEAVE.
(1)
(1)                                             ;RESTORE R0-R5
(1)
(1)  006100  013700  001326          .RES05:    MOV     $REG0,R0        ;RESTORE R0
(1)  006104  013701  001330                     MOV     $REG1,R1        ;RESTORE R1
(1)  006110  013702  001332                     MOV     $REG2,R2        ;RESTORE R2
(1)  006114  013703  001334                     MOV     $REG3,R3        ;RESTORE R3
(1)  006120  013704  001336                     MOV     $REG4,R4        ;RESTORE R4
(1)  006124  013705  001340                     MOV     $REG5,R5        ;RESTORE R5
(1)  006130  000002                             RTI                     ;LEAVE
(1)
(1)                                             ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(1)                                             ;-----------------------------------------------------
(1)
(1)  006132  104402  001357          .CONVR:    TYPE    ,$CRLF          ;PRINT A CARRIAGE RETURN
(1)  006136  010046                  .CNVRT:    MOV     R0,-(SP)        ;SAVE R0
(1)  006140  010146                             MOV     R1,-(SP)        ;SAVE R1
(1)  006142  010346                             MOV     R3,-(SP)        ;SAVE R3
(1)  006144  010446                             MOV     R4,-(SP)        ;SAVE R4
(1)  006146  010546                             MOV     R5,-(SP)        ;SAVE R5
(1)  006150  017601  000012                     MOV     @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
(1)  006154  062766  000002  000012             ADD     #2,12(SP)       ;POINT TO WHERE MAIN PROGRAM WILL RESUME
(1)  006162  012137  006306                     MOV     (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
(1)  006166  112105                  1$:        MOVB    (R1)+,R5        ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
(1)  006170  112100                             MOVB    (R1)+,R0        ;GET THE NUMBER OF SPACES TO PRINT
(1)  006172  013104                             MOV     @(R1)+,R4       ;COPY THE WORD TO BE CONVERTED
(1)  006174  110537  006310                     MOVB    R5,CHRCNT       ;COPY THE CHARACTER COUNT
(1)  006200  010403                  3$:        MOV     R4,R3           ;COPY THE ARGUMENT WORD AGAIN
(1)  006202  042703  177770                     BIC     #^C<7>,R3       ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
(1)  006206  062703  000060                     ADD     #060,R3         ;MAKE AN ASCII CHARACTER OUT OF THEM
(1)  006212  110346                             MOVB    R3,-(SP)        ;SAVE THAT CHARACTER
(1)  006214  006004                             ROR     R4              ;MOVE THE NEXT THREE BITS INTO PLACE
(1)  006216  006204                             ASR     R4              ;MOVE THEM AGAIN
(1)  006220  006204                             ASR     R4              ;AND FINALLY A THIRD TIME
(1)  006222  005305                             DEC     R5              ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
(1)                                             ;BUILT?
(1)  006224  001365                             BNE     3$              ;IF NO, GO BUILD THE NEXT ONE.
(1)  006226  012703  010510                     MOV     #MDATA,R3       ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1)  006232  112623                  4$:        MOVB    (SP)+,(R3)+     ;STORE THE CHARACTER, STARTING WITH THE MOST
(1)  006234  105337  006310                     DECB    CHRCNT          ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1)  006240  001374                             BNE     4$              ;IF NO, GO TRANSFER ANOTHER
(1)  006242  105700                             TSTB    R0              ;ARE ANY SPACES TO BE PRINTED?
(1)  006244  001404                             BEQ     6$              ;IF NO, DON'T SET UP ANY
(1)  006246  112723  000040          5$:        MOVB    #040,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
(1)  006252  105300                             DECB    R0              ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1)  006254  001374                             BNE     5$              ;IF YES, GO ADD ANOTHER SPACE
(1)  006256  105013                  6$:        CLRB    (R3)            ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
```

```
(1)  006260  104402  010510                TYPE    ,MDATA          ;PRINT THE STRING WE JUST BUILT
(1)  006264  005337  006306                DEC     WRDCNT          ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1)  006270  001336                        BNE     1$              ;IF YES, GO CONVERT THEM
(1)  006272  012605                        MOV     (SP)+,R5        ;RESTORE R5
(1)  006274  012604                        MOV     (SP)+,R4        ;RESTORE R4
(1)  006276  012603                        MOV     (SP)+,R3        ;RESTORE R3
(1)  006300  012601                        MOV     (SP)+,R1        ;RESTORE R1
(1)  006302  012600                        MOV     (SP)+,R0        ;RESTORE R0
(1)  006304  000002                        RTI                     ;RETURN TO THE MAIN PROGRAM
(1)  006306  000000              WRDCNT:   0
(1)  006310     000              CHRCNT:   .BYTE                   ;NUMBER OF CHARACTERS TO PRINT
(1)  006311     000              SPACNT:   .BYTE   0               ;NUMBER OF SPACES TO PRINT
(1)
(1)  006312  000000              BINWRD:   0
(1)
(1)
(1)                                        ;TRAP DISPATCH SERVICE
(1)                                        ;ARGUMENT OF TRAP IS EXTRACTED
(1)                                        ;AND USED AS OFFSET TO OBTAIN POINTER
(1)                                        ;TO SELECTED SUBROUTINE
(1)
(1)  006314  010046              .TRPSR:   MOV     R0,-(SP)        ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
(1)  006316  016600  000002                MOV     2(SP),R0        ;GET TRAP ADDRESS
(1)  006322  005740                        TST     -(R0)           ;GET TRAP
(1)  006324  111000                        MOVB    (R0),R0         ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
(1)  006326  006300                        ASL     R0              ;POSITION OFFSET FOR TABLE INDEXING
(1)  006330  016000  001742                MOV     .TRPTAB(R0),R0  ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1)  006334  000200                        RTS     R0              ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                                        ;DEVICE CLEAR ROUTINE
(1)                                        ;ISSUE A DEVICE CLEAR
(1)                                        ;--------------------
(1)  006336                      .DEVICE.CLR:
(1)  006336  052777  000020  173444        BIS     #DCLR,@DZVCSR   ;SET DCLR
(1)  006344  032777  000020  173436  1$:    BIT     #DCLR,@DZVCSR   ;DID IT CLEAR?
(1)  006352  001374                        BNE     1$              ;BR IF NO
(1)  006354  000002                        RTI                     ;EXIT ROUTINE
(1)
(1)                                        ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                                        ;-----------------------------------------------------------
(1)  006356  104413              .DCLASM:DEVICE.CLR                ;ISSUE A DEVICE CLEAR
(1)  006360  153777  001424  173422        BISB    MNTFLG,@DZVCSR  ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1)  006366  000002                        RTI                     ;RETURN TO CALLING ROUTINE
(1)
(1)  006370                      .DELAY:
(1)  006370  010046                        MOV     R0,-(SP)        ;SAVE R0
(1)  006372  013700  006406                MOV     DLYCNT,R0       ;SET COUNT
(1)  006376  005300              1$:       DEC     R0              ;DELAY
(1)  006400  001376                        BNE     1$              ;
(1)  006402  012600                        MOV     (SP)+,R0        ;RESTORE R0
(1)  006404  000002                        RTI                     ;LEAVE ROUTINE
(1)  006406  000001              DLYCNT:   .WORD     1             ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                                        ;ADVANCE TO NEXT TEST HANDLER
(1)                                        ;---------------------------
(1)
```

```
    (1)  006410  013716  001362          .ADVANCE:MOV    NEXT,(SP)       ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
    (1)  006414  005037  001364                   CLR     LOCK            ;RESET TIGHT LOOP ADDRESS
    (1)  006420  000002                           RTI                     ;CHECK TO SEE IF OLD TEST GETS REPEATED
    (1)
    (1)                                           ;ROUTINE TO SHIFT LINE POINTER
    (1)                                           ;AND SWITCH TESTS IF NECESSARY
    (1)                                           ;----------------------------
    (1)  006422  106302                  .SHIFT:  ASLB    R2              ;POINT TO THE NEXT LINE
    (1)  006424  032702  000020                   BIT     #BIT4,R2        ;HAVE WE PASSED ALL LINE POINTERS?
    (1)  006430  001402                           BEQ     1$              ;IF NOT, RETURN TO THE TEST
    (1)  006432  022626                           POP2SP                  ;REMOVE THE TRAP CALL FROM THE STACK
    (1)  006434  104400                           ADVANCE                 ;GO TO THE NEXT TEST
    (1)  006436  000002                  1$:      RTI                     ;RETURN TO THE PRESENT TEST
    (1)
```

```
(1)                                             ;LINE PARAMETER REGISTER SETUP ROUTINE
(1)
(1)  006440  010146                 .LPRSET:MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
(1)  006442  010246                         MOV     R2,-(SP)        ;SAVE CONTENTS OF R2
(1)  006444  013701  001370                 MOV     PAR,R1          ;MOVE DEFAULT PARAM. INTO R1
(1)  006450  012702  000001                 MOV     #1,R2   ;INIT. FOR LINE 1
(1)  006454  010177  173340         1$:     MOV     R1,@DZVLPR      ;LOAD PARAM. REGISTER
(1)  006460  005201                         INC     R1              ;SET R1 FOR NEXT LINE
(1)  006462  106302                         ASLB    R2              ;SET R2 FOR NEXT LINE
(1)  006464  032702  000020                 BIT     #BIT4,R2        ;ALL LINES DONE?
(1)  006470  001771                         BEQ     1$              ;IF NO LOAD NEXT LINE
(1)  006472  012602                         MOV     (SP)+,R2        ;RELOAD R2
(1)  006474  012601                         MOV     (SP)+,R1        ;RELOAD R1
(1)  006476  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ROUTINE TO ZERO DATA BUFFER
(1)
(1)  006500  010046                 .BUFSET:MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
(1)  006502  012700  001426                 MOV     #TDO,R0         ;SET R0 TO TOP OF BUFFER
(1)  006506  005020                 1$:     CLR     (R0)+           ;CLEAR BUFFER LOCATION
(1)  006510  022700  001446                 CMP     #STOP,R0        ;IS BUFFER ALL CLEARED
(1)  006514  001374                         BNE     1$              ;IF NOT CLEAR NEXT LOCATION
(1)  006516  012600                         MOV     (SP)+,R0        ;RELOAD R0
(1)  006520  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ERROR HANDLER
(1)                                             ;-------------
(1)
(1)  006522  004737  007150         $ERROR: JSR     PC,SERV.G       ;FIND OUT IF <^G> WAS HIT
(1)  006526  032777  010000  172550         BIT     #SW12,@SWR      ;BELL ON ERROR?
(1)  006534  001406                         BEQ     XBX             ;BR IF NO BELL
(1)  006536  105777  172552                 TSTB    @STPS           ;TTY READY.
(1)  006542  100003                         BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
(1)  006544  112777  000207  172544         MOVB    #207,@STPB      ;PUSH A BELL AT THE TTY.
(1)  006552  032777  020000  172524 XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
(1)  006560  001113                         BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
(1)  006562  021637  001262                 CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
(1)  006566  001404                         BEQ     1$              ;BR IF YES
(1)  006570  011637  001262                 MOV     (SP),$ERRPC     ;RECORD BEING HERE
(1)  006574  105037  001247                 CLRB    $ERFLG          ;PREPARE HEADER
(1)  006600  104407                 1$:     SAV05                   ;SAVE ALL PROC REGISTERS
(1)  006602  011605                         MOV     (SP),R5         ;GET THE PC OF ERROR
(1)  006604  162705  000002                 SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
(1)  006610  011504                         MOV     (R5),R4         ;GET ERROR INSTRUCTION
(1)  006612  110437  001260                 MOVB    R4,$ITEMB       ;COPY TEST NUMBER FOR APT HANDLING
(1)  006616  006304                         ASL     R4              ;MULT BY TWO
(1)  006620  061504                         ADD     (R5),R4         ;DOUBLE IT
(1)  006622  006304                         ASL     R4              ;MULT AGAIN
(1)  006624  042704  177001                 BIC     #177001,R4      ;CLEAR JUNK
(1)  006630  062704  016122                 ADD     #.ERRTAB,R4     ;GET POINTER
(1)  006634  012437  006760                 MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
(1)  006640  012437  006772                 MOV     (R4)+,DATAHD    ;GET DATA HEADRER
(1)  006644  011437  007004                 MOV     (R4),DATABP     ;GET DATA TABLE
(1)  006650  105737  001247                 TSTB    $ERFLG          ;TYPE HEADER
(1)  006654  001403                         BEQ     TYPMSG          ;BR IF YES
(1)  006656  005737  007004                 TST     DATABP          ;DOES DATA TABLE EXIST?
```

```
(1)  006662  001044                        BNE     TYPDAT          ;BR IF YES.
(1)  006664  104402  001357          TYPMSG: TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN
(1)  006670  104402  001357                  TYPE    ,$CRLF          ;AND TYPE ANOTHER
(1)  006674  005737  001364                  TST     LOCK
(1)  006700  001402                          BEQ     1$
(1)  006702  104402  010063                  TYPE    ,MASTEK
(1)  006706  104402  010051          1$:     TYPE    ,MTSTN
(1)  006712  104412  007142                  CNVRT   ,XTSTN          ;SHOW IT
(1)  006716  104402  010143                  TYPE    ,MERRPC         ;TYPE PC.
(1)  006722  104412  007134                  CNVRT   ,ERTABO         ;SHOW IT
(1)  006726  104402  010013                  TYPE    ,MCSRX
(1)  006732  104412  004354                  CNVRT   ,XCSR
(1)  006736  104402  001357                  TYPE    ,$CRLF          ;GIVE A CR/LF
(1)  006742  112737  177777  001247          MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
(1)  006750  005737  006760                  TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
(1)  006754  001402                          BEQ     WTBS.FM         ;BR IF NO.
(1)  006756  104402                          TYPE                    ;TYPE
(1)  006760  000000          ERRMSG: 0                               ;    ERROR MESSAGE
(1)  006762                  WTBS.FM:                                ;
(1)  006762  005737  006772                  TST     DATAHD          ;DATA HEADER?
(1)  006766  001402                          BEQ     TYPDAT          ;BR IF NO
(1)  006770  104402                          TYPE                    ;TYPE
(1)  006772  000000          DATAHD: 0                               ;    DATA HEADER
(1)  006774  005737  007004          TYPDAT: TST     DATABP          ;DATA TABLE?
(1)  007000  001402                          BEQ     RESREG          ;BR IF NO.
(1)  007002  104411                          CONVRT                  ;SHOW
(1)  007004  000000          DATABP: 0                               ;    DATA TABLE
(1)  007006  104410          RESREG: RES05                           ;RESTORE PROC REGISTERS
(1)  007010  122737  000001  001140  HALTS:  CMPB    #APTENV,$ENV    ;IS APT RUNNING?
(1)  007016  001007                          BNE     15$             ;SKIP APT CALL IF NOT
(1)  007020  113737  001260  007032          MOVB    $ITEMB,5$       ;COPY ERROR NUMBER
(1)  007026  004737  005264                  JSR     PC,$ATY4        ;CALL APT SERVICE
(1)  007032  000000          5$:     .WORD   0                       ;ERROR NUMBER STUCK HERE
(1)  007034  000777          10$:    BR      10$                     ;LOCK UP HERE
(1)  007036  022737  004340  000042  15$:    CMP     #$ENDAD,@#42    ;CHECK TO SEE IF IN ACT-11 MODE
(1)  007044  001403                          BEQ     20$             ;IF SO, HANDLE ACCORDINGLY
(1)  007046  005777  172232                  TST     @SWR            ;HALT ON ERROR?
(1)  007052  100004                          BPL     EXITER          ;BR IF NO HALT ON ERROR
(1)  007054  016677  000002  172224  20$:    MOV     2(SP),@DISPLAY  ;SHOW ERROR PC IN DATA DISPLAY
(1)  007062  000000                          HALT                    ;HALT
(1)  007064  005237  001256          EXITER: INC     $ERTTL          ;UPDATE ERROR COUNT
(1)  007070  004737  007150                  JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
(1)  007074  032777  C004C0  172202          BIT     #SWC8,@SWR      ;GOTO TOP OF TEST?
(1)  007102  001007                          BNE     1$              ;BR IF YES
(1)  007104  032777  002000  172172          BIT     #SW10,@SWR      ;GOTO NEXT TEST?
(1)  007112  001407                          BEQ     2$              ;BR IF NO
(1)  007114  013737  001362  001252          MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
(1)  007122  012706  001120          1$:     MOV     #STACK,SP       ;RESET SP
(1)  007126  000177  172120                  JMP     @$LPADR         ;GOTO SPECIFIED TEST
(1)  007132  000002          2$:     RTI                             ;RETURN
(1)  007134  000001          ERTABO: 1
(1)  007136     006     002                  .BYTE   6,2
(1)  007140  001404                          SAVPC
(1)  007142  000001          XTSTN:  1
(1)  007144     002     002                  .BYTE   2,2
(1)  007146  001246                          $TSTNM
```

```
(1)  007150  017746  172136          SERV.G: MOV     @$TKB,-(SP)     ;OTHERWISE, GET THE LAST CHARACTER TYPED
(1)  007154  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY(EIGHTH) BIT
(1)  007160  122726  000007                  CMPB    #7,(SP)+        ;IS IT ^G?
(1)  007164  001076                          BNE     6$              ;IF NOT, IGNORE INPUT
(1)  007166  032777  004000  172114          BIT     #4000,@$TKS     ;RX BUSY?
(1)  007174  001365                          BNE     SERV.G          ;BR IF YES
(1)       007176                     GETSWR= .                               ;;GPA
(1)  007176  017737  172102  007404          MOV     @SWR,90$        ;SAVE (SWR).
(1)  007204  104402  007364          1$:     TYPE    ,89$            ;TYPE HEADER FOR OLD SWITCH REGISTER
(1)  007210  104412  007376                  CNVRT   ,88$            ;TYPE THE NUMBER ITSELF
(1)  007214  104402  007406                  TYPE    ,91$            ;AFTER HAVING CONVERTED IT TO ASCII
(1)  007220  105037  007412                  CLRB    92$             ;CLEAR SWR CHANGE FLAG
(1)  007224  005077  172054                  CLR     @SWR            ;CLEAR THE SOFTWARE SWITCH REGISTER
(1)  007230  105777  172054          3$:     TSTB    @$TKS           ;WAIT FOR DONE.
(1)  007234  100375                          BPL     3$              ;CONTINUE WAITING FOR IT
(1)  007236  017746  172050                  MOV     @$TKB,-(SP)     ;PUT THE CHARACTER ON THE STACK
(1)  007242  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY BIT
(1)  007246  122726  000015                  CMPB    #15,(SP)+       ;IS IT THE CARRIAGE RETURN CHAR?
(1)  007252  001433                          BEQ     4$              ;IF SO, GO PRINT CRLF
(1)  007254  105777  172034          2$:     TSTB    @$TPS           ;IS THE OUTPUT BUFFER AVAILABLE
(1)  007260  100375                          BPL     2$              ;IF NOT, WAIT FOR IT TO BE READY
(1)  007262  105237  007412                  INCB    92$             ;INDICATE THAT THE SWR WAS CHANGED
(1)  007266  014677  172024                  MOV     -(SP),@$TPB     ;PLACE THE CHARACTER THERE(ECHO BACK)
(1)  007272  000241                          CLC                     ;GET READY TO ROTATE
(1)  007274  006177  172004                  ROL     @SWR            ;MOVE THE EXISTING BITS OVER
(1)  007300  006177  172000                  ROL     @SWR            ;TO MAKE ROOM FOR THE INCOMING
(1)  007304  006177  171774                  ROL     @SWR            ;THREE BITS FROM THIS CHARACTER
(1)  007310  103735                          BCS     1$              ;ERROR
(1)  007312  022627  000060                  CMP     (SP)+,#60       ;IS IT LOWER THAN 0?
(1)  007316  002732                          BLT     1$              ;IF SO, GO ASK AGAIN
(1)  007320  026627  177776  000067          CMP     -2(SP),#67      ;IS IT HIGHER THAN 7?
(1)  007326  003326                          BGT     1$              ;IF SO, GO ASK AGAIN
(1)  007330  042746  177770                  BIC     #^C<7>,-(SP)    ;ISOLATE INFORMATION BITS
(1)  007334  052677  171744                  BIS     (SP)+,@SWR      ;ADD THEM TO THE SWITCH REGISTER
(1)  007340  000733                          BR      3$              ;GO CHECK FOR THE NEXT CHARACTER
(1)  007342  105737  007412          4$:     TSTB    92$             ;HAS THE SWR BEEN CHANGED?
(1)  007346  001003                          BNE     5$              ;IF YES GO TYPE CRLF
(1)  007350  013777  007404  171726          MOV     90$,@SWR        ;IF NOT RESTORE SWR
(1)  007356  104402  001357          5$:     TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
(1)  007362  000207                  6$:     RTS     PC              ;RETURN TO CALLING PROCEDURE
(1)
(1)  007364  020200  051450  051127  89$:     .ASCIZ  <200>? (SWR)=/?
(1)                                  .EVEN
(1)  007376  000001                  88$:     1
(1)  007400     006     000                  .BYTE   6,0
(1)  007402  007404                          90$
(1)  007404  000000                  90$:     .WORD   0
(1)  007406  036457  000057          91$:     .ASCIZ  ?/=/?
(1)  007412     000                  92$:     .BYTE   0
(1)       007414                     .EVEN
(2)                                  .SBTTL  POWER DOWN AND UP ROUTINES
(2)
(3)                                  ;;*****************************************************************************
(2)                                  ;POWER DOWN ROUTINE
(2)  007414  012737  007560  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(2)  007422  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
```

```
(4)  007430  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
(4)  007432  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
(4)  007434  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
(4)  007436  010346                          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
(4)  007440  010446                          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
(4)  007442  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
(4)  007444  017746  171634                  MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
(2)  007450  010637  007564                  MOV     SP,$SAVR6         ;;SAVE SP
(2)  007454  012737  007466  000024          MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
(2)  007462  000000                          HALT
(2)  007464  000776                          BR      .-2               ;;HANG UP
(2)
(3)                                  ;;*****************************************************************
(2)                                  ;POWER UP ROUTINE
(2)  007466  012737  007560  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
(2)  007474  013706  007564                  MOV     $SAVR6,SP         ;;GET SP
(2)  007500  005037  007564                  CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
(2)  007504  005237  007564          1$:     INC     $SAVR6            ;;WAIT FOR THE INC
(2)  007510  001375                          BNE     1$                ;;OF  WORD
(4)  007512  012677  171566                  MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
(4)  007516  012605                          MOV     (SP)+,R5          ;;POP STACK INTO R5
(4)  007520  012604                          MOV     (SP)+,R4          ;;POP STACK INTO R4
(4)  007522  012603                          MOV     (SP)+,R3          ;;POP STACK INTO R3
(4)  007524  012602                          MOV     (SP)+,R2          ;;POP STACK INTO R2
(4)  007526  012601                          MOV     (SP)+,R1          ;;POP STACK INTO R1
(4)  007530  012600                          MOV     (SP)+,R0          ;;POP STACK INTO R0
(2)  007532  012737  007414  000024          MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
(2)  007540  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
(2)  007546  104402                          TYPE                      ;;REPORT THE POWER FAILURE
(2)  007550  007566                  $PWRMG: .WORD   MPFAIL            ;;POWER FAIL MESSAGE POINTER
(2)  007552  012716                          MOV     (PC)+,(SP)        ;;RESTART AT RESTART
(2)  007554  011112                  $PWRAD: .WORD   RESTART           ;;RESTART ADDRESS
(2)  007556  000002                          RTI
(2)  007560  000000                  $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
(2)  007562  000776                          BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
(2)  007564  000000                  $SAVR6: 0                         ;;PUT THE SP HERE
(2)  007566  050200  051127  043040  MPFAIL: .ASCIZ   <200>/PWR FAILED. RESTART AT LAST TEST /
(2)  007631     200  047105  020104  MEPASS: .ASCIZ   <200>/END PASS CVDZA-C  /
(2)  007655     200  052522  047116  MR:     .ASCIZ   <200>/RUNNING   /
(2)  007671     200  051120  043517  MERR2:  .ASCIZ   <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2)  007740  044600  051516  043125  MERR3:  .ASCIZ   <200>/INSUFFICIENT DATA!/
(2)  007764  046200  041517  020113  MLOCK:  .ASCIZ   <200>/LOCK ON SELECTED TEST/
(2)  010013     103  051123  020072  MCSRX:  .ASCIZ    /CSR: /
(2)  010021     126  041505  020072  MVECX:  .ASCIZ   /VEC: /
(2)  010027     120  051501  042523  MPASSX: .ASCIZ   /PASSES: /
(2)  010040  051105  047522  051522  MERRX:  .ASCIZ   /ERRORS: /
(2)  010051     124  051505  020124  MTSTN:  .ASCIZ   /TEST NO: /
(2)  010063     052  000040          MASTEK: .ASCIZ   /* /
(2)  010066  052200  050131  020105  MNEW:   .ASCIZ   <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE:  /
(2)  010143     120  035103  000040  MERRPC: .ASCIZ   /PC: /
(2)  010150  046600  050101  047440  XHEAD:  .ASCIZ   <200>/MAP OF DZV11 STATUS/<200>
(2)  010176  044600  046114  043505  MBADLN: .ASCIZ   <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)                                  .EVEN
(2)  010240  000002                  XSTATQ: 2
(2)  010242     006     003                  .BYTE   6,3
(2)  010244  001344                          $TMP1
```

```
(2)  010246  006     002                    .BYTE   6,2
(2)  010250  001346                         $TMP2
(1)                                  .EVEN
(2)                                  ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
(2)                                  ;-------------------------------------------------------------
(2)                                  ;E=EXTERNAL LOOP BACK
(2)                                  ;I=INTERNAL LOOP BACK
(2)                                  ;S=STAGGERED LOOP BACK
(2)  010252  017605  000000          .SETFLG:MOV     @(SP),R5        ;PICK UP ADDRESS OF TAG
(2)  010256  042737  000040  010446          BIC     #40,INBUF       ;STRIP LOWER CASE
(2)  010264  122737  000105  010446          CMPB    #'E,INBUF       ;IS IT EXTERNAL LOOP BACK ?
(2)  010272  001005                          BNE     4$              ;NO
(2)  010274  013715  010364                  MOV     1$,(R5)         ;YES STORE INFO
(2)  010300  105037  001424                  CLRB    MNTFLG          ;SET MAINT BIT =0
(2)  010304  000422                          BR      7$              ;GET OUT
(2)  010306  122737  000111  010446  4$:     CMPB    #'I,INBUF       ;IS IT INTERNAL LOOP BACK ?
(2)  010314  001006                          BNE     5$              ;NO
(2)  010316  013715  010366                  MOV     2$,(R5)         ;YES STORE INFO
(2)  010322  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;SET UP THE MAINTENANCE FLAG LOADER
(2)  010330  000410                          BR      7$              ;GET OUT
(2)  010332  122737  000123  010446  5$:     CMPB    #'S,INBUF       ;IS IT STAGGERED LOOP BACK ?
(2)  010340  001007                          BNE     6$              ;WHAT ?
(2)  010342  013715  010370                  MOV     3$,(R5)         ;YES STORE INFO
(2)  010346  105037  001424                  CLRB    MNTFLG          ;ZERO BITS
(2)  010352  062716  000002          7$:     ADD     #2,(SP)         ;POP AROUND
(2)  010356  000002                          RTI
(2)  010360  104404                  6$:     INSTER                  ;RETRY
(2)  010362  000733                          BR      .SETFLG         ;DITTO
(2)  010364  000200                  1$:     .WORD   200             ;EXTERNAL =  E
(2)  010366  000000                  2$:     .WORD   0               ;INTERNAL = I
(2)  010370  100000                  3$:     .WORD   100000          ;STAGGERED = S
(2)
```

```
(2)                                             ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
(2)                                             ;BUFFER TO THE CHARACTERS "E" AND "C".
(2)                                             ;IF THE CHARACTER IS "E" CLEAR THE FLAG
(2)                                             ;IF THE CHARACTER IS "C" SET THE FLAG
(2)
(2)  010372  017605  000000          .PAWCH:MOV     @(SP),R5
(2)  010376  142737  000040  010446          BICB    #40,INBUF       ;SET FOR LOWER CASE INPUT
(2)  010404  122737  000105  010446          CMPB    #'E,INBUF       ;IS IT "E" ?
(2)  010412  001002                          BNE     1$
(2)  010414  105015                          CLRB    (R5)            ;000
(2)  010416  000406                          BR      2$
(2)  010420  122737  000103  010446  1$:     CMPB    #'C,INBUF       ;IS IT "C" ?
(2)  010426  001005                          BNE     3$
(2)  010430  112715  177777                  MOVB    #-1,(R5)        ;3177
(2)  010434  062716  000002          2$:     ADD     #2,(SP)
(2)  010440  000002                          RTI
(2)  010442  104404                  3$:     INSTER                  ;RETRY
(2)  010444  000752                          BR      .PAWCH
(2)
(2)                                             ;BUFFERS FOR INPUT-OUTPUT
(2)
(2)  010446  000000                  INBUF:  0
(2)          010510                  .=.+40
(2)                                  ; TEMP: 0       ; TEMP AREA UNUSED.                             ;;GPA
(2)                                  ; .=.+40        ; DELETED TO CONSERVE SPACE                     ;;GPA
(2)  010510  000000                  MDATA:  0
(2)          010552                  .=.+40
(2)
```

```
(2)
(2)                                     ;
(2)                                     ;ROUTINE USED TO "CYCLE" THROUGH UP TO SIXTEEN DZV11'S
(2)                                     ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
(2)                                     ;AND RUNS THE SPECIFIED DZV11'S.  THIS ROUTINE *MUST*
(2)                                     ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
(2)                                     ;SETUP NECESSARY.
(2)                                     ;
(2)
(2)  010552  005737  001406        CYCLE: TST     DZVACTV          ;ARE ANY DZV11'S TO BE TESTED?
(2)  010556  001004                       BNE     1$               ;BR IF OK.
(2)  010560  104402  007671               TYPE    ,MERR2           ;NO DZV11'S SELECTED!!
(2)  010564  000000                       HALT                     ;STOP THE SHOW.
(2)  010566  000776                       BR      .-2              ;DISQUALIFY CONT. SW.
(2)  010570  013737  004646  001354  1$:  MOV     $MXCNT,$TIMES    ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
(2)  010576  033737  001412  001406       BIT     RUN,DZVACTV      ;IS THIS ONE "ACTIVE"
(2)  010604  001017                       BNE     2$               ;BR IF GOOD ONE FOUND.
(2)  010606  006137  001412               ROL     RUN              ;UPDATE POINTER
(2)  010612  005537  001412               ADC     RUN              ;CATCH CARRY FROM RUN
(2)  010616  062737  000012  001420       ADD     #12,ACTIVE       ;UPDATE ADDRESS POINTER.
(2)  010624  022737  001740  001420       CMP     #DZV.END,ACTIVE  ;HAVE WE PASSED THE END OF THE MAP?
(2)  010632  001356                       BNE     1$               ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
(2)  010634  012737  001500  001420       MOV     #DZV.MAP,ACTIVE  ;RESET ADDRESS POINTER.
(2)  010642  000752                       BR      1$               ;KEEP LOOKING FOR ACTIVE DZV11
(2)  010644  006137  001412        2$:    ROL     RUN              ;UPDATE POINTER.
(2)  010650  005537  001412               ADC     RUN              ;CATCH CARRY.
(2)  010654  013700  001420               MOV     ACTIVE,R0        ;GET ADDRESS POINTER.
(2)  010660  062737  000012  001420       ADD     #12,ACTIVE       ;UPDATE.
(2)  010666  022737  001740  001420       CMP     #DZV.END,ACTIVE
(2)                                                               ;ALL DONE?
(2)  010674  001003                       BNE     3$               ;BR IF NO.
(2)  010676  012737  001500  001420       MOV     #DZV.MAP,ACTIVE  ;RESTORE POINTER.
(2)  010704  012037  001174        3$:    MOV     (R0)+,$BASE      ;LOAD SYSTEM CTRL. REG
(2)  010710  012037  002040               MOV     (R0)+,DZVRIV     ;LOAD VECTOR
(2)  010714  012037  001366               MOV     (R0)+,LINE       ;SET UP DZV LINES ACTIVE
(2)  010720  012037  001370               MOV     (R0)+,PAR        ;SET UP PARAMETERIZATION
(2)  010724  012037  001372               MOV     (R0)+,MODE       ;SET UP MAINTENANCE MODE
(2)  010730  105037  001424               CLRB    MNTFLG  ;RESET MAINT. FLAG IF
(2)  010734  005737  001372               TST     MODE             ;RUNNING TESTS
(2)  010740  001003                       BNE     9$               ;IN
(2)  010742  112737  000010  001424       MOVB    #MAINT,MNTFLG    ;INTERNAL MAINT. MODE
(2)  010750  004737  011116        9$:    JSR     PC,DZVLEV        ;SET UP
(2)  010754  005737  000042               TST     @#42             ;ARE WE UNDER MONITOR CONTROL?
(2)  010760  001051                       BNE     7$               ;IF YES, SKIP THIS SETUP
(2)  010762  032777  000002  170314       BIT     #SW01,@SWR       ;IF SW01=1, GET STARTINC TEST #
(2)  010770  001445                       BEQ     7$               ;BR IF NO TEST IS TO BE INPUTTED
(2)  010772  104402  001357        4$:    TYPE    ,$CRLF
(3)  010776  104403                       INSTR                    ;CALL THE STRING INPUT ROUTINE
(3)  011000  010051                       MTSTN                    ;POINTER TO MESSAGE TO BE PRINTED
(3)  011002  104405                       PARAM                    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(3)  011004  000001                       1                        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  011006  001000                       1000                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  011010  001246                       $TSTNM                   ;POINTER TO MAP LOCATION TO BE FILLED
(3)  011012     000                       .BYTE   0                ;MASK OF INVALID BITS FOR THIS PARAMETER
(3)  011013     001                       .BYTE   1                ;NUMBER OF PARAMETERS TO STORE
(2)  011014  012700  012112               MOV     #TST1,R0
```

```
(2)  011020  022710  000004          5$:     CMP     #4,(R0)
(2)  011024  001020                          BNE     6$
(2)  011026  022760  012737  000002          CMP     #12737,2(R0)
(2)  011034  001014                          BNE     6$
(2)  011036  023760  001246  000004          CMP     $TSTNM,4(R0)    ;IS THIS THE TEST ?
(2)  011044  001010                          BNE     6$              ;IF NOT, DON'T PROCESS NUMBER
(2)  011046  010037  001252                  MOV     R0,$LPADR       ;SAVE PC
(2)  011052  062737  000002  001252          ADD     #2,$LPADR       ;POP OVER PREVIOUS SCOPE
(2)  011060  104402  001357                  TYPE    ,$CRLF
(2)  011064  000412                          BR      8$
(2)  011066  005720                  6$:     TST     (R0)+
(2)  011070  020027  015570                  CMP     R0,#TLAST+10
(2)  011074  001351                          BNE     5$
(2)  011076  104402  001356                  TYPE    ,$QUES
(2)  011102  000733                          BR      4$
(2)  011104  012737  012112  001252  7$:     MOV     #TST1,$LPADR    ;PREPARE TEST ADDRESS
(2)  011112                          8$:
(2)  011112  000177  170134          RESTART:JMP     @$LPADR         ;GO START TESTING.***WARNING!****
(2)                                                                  ;THIS JUMP IS USED BY POWER UP ROUTINE!!!!
(2)
(2)                                  ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
(2)  011116  013700  002040          DZVLEV: MOV     DZVRIV,R0       ;PLACE THE BASE VECTOR ADDRESS IN R0
(2)  011122  062700  000002                  ADD     #2,R0           ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
(2)  011126  010037  002042                  MOV     R0,DZVRIS       ;STORE IT HERE
(2)  011132  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
(2)  011136  010037  002044                  MOV     R0,DZVTIV       ;STORE IT HERE
(2)  011142  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
(2)  011146  010037  002046                  MOV     R0,DZVTIS       ;STORE IT HERE
(2)
(2)                                  ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
(2)                                  ;OF THE DEVICE
(2)  011152  013700  001174                  MOV     $BASE,R0        ;COPY THE ADDRESS BEING LOADED
(2)  011156  010037  002010                  MOV     R0,DZVCSR       ;XXX0
(2)  011162  005200                          INC     R0
(2)  011164  010037  002012                  MOV     R0,HDZVCSR      ;XXX1
(2)  011170  005200                          INC     R0
(2)  011172  010037  002014                  MOV     R0,DZVRBUF      ;XXX2
(2)  011176  010037  002020                  MOV     R0,DZVLPR       ;XXX2
(2)  011202  005200                          INC     R0
(2)  011204  010037  002016                  MOV     R0,HDZVRBUF     ;XXX3
(2)  011210  010037  002022                  MOV     R0,HDZVLPR      ;XXX3
(2)  011214  005200                          INC     R0
(2)  011216  010037  002024                  MOV     R0,DZVTCR       ;XXX4
(2)  011222  005200                          INC     R0
(2)  011224  010037  002026                  MOV     R0,HDZVTCR      ;XXX5
(2)  011230  005200                          INC     R0
(2)  011232  010037  002030                  MOV     R0,DZVMSR       ;XXX6
(2)  011236  010037  002034                  MOV     R0,DZVTDR       ;XXX6
(2)  011242  005200                          INC     R0
(2)  011244  010037  002032                  MOV     R0,HDZVMSR      ;XXX7
(2)  011250  010037  002036                  MOV     R0,HDZVTDR      ;XXX7
(2)  011254  000207                          RTS     PC
```

```
(2)                                     ;CONVERT DECIMAL ASCII STRING TO OCTAL
(2)  011256  000002             .PARMD: RTI             ; DECIMAL PARAMETERS UNUSED.        ;;GPA
(2)                             .REM &                  ; DELETED TO CONSERVE SPACE...      ;;GPA
(2)                                                     ;...AND REMAIN UNDER 4KW SIZE.      ;;GPA
(2)                             .PARMD: MOV     (SP),R5
(2)                                     MOV     (R5)+,6$
(2)                                     MOV     (R5)+,7$
(2)                                     MOV     (R5)+,8$
(2)                                     MOVB    (R5)+,9$
(2)                                     MOVB    (R5)+,10$
(2)                                     MOV     R5,(SP)
(2)                             2$:     CLR     R5
(2)                                     MOV     #INBUF,R4
(2)                                     CMPB    #15,(R4)
(2)                                     BEQ     3$
(2)                             1$:     CMPB    (R4),#'0
(2)                                     BLT     3$
(2)                                     CMPB    (R4),#'9
(2)                                     BGT     3$
(2)                                     BICB    #'0,(R4)
(2)                                     CLR     R2
(2)                                     BISB    (R4)+,R2
(2)                                     ADD     R2,R5
(2)                                     CMPB    #15,(R4)
(2)                                     BEQ     4$
(2)                                     ASL     R5      ;X2
(2)                                     MOV     R5,R2   ;SAVE X2
(2)                                     ASL     R5      ;X4
(2)                                     ASL     R5      ;X8
(2)                                     ADD     R2,R5   ;TIMES 10
(2)                                     BR      1$
(2)                             3$:     INSTER
(2)                                     BR      2$
(2)
(2)                                     ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(2)
(2)                             4$:     CMP     R5,7$
(2)                                     BHI     3$
(2)                                     CMP     R5,6$
(2)                                     BLO     3$
(2)                                     BITB    9$,R5
(2)                                     BNE     3$
(2)
(2)                                     ;STORE NUMBER AT SPECIFIED ADDRESS
(2)
(2)                                     MOV     8$,R4
(2)                             5$:     MOV     R5,(R4)+
(2)                                     ADD     #2,R5
(2)                                     DECB    10$
(2)                                     BNE     5$
(2)                                     RTI
(2)                             6$:     0
(2)                             7$:     0
(2)                             8$:     0
(2)                             9$:     .BYTE 0
(2)                             10$:    .BYTE 0
```

```
   (2)                                  ; END OF .PARMD DELETE RANGE          &                ;;GPA
```

```
(2)                                         ;*ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
(2)                                         ;*IF BIT7 IN THE ENVIRONMENT MODE ($ENVM) BYTE IS SET,
(2)                                         ;*THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
(2)
(2)  011260  012700  001500          SETAPT: MOV     #DZV.MAP,R0     ;POINT TO THE DEVICE MAP TABLE
(2)  011264  013701  001174                  MOV     $BASE,R1        ;BUILD DEVICE ADDRESSES IN R1
(2)  011270  013702  001170                  MOV     $VECT1,R2       ;BUILD DEVICE VECTORS IN R2
(2)  011274  042702  177007                  BIC     #^C<770>,R2     ;STRIP AWAY OTHER INFORMATION
(2)
(2)  011300  012704  001204                  MOV     #$DDW0,R4       ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
(2)  011304  013705  001176                  MOV     $DEVM,R5        ;GET THE MAP OF ACTIVE DEVICES
(2)  011310  105037  001414                  CLRB    DZVNUM          ;INITIALIZE NO. OF DEVICES IN SYSTEM
(2)  011314  005037  001410                  CLR     SAVACTV         ;CLEAR THE ACTIVE BIT MAP
(2)  011320  006005                  1$:     ROR     R5              ;GET A DEVICE SELECTION BIT
(2)  011322  103407                          BCS     3$              ;IF IT IS SELECTED, GO SET UP A MAP
(2)  011324  001422                          BEQ     5$              ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
(2)  011326  005724                          TST     (R4)+           ;POINT TO NEXT DEVICE DESCRIPTOR
(2)  011330  062701  000010          2$:     ADD     #10,R1          ;SET UP THE NEXT ADDRESS
(2)  011334  062702  000010                  ADD     #10,R2          ;SET UP THE NEXT VECTOR GROUP
(2)  011340  000767                          BR      1$              ;GO SEE IF MORE DEVICES REMAIN
(2)  011342  006137  001410          3$:     ROL     SAVACTV         ;SET BIT IN ACTIVE DEVICE MAP
(2)  011346  105237  001414                  INCB    DZVNUM          ;INCREMENT NO. OF ACTIVE DEVICES IN SYSTEM
(2)  011352  010120                          MOV     R1,(R0)+        ;LOAD DEVICE ADDRESS
(2)  011354  010220                          MOV     R2,(R0)+        ;LOAD THE VECTOR ADDRESS
(2)  011356  013720  001200                  MOV     $CDW1,(R0)+     ;GET THE NUMBER OF LINES IN OPERATION
(2)  011362  012420                          MOV     (R4)+,(R0)+     ;LOAD DEVICE PARAMETERS
(2)  011364  013720  001202                  MOV     $CDW2,(R0)+     ;LOAD DEFAULT TESTING MODE
(2)  011370  000757                          BR      2$              ;GO BUILD THE NEXT ADDRESS
(2)  011372  012710  177777          5$:     MOV     #-1,(R0)        ;TERMINATE THE DEVICE MAP
(2)  011376  012737  001142  001304          MOV     #$SWREG,SWR     ;SET TO SOFTWARE APT SWITCH REGISTER
(2)  011404  000207                          RTS     PC              ;RETURN TO PRINT STATUS TABLE
(2)
(2)
(2)                                         ;*ROUTINE USED TO "AUTO SIZE" THE DZV11
(2)                                         ;*CSR AND VECTOR.
(2)                                         ;*NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
(2)                                         ;*      ADDRESS RANGE (160000:163770)
(2)                                         ;*      AND THE VECTOR MAY BE ANY WHERE IN THE
(2)                                         ;*      FLOATING VECTOR RANGE (300:770)
(2)                                         ;*
(2)
(2)  011406                          AUTO.SIZE:
(2)  011406  000005                          RESET                   ;INSURE A BUS INIT.
(2)  011410  105337  001422                  DECB    INIFLG          ;SHOW THAT I WAS HERE
(2)  011414  012702  001500          CSRMAP: MOV     #DZV.MAP,R2     ;LOAD MAP POINTER.
(2)  011420  012703  001204                  MOV     #$DDW0,R3       ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
(2)  011424  005022                  1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
(2)  011426  022702  001740                  CMP     #DZV.END,R2     ;ALL DONE?
(2)  011432  001374                          BNE     1$              ;BR IF NO
(2)  011434  105037  001414                  CLRB    DZVNUM          ;SET OCTAL NUMBER OF DZV11'S TO 0
(2)  011440  012702  001500                  MOV     #DZV.MAP,R2
(2)  011444  012701  160000                  MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
(2)  011450  012737  011714  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTENT DEVICE TIME OUT
(2)  011456  052711  000040          2$:     BIS     #BIT5,(R1)      ;TRY TO SET MASTER SCAN ENABLE
(2)  011462  052761  000017  000004          BIS     #17,4(R1)       ;TRY TO TRANSMIT ON ANY LINE
(2)  011470  005000                          CLR     R0              ;USE R0 AS A COUNTER
```

```
(2) 011472 005711                    7$:    TST    (R1)          ;HAS TRANSMITTER READY COME UP?
(2) 011474 100403                           BMI    8$            ;IF SO, GO GET A FINAL CHECK
(2) 011476 005300                           DEC    R0            ;REDUCE COUNT. TIME UP?
(2) 011500 001374                           BNE    7$            ;IF NOT, KEEP WAITING
(2) 011502 000437                           BR     3$            ;ASSUME IT'S NOT A DZV11
(2) 011504 032761 000017 000004     8$:    BIT    #17,4(R1)     ;ARE ANY TCR BITS  STILL SET? THEY SHOULD BE
(2) 011512 001433                           BEQ    3$            ;IF IT'S NOT, ASSUME IT'S NOT A DZV11
(2) 011514 032711 000040                    BIT    #BIT5,(R1)    ;IS MASTER SCAN ENABLE STILL SET?
(2) 011520 001430                           BEQ    3$            ;IF NOT, ASSUME IT'S NOT A DZV11
(2) 011522 052711 000020                    BIS    #20,(R1)      ;SET DEVICE CLEAR
(2) 011526 000240                           NOP
(2) 011530 032711 000040                    BIT    #40,(R1)      ;DID SCANNER CLEAR
(2) 011534 001022                           BNE    3$            ;IF NOT ASSUME IT IS NOT DZV
(2) 011536 005061 000004                    CLR    4(R1)         ;GET RID OF TCR BITS
(2)                         ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DZV11 CSR ADDRESS.
(2) 011542 010122                           MOV    R1,(R2)+      ;STORE CSR IN CORE TABLE.
(2) 011544 005722                           TST    (R2)+         ;POP OVER VECTOR STORE AREA
(2) 011546 012722 000017                    MOV    #17,(R2)+     ;SET THE DEFAULT LINE SELECTION PARAMETER
(2) 011552 012712 017470                    MOV    #17470,(R2)   ;SET THE DEFAULT PARAMETERS
(2) 011556 012223                           MOV    (R2)+,(R3)+   ;COPY PARAMETERS INTO ETABLE DESCRIPTOR
(2) 011560 005022                           CLR    (R2)+         ;SET THE DEFAULT MODE OF OPERATION
(2) 011562 012712 177777                    MOV    #-1,(R2)      ;TERMINATE LIST
(2) 011566 105237 001414                    INCB   DZVNUM        ;UPDATE DEVICE COUNTER
(2) 011572 122737 000020 001414             CMPB   #20,DZVNUM    ;ARE MAX. NO. OF DEV FOUND?
(2) 011600 001405                           BEQ    100$          ;YES DON'T LOOK FOR ANY MORE.
(2) 011602 062701 000010            3$:     ADD    #10,R1        ;UPDATE CSR POINTER ADDRESS
(2) 011606 022701 164000                    CMP    #164000,R1
(2) 011612 001321                           BNE    2$            ;BR IF MORE ADDRESS TO CHECK.
(2) 011614                          100$:
(2) 011614 105737 001414                    TSTB   DZVNUM        ;WERE ANY DZV11'S FOUND AT ALL?
(2) 011620 001430                           BEQ    5$            ;ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
(2) 011622 113701 001414                    MOVB   DZVNUM,R1
(2) 011626 012737 000001 001410             MOV    #1,SAVACTV    ;CREATE A BIT MAP OF THE ACTIVE
(2) 011634 005301                   4$:     DEC    R1            ;DEVICES IN THE SYSTEM
(2) 011636 001404                           BEQ    98$
(2) 011640 000261                           SEC
(2) 011642 006137 001410                    ROL    SAVACTV
(2) 011646 000772                           BR     4$
(2) 011650 013737 001500 001174     98$:    MOV    DZCR0,$BASE   ;POINT TO THE ADDRESS OF FIRST DEVICE
(2) 011656 013737 001510 001202             MOV    MANT0,$CDW2   ;INDICATE TO ETABLE WHAT MODE IS BEING USED
(2) 011664 012737 000006 000004     99$:    MOV    #6,@#4        ;RESTORE TRAP VECTOR
(2) 011672 013737 001410 001176             MOV    SAVACTV,$DEVM ;SAVE ACTIVE REGISTER
(2) 011700 000410                           BR     VECMAP        ;GO FIND THE VECTOR NOW.
(2) 011702 104402 007671            5$:     TYPE   ,MERR2        ;NOTIFY OPR THAT NO DZV11'S FOUND.
(2) 011706 005000                           CLR    R0            ;MAKE DATA DISPLAY ZERO
(2) 011710 000000                           HALT                 ;STOP THE SHOW
(2) 011712 000776                           BR     .-2           ;DISABLE CONT. SW.
(2) 011714 012716 011602            6$:     MOV    #3$,(SP)      ;ENTERED BY NON-EXISTENT TIME-OUT
(2) 011720 000002                           RTI                  ;RETURN TO MAINSTREAM
(2)
(2) 011722 012737 000200 000022     VECMAP: MOV    #MASK,@#22    ;SET IOT TRAP PRIORITY
(2) 011730 012737 012044 000020             MOV    #4$,@#20      ;SET IOT TRAP VECTOR
(2) 011736 012702 001500                    MOV    #DZV.MAP,R2   ;SET SOFTWARE POINTER
(2) 011742 012700 000300                    MOV    #300,R0       ;FLOATING VECTORS START HERE.
(2) 011746 012701 000302                    MOV    #302,R1       ;PC OF IOT INSTR.
(2) 011752 010120                   1$:     MOV    R1,(R0)+      ;START FILLING VECTOR AREA
```

```
    (2)  011754  012721  000004                        MOV     #4,(R1)+        ;WITH -+2: IOT
    (2)  011760  022021                                CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
    (2)  011762  020127  001000                        CMP     R1,#1000        ;HAS THE VECTOR AREA BEEN EXCEEDED?
    (2)  011766  101771                                BLOS    1$              ;BR IF MORE TO FILL
    (2)  011770  013704  001410                        MOV     SAVACTV,R4      ;STORE TEMPORARILY
    (2)  011774  006004                         2$:   ROR     R4              ;BRING OUT A BIT
    (2)  011776  103036                                BCC     5$              ;BR IF ALL DONE
    (2)  012000  106427  000000                        MTPS    #0              ;ZERO CPU PRIO
    (2)  012004  012772  040040  000000                MOV     #BIT14+BIT5,@(R2)       ;SET TIE AND MAS SCAN
    (2)  012012  011201                                MOV     (R2),R1         ;GET CSR
    (2)  012014  112761  000017  000004                MOVB    #17,4(R1)       ;SET THE TCR BITS FOR ALL LINES
    (2)                                                                        ;ATTEMPT TO FORCE AN INTERRUPT
    (2)  012022  005200                                INC     R0              ;STALL
    (2)  012024  001376                                BNE     .-2             ;       FOR TIME TO INTERRUPT
    (2)  012026  012762  000300  000002                MOV     #300,2(R2)      ;NO INTERRUPT ASSUME 300 AND FIX DZV11 LATER
    (2)  012034  000005                                RESET                   ;INIT
    (2)  012036  062702  000012                 3$:    ADD     #12,R2          ;POP SOFTWARE POINTER
    (2)  012042  000754                                BR      2$              ;KEEP GOING
    (2)  012044  011662  000002                 4$:    MOV     (SP),2(R2)      ;GET VECTOR ADDRESS
    (2)  012050  162762  000010  000002                SUB     #10,2(R2)       ;POINT BACK TO THE CORRECT VECTOR
    (2)  012056  042762  000007  000002                BIC     #7,2(R2)        ;CLEAR JUNK
    (2)  012064  022626                                POP2SP                  ;POP IOT JUNK OFF STACK
    (2)  012066  012716  012036                        MOV     #3$,(SP)        ;SET FOR RETURN
    (2)  012072  000002                                RTI
    (2)  012074  013737  001502  001170         5$:    MOV     DZVC0,$VECT1    ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
    (2)  012102  012737  004404  000020                MOV     #.SCOPE,IOTVEC  ;RESTORE THE SCOPE TRAP
    (2)  012110  000207                                RTS     PC              ;ALL DONE WITH "AUTO SIZING"
    (2)
```

```
                                        ;************************* TEST 1 *********************************
                                        ;*THIS TEST PROVES THE BUS REPLY RESPONSE
                                        ;*DURING A READ OR WRITE TO THE FOLLOWING ADDRESS:
                                        ;*      DZVCSR, DZVRBUF, DZVTCR, DZVMSR
                                ;:*  TEST 1
                                ;:*************************************************************************
012112  000004                TST1:   SCOPE
012114  012737  000001  001246        MOV     #1,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
012122  012737  012302  001362        MOV     #TST2,NEXT      ;POINT TO THE START OF THE NEXT TEST
012130  012737  012270  000004        MOV     #5$,4           ;SET TRAP VECTOR
012136  012737  000200  000006        MOV     #MASK,6         ;SET PRIORITY TO HIGH(MASK INTERRUPTS)
012144  012737  012152  001364        MOV     #1$,LOCK        ;SET RETURN IF SW09=11
012152  013700  002010            1$: MOV     DZVCSR,R0       ;SET ADDRESS TO TEST
012156  011001                      MOV     (R0),R1         ;READ THE ADDRESS
012160  000240                      NOP                     ;WASTE TIME
012162  005010                      CLR     (R0)            ;WRITE THE ADDRESS
012164  000240                      NOP                     ;WASTE TIME
012166  012737  012174  001364        MOV     #2$,LOCK        ;SET RETURN ADDRESS FOR SW09
012174  013700  002014            2$: MOV     DZVRBUF,R0      ;SET ADDRESS TO TEST
012200  011001                      MOV     (R0),R1         ;READ THE ADDRESS
012202  000240                      NOP                     ;
012204  005010                      CLR     (R0)            ;WRITE THE ADDRESS
012206  000240                      NOP                     ;WASTE TIME
012210  012737  012216  001364        MOV     #3$,LOCK        ;SET RETURN ADDRESS FOR SW09
012216  013700  002024            3$: MOV     DZVTCR,R0       ;SET ADDRESS TO TEST
012222  011001                      MOV     (R0),R1         ;READ THE ADDRESS
012224  000240                      NOP
012226  005010                      CLR     (R0)            ;WRITE THE ADDRESS
012230  000240                      NOP
012232  012737  012240  001364        MOV     #4$,LOCK        ;SET RETURN ADDRESS
012240  013700  002030            4$: MOV     DZVMSR,R0       ;SET ADDRESS TO TEST
012244  011001                      MOV     (R0),R1         ;READ FROM ADDRESS
012246  000240                      NOP
012250  005010                      CLR     (R0)            ;WRITE THE ADDRESS
012252  000240                      NOP                     ;
012254  012737  000006  000004        MOV     #6,4            ;SET TRAP CATCHER BACK TO NORMAL
012262  005037  000006                CLR     6               ;
012266  104400                      ADVANCE                 ;SCOPE THIS TEST
012270  011601                5$: MOV     (SP),R1         ;SAVE PC OF TRAP
012272  022626                      POP2SP                  ;POP TRAP OFF STACK
012274  104001                      ERROR   1               ;*NO BUS REPLY RESPONSE.
012276  104401                      SCOP1                   ;SW09=1?
012300  000111                      JMP     (R1)            ;RTI
                                        ;************************* TEST 2 *********************************
                                        ;*THIS TEST PROVES THAT BIT 'DCLR"
                                        ;*CAN BE SET AND THAT IT WILL CLEAR
                                        ;*BY ITSELF
                                ;:*  TEST 2
                                ;:*************************************************************************
012302  000004                TST2:   SCOPE
012304  012737  000002  001246        MOV     #2,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
012312  012737  012346  001362        MOV     #TST3,NEXT      ;POINT TO THE START OF THE NEXT TEST
012320  013700  002010                MOV     DZVCSR,R0       ;SET POINTER
012324  012710  000020                MOV     #DCLR,(R0)      ;SET DCLR
012330  005005                      CLR     R5              ;SET EXPECTED TO 0
```

```
 3044  012332  005003                        CLR     R3              ;DUAL LOOP COUNTER
 3045  012334  011004                2$:     MOV     (R0),R4         ;IS DCLR CLEAR?
 3046  012336  001403                        BEQ     3$              ;IF YES , GO TO THE NEXT TEST
 3047  012340  105203                        INCB    R3              ;IF NO,COUNT 1 OF 256 TICKS
 3048  012342  001374                        BNE     2$              ;HAS THE TIME EXPIRED? IF NO, GO TEST BIT AGAIN
 3049  012344  104002                        ERROR   2               ;*DCLR FAILED TO CLEAR
 3050  012346                        3$:
 3051                                        ;************************ TEST 3 ***************  **************
   (1)                                       ;*TEST TO VERIFY THAT THE R/W BITS OF THE
   (1)                                       ;*DZVCSR REGISTER CAN BE SET. THEN VERIFY THAT
   (1)                                       ;*THESE BITS CAN BE CLEARED. AND FINALLY, VERIFY
   (1)                                       ;*THAT AFTER BEING SET AGAIN THEY CAN BE
   (1)                                       ;*CLEARED BY A "DEVICE CLEAR".
   (1)                                       ;*THE BITS TESTED ARE:  MAINT, MSENAB, SILOEN,
   (1)                                       ;*RIE, AND TIE.
   (3)                                ;:*  TEST 3
   (6)                                ;:*************************************************************************
   (5)  012346  000004                TST3:   SCOPE
   (3)  012350  012737  000003  001246        MOV     #3,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
   (3)  012356  012737  012524  001362        MOV     #TST4,NEXT      ;POINT TO THE START OF THE NEXT TEST
   (1)  012364  013700  002010                MOV     DZVCSR,R0       ;GET BASE ADDRESS
   (1)  012370  012703  012504                MOV     #5$,R3          ;SET R3 TO TOP OF TABLE
   (1)  012374  011305                1$:     MOV     (R3),R5         ;SET BIT
   (1)  012376  012737  012404  001364        MOV     #11$,LOCK       ;SETUP FOR TIGHT SCOPE LOOP
   (1)  012404  010510                11$:    MOV     R5,(R0)         ;SET BIT IN DEVICE
   (1)  012406  011004                        MOV     (R0),R4         ;READ THE BIT FROM DEVICE
   (1)  012410  020504                        CMP     R5,R4           ;WAS BIT SET?
   (1)  012412  001401                        BEQ     2$              ;BR IF YES
   (1)  012414  104002                        ERROR   2               ;*BIT R/W FAILURE
   (1)  012416  104401                2$:     SCOP1                   ;IS SWITCH 9 SET?
   (1)  012420  012737  012426  001364        MOV     #12$,LOCK       ;SET FOR NEXT TIGHT SCOPE LOOP
   (1)  012426  040510                12$:    BIC     R5,(R0)         ;CLEAR THE BIT.
   (1)  012430  011004                        MOV     (R0),R4         ;READ DEVICE
   (1)  012432  001403                        BEQ     3$              ;BR IF BITS WERE CLEARED.
   (1)  012434  005005                        CLR     R5              ;CLEAR FOR ERROR PRINTOUT
   (1)  012436  104002                        ERROR   2               ;*BIT FAILED TO CLEAR
   (1)  012440  011305                        MOV     (R3),R5         ;RESTORE THE BIT.
   (1)  012442  104401                3$:     SCOP1                   ;SW09 SET?
   (1)  012444  012737  012452  001364        MOV     #13$,LOCK       ;SET UP FOR NEXT TIGHT SCOPE
   (1)  012452  010510                13$:    MOV     R5,(R0)         ;SET THE BIT AGAIN
   (1)  012454  104413                        DEVICE.CLR              ;ISSUE DEVICE CLEAR
   (1)  012456  011004                        MOV     (R0),R4         ;READ THE BIT.
   (1)  012460  001403                        BEQ     4$              ;BR IF BIT CLEARED BY INIT (DEVICE CLEAR)
   (1)  012462  005005                        CLR     R5              ;SET EXPECTED TO ZERO
   (1)  012464  104002                        ERROR   2               ;*BIT NOT CLEARED BY DEVICE CLEAR
   (1)  012466  011305                        MOV     (R3),R5         ;RESTORE BIT AGAIN
   (1)  012470  104401                4$:     SCOP1                   ;SW09 SET?
   (1)  012472  062703  000002                ADD     #2,R3           ;POP R3
   (1)  012476  005713                        TST     (R3)            ;IS THIS THE END OF TABLE?
   (1)  012500  001407                        BEQ     6$              ;IF YES GET OUT
   (1)  012502  000734                        BR      1$              ;OTHERWISE TEST NEXT BIT
   (1)  012504  000010                5$:     #MAINT                  ;CSR BIT: INTERNAL MAINTENANCE
   (1)  012506  000040                        #MSENAB                 ;CSR BIT: MASTER SCAN ENABLE
   (1)  012510  010000                        #SILOEN                 ;CSR BIT: SILO ENABLE
   (1)  012512  000100                        #RIE                    ;CSR BIT: RECEIVER INTER. ENABLE
   (1)  012514  040000                        #TIE                    ;CSR BIT: TRANS. INTER. ENABLE
```

```
(1)  012516  000000                        #0                      ;END OF TABLE
(1)  012520  005037  001364          6$:    CLR     LOCK            ;ZERO LOCK INDICATOR
3052                                        ;************************ TEST 4 ********************************
(1)                                         ;*THIS TESTS THAT ALL OF THE TCR BITS
(1)                                         ;*CAN BE: SET, CLEARED, AND CLEARED BY A DEVICE CLEAR.
(1)                                         ;*THIS TEST ALSO DETERMINES IF THE DTR BITS CAN
(1)                                         ;*BE SET, CLEARED, AND CLEARED BY A RESET.
(3)                                   ;:*  TEST 4
(6)                                   ;;*************************************************************
(5)  012524  000004                   TST4:  SCOPE
(3)  012526  012737  000004  001246          MOV     #4,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
(3)  012534  012737  012730  001362          MOV     #TST5,NEXT      ;POINT TO THE START OF THE NEXT TEST
(1)  012542  013700  002024                   MOV     DZVTCR,R0       ;SET DEVICE ADDRESS
(1)  012546  012703  012634                   MOV     #5$,R3          ;SET R3 POINTER TO TOP OF TABLE
(1)  012552  012737  012562  001364    1$:    MOV     #11$,LOCK       ;SET LOCK FOR SW09 SCOPE LOOP
(1)  012560  011305                           MOV     (R3),R5         ;SET EXPECTED RESULTS
(1)  012562  010510                   11$:    MOV     R5,(R0)         ;SET THE BIT
(1)  012564  011004                           MOV     (R0),R4         ;READ THE BIT FROM THE DEVICE
(1)  012566  020504                           CMP     R5,R4           ;DID THE BIT SET?
(1)  012570  001401                           BEQ     2$              ;BR IF YES
(1)  012572  104002                           ERROR   2               ;*BIT FAILED TO SET.
(1)  012574  104401                   2$:     SCOP1                   ;SW09 SET?
(1)  012576  012737  012604  001364           MOV     #3$,LOCK        ;SET UP FOR NEXT TIGHT SCOPE LOOP
(1)  012604  040510                   3$:     BIC     R5,(R0)         ;CLEAR THE BIT
(1)  012606  011004                           MOV     (R0),R4         ;READ THE REGISTER
(1)  012610  001403                           BEQ     4$              ;BR IF YES
(1)  012612  005005                           CLR     R5              ;SET EXPECTED TO 0
(1)  012614  104002                           ERROR   2               ;*REPORT BIT NOT CLEAR
(1)  012616  011305                           MOV     (R3),R5         ;RESTORE R5
(1)  012620  104401                   4$:     SCOP1                   ;SW09 SET?
(1)  012622  062703  000002                   ADD     #2,R3           ;POP POINTER TO NEXT TABLE ENTRY
(1)  012626  005713                           TST     (R3)            ;END OF TABLE?
(1)  012630  001412                           BEQ     6$              ;IF YES JUMP OVER TABLE
(1)  012632  000747                           BR      1$              ;START TESTING NEXT BIT
(1)  012634  000001                   5$:     #TCR0                   ;TCR BIT FOR LINE 0
(1)  012636  000002                           #TCR1                   ;TCR BIT FOR LINE 1
(1)  012640  000004                           #TCR2                   ;TCR BIT FOR LINE 2
(1)  012642  000010                           #TCR3                   ;TCR BIT FOR LINE 3
(1)  012644  000400                           #DTR0                   ;DTR BIT FOR LINE 0
(1)  012646  001000                           #DTR1                   ;DTR BIT FOR LINE 1
(1)  012650  002000                           #DTR2                   ;DTR BIT FOR LINE 2
(1)  012652  004000                           #DTR3                   ;DTR BIT FOR LINE 3
(1)  012654  000000                           #0                      ;END OF TABLE
(1)  012656  005037  001364           6$:     CLR     LOCK            ;CLEAR TIGHT SCOPE LOOP INDIC.
(1)  012662  012710  177777                   MOV     #-1,(R0)        ;SET ALL BITS IN TCR REGISTER
(1)  012666  012705  007400                   MOV     #007400,R5      ;SET EXPECTED
(1)  012672  104413                           DEVICE.CLR              ;SET DCLR BIT IN CSR
(1)  012674  011004                           MOV     (R0),R4         ;READ REGISTER
(1)  012676  020504                           CMP     R5,R4           ;TCR BITS CLEARED?
(1)  012700  001401                           BEQ     7$              ;IF YES BRANCH
(1)  012702  104002                           ERROR   2               ;TCR BITS NOT CLEARED!
(1)  012704  005005                   7$:     CLR     R5              ;SET EXPECTED TO ZERO
(1)  012706  005227  000000           8$:     INC     #0              ;DELAY FOR ACT
(1)  012712  001375                           BNE     8$              ;
(1)  012714  012710  177777                   MOV     #-1,(R0)        ;SET ALL POSSIBLE BITS
(1)  012720  000005                           RESET                   ;DO BUS INIT
```

```
(1)  012722  011004                        MOV     (R0),R4         ;DiD REGISTER CLEAR?
(1)  012724  001401                        BEQ     9$              ;IF YES GET OUT
(1)  012726  104002                        ERROR   2               ;REGISTER DID NOT CLEAR!
(1)  012730                          9$:
                                           ;************************* TEST 5 *********************************
3053                                       ;*THIS TEST VERIFIES THAT
(1)                                        ;*BITS 'RDONE,TRDY, BIT9, BIT8,
(1)                                        ;*AND SILOAL" ARE READ ONLY AND THAT TRDY IS
(1)                                        ;*ZERO UNTIL A LINE IS SELECTED AND MSENAB IS SET.
(1)                                        ;*
(3)                                  ;:*  TEST 5
(6)                                  ;:*************************************************************************
(5)  012730  000004                  TST5:  SCOPE
(3)  012732  012737  000005  001246         MOV     #5,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
(3)  012740  012737  013032  001362         MOV     #TST6,NEXT      ;POINT TO THE START OF THE NEXT TEST
(1)  012746  013700  002010                 MOV     DZVCSR,R0       ;SET ADDRESS TO R0
(1)  012752  104413                         DEVICE.CLR              ;DO A DEVICE CLEAR
(1)  012754  005005                         CLR     R5              ;SET EXPECTED TO 0
(1)  012756  012710  121600                 MOV     #RDONE+TRDY+BIT9+BIT8+SILOAL,(R0)
(1)                                                                 ;WRITE THE BITS
(1)  012762  011004                         MOV     (R0),R4         ;READ BACK THE BITS
(1)  012764  001401                         BEQ     2$              ;BR IF NONE ARE SET.
(1)  012766  104002                         ERROR   2               ;*BITS WERE SET.
(1)  012770  012705  100040          2$:    MOV     #TRDY+MSENAB,R5 ;SET EXPECTED BIT
(1)  012774  052777  000017  167022         BIS     #17,@DZVTCR     ;SET TCR BITS FOR ALL LINES
(1)  013002  052710  000040                 BIS     #MSENAB,(R0)    ;SET SCAN ENABLE
(1)  013006  005002                         CLR     R2              ;SET COUNTER TO ZERO
(1)  013010  011004                  3$:    MOV     (R0),R4         ;READ THE REGISTER
(1)  013012  042704  001400                 BIC     #BIT9!BIT8,R4   ;MASK OUT LINE NO.
(1)  013016  020504                         CMP     R5,R4           ;BIT SET?
(1)  013020  001404                         BEQ     4$              ;BR IF YES
(1)  013022  104414                         DELAY                   ;STALL TIME
(1)  013024  005202                         INC     R2              ;UPDATE COUNTER
(1)  013026  001370                         BNE     3$              ;BR IF COUNTER NOT DONE.
(1)  013030  104002                         ERROR   2               ;*TRDY NOT SET!
(1)  013032                          4$:
                                           ;************************* TEST 6 ********* **********************
3054                                       ;*THIS TEST VERIFIES THAT:
3055                                       ;*TIE,SILOEN,RIE,MSENAB,AND MAINT ARE THE
3056                                       ;*ONLY R/W BITS IN THE DZVCSR AND THAT
3057                                       ;*SETTING 'DCLR" IN THE CSR WILL CLEAR THESE BITS.
3058                                 ;:*  TEST 6
3060                                 ;:*************************************************************************
(5)                                  ;:
(4)  013032  000004                  TST6:  SCOPE
(2)  013034  012737  000006  001246         MOV     #6,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
(2)  013042  012737  013162  001362         MOV     #TST7,NEXT      ;POINT TO THE START OF THE NEXT TEST
3061 013050  104413                         DEVICE.CLR              ;SET DCLR IN CSR
3062 013052  013700  002010                 MOV     DZVCSR,R0       ;SET UP FOR ERROR MESSAGE
3063 013056  012710  177757                 MOV     #^C<DCLR>,(R0)  ;TRY TO SET ALL BITS EXCEPT DCLR
3064 013062  012705  050150                 MOV     #TIE!SILOEN!RIE!MSENAB!MAINT,R5 ;MAKE EXPECTED
3065 013066  011004                         MOV     (R0),R4         ;ACTUAL
3066 013070  020405                         CMP     R4,R5           ;CMP EXPECTED VS ACTUAL
3067 013072  001401                         BEQ     1$              ;YES
3068 013074  104002                         ERROR   2               ;*NO
3069 013076  105010                  1$:    CLRB    (R0)            ;CLEAR LOW BYTE OF CSR
3070 013100  105005                         CLRB    R5              ;CLEAR LOW BYTE OF EXPECTED DATA
```

```
3071  013102  011004                        MOV     (R0),R4         ;READ CSR
3072  013104  020405                        CMP     R4,R5           ;DOES CSR COMPARE WITH EXPECTED?
3073  013106  001401                        BEQ     3$              ;BRANCH IF YES
3074  013110  104002                        ERROR   2               ;IF NOT PRINT ERROR
3075  013112  012710  177757        3$:     MOV     #^C<DCLR>,(R0)  ;SET ALL CSR BITS POSSIBLE
3076  013116  105077  166670                CLRB    @HDZVCSR        ;CLEAR HIGH BYTE OF CSR
3077  013122  012705  000150                MOV     #RIE!MSENAB!MAINT,R5    ;SET EXPECTED IN R5
3078  013126  011004                        MOV     (R0),R4         ;READ CSR REGISTER
3079  013130  020405                        CMP     R4,R5           ;DOES ACTUAL=EXPECTED
3080  013132  001401                        BEQ     4$              ;IF YES CONTINUE
3081  013134  104002                        ERROR   2               ;IF NO PRINT ERROR
3082  013136  012710  177757        4$:     MOV     #^C<DCLR>,(R0)  ;SET ALL POSSIBLE CSR BITS
3083  013142  005005                        CLR     R5              ;SET R5 TO EXPECTED RESULTS
3084  013144  052710  000020                BIS     #DCLR,(R0)      ;DEVICE MASTER RESET
3085  013150  000240                        NOP
3086  013152  011004                        MOV     (R0),R4         ;ACTUAL
_087  013154  020405                        CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
3088  013156  001401                        BEQ     2$              ;YES
3089  013160  104002                        ERROR   2               ;*NO
3090  013162                          2$:
3091                                        ;************************* TEST 7 *********************************
 (1)                                        ;*THIS TEST PERFORMS RESET TESTING AND
 (1)                                        ;*TESTING OF READ ONLY REGISTER DZVRBUF
 (1)                                        ;*AND TESTING OF WRITE ONLY REGISTER DZVLPR
 (3)                                ;:*  TEST 7
 (6)                                ;;*************************************************************************
 (5)  013162  000004                TST7:   SCOPE
 (3)  013164  012737  000007  001246        MOV     #7,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
 (3)  013172  012737  013246  001362        MOV     #TST10,NEXT     ;POINT TO THE START OF THE NEXT TEST
 (1)  013200  104413                        DEVICE.CLR              ;CLEAR DZV11
 (1)  013202  013700  002014                MOV     DZVRBUF,R0      ;SET UP FOR ERROR MESSAGE
 (1)  013206  011005                        MOV     (R0),R5         ;COPY PRESENT CONTENTS
 (1)  013210  042705  106000                BIC     #DVALID!BIT11!BIT10,R5 ;CLEAR ILLEGAL BITS
 (1)  013214  012777  177777  166576        MOV     #-1,@DZVLPR     ;TRY TO WRITE ALL 1'S
 (1)  013222  011004                        MOV     (R0),R4         ;ACTUAL
 (1)  013224  020405                        CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
 (1)  013226  001401                        BEQ     1$              ;IF YES,GO CONTINUE PROCESSING
 (1)  013230  104002                        ERROR   2               ;*ERROR- BIT PATTERN NOT CORRECT
 (1)  013232  005077  166562        1$:     CLR     @DZVLPR ;TRY TO WRITE ALL ZEROES
 (1)  013236  011004                        MOV     (R0),R4         ;READ REGISTER
 (1)  013240  020405                        CMP     R4,R5           ;CMP ACTUAL VS. EXPECTED
 (1)  013242  001401                        BEQ     2$              ;BRANCH IF EQUAL
 (1)  013244  104002                        ERROR   2               ;VALUES DID NOT COMPARE
 (1)  013246                        2$:
3092                                        ;************************* TEST 10 ********************************
 (1)                                        ;*THIS TEST PERFORMS RESET TESTING AND
 (1)                                        ;*TESTING OF READ ONLY REGISTER DZVMSR
 (1)                                        ;*AND TESTING OF WRITE ONLY REGISTER DZVTDR
 (3)                                ;:*  TEST 10
 (6)                                ;;*************************************************************************
 (5)  013246  000004                TST10:  SCOPE
 (3)  013250  012737  000010  001246        MOV     #10,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
 (3)  013256  012737  013332  001362        MOV     #TST11,NEXT     ;POINT TO THE START OF THE NEXT TEST
 (1)  013264  104413                        DEVICE.CLR              ;CLEAR DZV11
 (1)  013266  013700  002030                MOV     DZVMSR,R0       ;SET UP FOR ERROR MESSAGE
 (1)  013272  011005                        MOV     (R0),R5         ;COPY PRESENT CONTENTS
```

```
  (1) 013274 042705 170360                BIC     #170360,R5      ;CLEAR ILLEGAL BITS
  (1) 013300 112777 177777 166526         MOVB    #-1,@DZVTDR     ;TRY TO WRITE ALL 1'S
  (1) 013306 011004                       MOV     (R0),R4         ;ACTUAL
  (1) 013310 020405                       CMP     R4,R5           ;CMP ACTUAL VS EXPECTED
  (1) 013312 001401                       BEQ     1$              ;IF YES,GO CONTINUE PROCESSING
  (1) 013314 104002                       ERROR   2               ;*ERROR- BIT PATTERN NOT CORRECT
  (1) 013316 005077 166512          1$:   CLR     @DZVTDR ;TRY TO WRITE ALL ZEROES
  (1) 013322 011004                       MOV     (R0),R4         ;READ REGISTER
  (1) 013324 020405                       CMP     R4,R5           ;CMP ACTUAL VS. EXPECTED
  (1) 013326 001401                       BEQ     2$              ;BRANCH IF EQUAL
  (1) 013330 104002                       ERROR   2               ;VALUES DID NOT COMPARE
  (1) 013332                        2$:
 3093
 3094                                     ;************************* TEST 11 *********************************
 3095                                     ;*VERIFY THAT SETTING "DTR" FOR A LINE WILL
 3096                                     ;*BRING UP "CO" AND "RING" FOR:
 3097                                     ;*THE SAME LINE IF IN EXTERNAL MODE
 3098                                     ;*THE STAGGERED LINE IF IN STAGGERED MODE.
 3099                                     ;*LINES ARE STAGGERED AS FOLLOWS:
 3100                                     ;*LINE0 WITH LINE1; LINE2 WITH LINE3.
 3101                                     ;*THIS TEST IS ONLY RUN IF AN H325,OR H329
 3102                                     ;*IS CONNECTED ON THE DZV UNDER TEST.
 3104
 3105                                     ;:*  TEST 11
  (5)                                     ;:***************************************************************
  (4) 013332 000004                TST11: SCOPE
  (2) 013334 012737 000011 001246         MOV     #11,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
  (2) 013342 012737 013526 001362         MOV     #TST12,NEXT     ;POINT TO THE START OF THE NEXT TEST
 3106 013350 005737 001372                TST     MODE            ;TEST TO SEE IF TESTING WITH
 3107 013354 001001                       BNE     8$              ;CONNECTOR
 3108 013356 104400                       ADVANCE                 ;IF NO, GO TO NEXT TEST
 3109 013360 012737 013450 001364   8$:   MOV     #10$,LOCK       ;SET FOR TIGHT SCOPE LOOP
 3110 013366 104413                       DEVICE.CLR              ;SET DCLR IN CSR TO ZERO DEVICE
 3111 013370 013700 002030                MOV     DZVMSR,R0       ;SET REGISTER
 3112 013374 005003                       CLR     R3              ;ZERO LINE NUMBER
 3113 013376 012702 000001                MOV     #1,R2           ;SET POINTER
 3114 013402 130237 001366          1$:   BITB    R2,LINE         ;TEST THIS LINE?
 3115 013406 001003                       BNE     3$              ;YES
 3116 013410 005203                 2$:   INC     R3              ;LINE #
 3117 013412 104420                       SHIFT                   ;GET NEXT LINE
 3118 013414 000772                       BR      1$              ;TEST NEXT LINE
 3119 013416 010204                 3$:   MOV     R2,R4           ;SAVE BINARY BIT FOR LINE #
 3120 013420 105737 001372                TSTB    MODE            ;RUNNING IN EXTERNAL MODE?
 3121 013424 100406                       BMI     5$              ;IF YES SKIP STAGGERED SETUP
 3122 013426 032703 000001                BIT     #BIT0,R3        ;IF EVEN LINE
 3123 013432 001402                       BEQ     4$              ;GO GET ODD PARTNER
 3124 013434 006204                       ASR     R4              ;OTHERWISE GET EVEN COMPANION
 3125 013436 000401                       BR      5$              ;GO SETUP EXPECTED RESULTS
 3126 013440 006304                 4$:   ASL     R4              ;FIND ODD PARTNER
 3127 013442 010405                 5$:   MOV     R4,R5           ;LOAD R5 FOR EXPECTED
 3128 013444 000305                       SWAB    R5              ;PLACE IN UPPER BYTE
 3129 013446 150405                       BISB    R4,R5           ;SET FOR RING BITS
 3130 013450 150277 166352          10$:  BISB    R2,@HDZVTCR     ;SET DTR BIT
 3131 013454 104414                       DELAY                   ;DELAY FOR CABLE LAG
 3132 013456 011004                       MOV     (R0),R4         ;MOVE RESULTS OF MSR REGISTER TO R4
 3133 013460 020504                       CMP     R5,R4           ;RESULTS=EXPECTED?
```

```
 3134  013462  001401                          BEQ     6$              ;IF YES CONTINUE
 3135  013464  104002                          ERROR   2               ;IF NOT PRINT ERROR RESULTS
 3136  013466  104401                  6$:     SCOP1                   ;IS SW09 SET?
 3137  013470  012737  013476  001364          MOV     #11$,LOCK       ;SET UP FOR NEXT TIGHT SCOPE
 3138  013476  140277  166324          11$:    BICB    R2,@HDZVTCR     ;CLEAR  DTR BIT FOR LINE UNDER TEST
 3139  013502  104414                          DELAY                   ;DELAY FOR CABLE LAG
 3140  013504  011004                          MOV     (R0),R4         ;LOAD MSR REGISTER INTO R4
 3141  013506  001402                          BEQ     7$              ;IF CO AND RING CLEARED CONTINUE
 3142  013510  005005                          CLR     R5              ;OTHERWISE SET EXPECTED FOR ERROR
 3143  013512  104002                          ERROR   2               ;PRINTOUT
 3144  013514  104401                  7$:     SCOP1                   ;IS SW09 SET?
 3145  013516  012737  013450  001364          MOV     #10$,LOCK       ;RESET TIGHT SCOPE LOOP
 3146  013524  000731                          BR      2$              ;GET NEXT LINE
 3147
 3148                                          ;************************* TEST 12 *********************************
   (1)                                         ;* THIS TEST VERIFIES THAT TRDY IS SET WHEN A LINE
   (1)                                         ;* IS READY TO BE LOADED, AND THAT THE LINE SPECI-
   (1)                                         ;* FIED IN BITS 8-9 OF DZVCSR CORRESPOND
   (1)                                         ;* TO THE LINE SELECTED IN DZVTCR
   (3)                                  ;:*  TEST 12
   (6)                                  ;:*************************************************************************
   (5)  013526  000004                  TST12:  SCOPE
   (3)  013530  012737  000012  001246          MOV     #12,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
   (3)  013536  012737  013660  001362          MOV     #TST13,NEXT     ;POINT TO THE START OF THE NEXT TEST
   (1)  013544  104413                          DEVICE.CLR              ;ISSUE A 'DEVICE CLEAR" (RESET)
   (1)  013546  012737  013602  001364          MOV     #2$,LOCK        ;SET UP FOR TIGHT SCOPE LOOP
   (1)  013554  005037  001374                  CLR     SAVLIN          ;INITIALIZE FOR ERROR PRINTOUT
   (1)  013560  013700  002010                  MOV     DZVCSR,R0       ;SET POINTER
   (1)  013564  012705  100040                  MOV     #MSENAB!TRDY,R5 ;START THE EXPECTED LINE NUMBER AT 0
   (1)  013570  012702  000001                  MOV      #1,R2          ;USING R2 AS A BIT POINTER, POINT TO LINE 0
   (1)  013574  130237  001366          1$:     BITB    R2,LINE         ;IS THIS LINE SELECTED?
   (1)  013600  001421                          BEQ     6$              ;IF NO, SKIP THE STARTUP
   (1)  013602  050277  166216          2$:     BIS     R2,@DZVTCR      ;SET THE GO BIT FOR THIS LINE
   (1)  013606  052710  000040                  BIS     #MSENAB,(R0)    ;START THE SCANNER
   (1)  013612  005004                          CLR     R4              ;SET FOR DELAY
   (1)  013614  005710                  3$:     TST     (R0)            ;TX READY?
   (1)  013616  100404                          BMI     4$              ;BR IF YES
   (1)  013620  104414                          DELAY                   ;DELAY
   (1)  013622  005204                          INC     R4              ;COUNTER
   (1)  013624  001373                          BNE     3$              ;BR IF <>0!
   (1)  013626  104003                          ERROR   3               ;*TX NOT READY!
   (1)  013630  011004                  4$:     MOV     (R0),R4         ;GET THE LINE POINTED TO BY THE SCANNER
   (1)  013632  020405                          CMP     R4,R5           ;IS THE LINE NUMBER WHAT IT SHOULD BE?
   (1)  013634  001401                          BEQ     5$              ;IF YES,GO WORK ON THE NEXT LINE
   (1)  013636  104002                          ERROR   2               ;*LINE NUMBER DID NOT MATCH TCR BIT
   (1)  013640  104401                  5$:     SCOP1                   ;IS SW09 SET?
   (1)  013642  104413                          DEVICE.CLR              ;SET DCLR IN CSR;SETUP FOR NEXT LINE
   (1)  013644  062705  000400          6$:     ADD     #400,R5         ;POINT TO THE NEXT EXPECTED LINE
   (1)  013650  104420                          SHIFT                   ;POINT TO THE NEXT LINE.ARE ALL LINES TESTED?
   (1)  013652  005237  001374                  INC     SAVLIN          ;ADJUST FOR ERROR PRINTOUT
   (1)  013656  000746                          BR      1$              ;IF NOT, GO DO THE NEXT LINE
 3149                                           ;************************* TEST 13 *********************************
 3150                                           ;*TEST TO TRANSMIT ONE CHAR AND
 3151                                           ;*RECEIVE ONE CHAR ON ONE LINE
 3152                                           ;*AT A TIME. THE CHAR IS '252" AND
 3153                                           ;*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
                                ;:*****************************************************************
013660  000004                 TST13:  SCOPE
013662  012737  000013  001246         MOV     #13,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
013670  012737  014150  001362         MOV     #TST14,NEXT     ;POINT TO THE START OF THE NEXT TEST
013676  012737  014132  001364         MOV     #16$,LOCK       ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
013704  104417                       DCLASM                  ;SET DCLR IN CSR AND SET MAINT MODE
013706  104421                       LPRSET                  ;LOAD LPR REGISTER FOR ALL LINES
013710  005037  001374               CLR     SAVLIN          ;INIT. FOR ERROR PRINTOUT
013714  105037  001425               CLRB    DONFLG          ;INIT FOR TCR BIT HANDLER
013720  012702  000001               MOV     #1,R2           ;LINE POINTER
013724  012701  000252               MOV     #252,R1         ;SAVE CHARACTER TO BE TRANSMITTED
013730  052777  000040  166052       BIS     #MSENAB,@DZVCSR ;START SCANNER
013736  030237  001366         3$:   BIT     R2,LINE         ;VALID LINE ?
013742  001467                       BEQ     15$             ;NO SET UP NEXT LINE
013744  010277  166054               MOV     R2,@DZVTCR      ;SET TCR BIT
013750  005005                 5$:   CLR     R5              ;SET R5 FOR A DELAY LOOP
013752  105777  166032               TSTB    @DZVCSR         ;IS REC DONE = 0 ?
013756  100001                       BPL     6$              ;IF YES, ALLOW TIME FOR TRDY TO SET
013760  104020                       ERROR   20              ;*REC DONE SHOULD = 0
013762  005777  166022         6$:   TST     @DZVCSR         ;TRDY SET?
013766  100404                       BMI     7$              ;IF YES BRANCH
013770  104414                       DELAY                   ;IF NO THEN WAIT FOR IT
013772  005205                       INC     R5              ;DELAY LOOP
013774  001372                       BNE     6$              ;BRANCH BACK AND TEST AGAIN
013776  104003                       ERROR   3               ;*TRDY FAILED TO SET!
014000  105737  001425         7$:   TSTB    DONFLG          ;HAVE WE ALREADY SENT CHARAC.
014004  001041                       BNE     13$             ;IF YES GO CLEAR TCR BIT
014006  105237  001425               INCB    DONFLG          ;IF NOT INDICATE HAVING BEEN HERE
0 4012  110177  166016               MOVB    R1,@DZVTDR      ;LOAD CHARACTER
014016  013705  001374               MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
014022  005737  001372               TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
014026  100006                       BPL     10$             ;IF NOT, SKIP STAGGERED SETUP
                                (1)
                                (1)     ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
                                (1)
014030  006205                 (1)   ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
014032  103402                 (1)   BCS     8$              ;IF IT IS SET, GO CLEAR IT
014034  000261                 (1)   SEC                     ;IF IT IS CLEAR SET IT HERE
014036  000401                 (1)   BR      9$              ;SKIP THE CLEARING
014040  000241                 (1)8$: CLC                    ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
014042  006105                 (1)9$: ROL     R5              ;GET THE NEW BIT BACK INTO R5
014044  000305                 10$:  SWAB    R5              ;MOVE THE LINE NUMBER TO THE UPPER BYTE
014046  150105                       BISB    R1,R5           ;ADD CHARACTER
014050  052705  100000               BIS     #DVALID,R5      ;ADD DATA VALID
014054  005003                       CLR     R3
014056  105777  165726         11$:  TSTB    @DZVCSR         ;IS RDONE SET?
014062  100404                       BMI     12$             ;IF YES GO GET CHAR.
014064  104414                       DELAY                   ;IF NOT THEN WAIT
014066  005203                       INC     R3              ;DELAY LOOP
014070  001372                       BNE     11$             ;DELAY DONE?
014072  104004                       ERROR   4               ;*RDONE FAILED TO SET!
```

```
 3196  014074  017704  165714          12$:    MOV     @DZVRBUF,R4     ;LOAD THE VALUE ACTUALLY RECEIVED
 3197  014100  020405                          CMP     R4,R5           ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
 3198  014102  001722                          BEQ     5$              ;IF YES, GO DO THE NEXT LINE
 3199  014104  104006                          ERROR   6               ;*NO DATA/CONTENTS DID NOT COMPARE
 3200  014106  000720                          BR      5$              ;GO BACK AND WAIT TO CLEAR TCR BIT
 3201  014110  104401                  13$:    SCOP1                   ;CHECK TO SEE IF SWITCH NINE IS SET
 3202  014112  105037  001425                  CLRB    DONFLG          ;SET UP FOR NEXT LINE
 3203  014116  005077  165702                  CLR     @DZVTCR         ;CLEAR PREVIOUS TCR BIT
 3204  014122  005237  001374          15$:    INC     SAVLIN          ;SET LINE INDICATOR FOR NEXT LINE
 3205  014126  104420                          SHIFT                   ;CALCULATE NEXT LINE
 3206  014130  000702                          BR      3$              ;GET GET STARTED
 3207
 3208                                  ;TIGHT SCOPE LOOP FOR THIS TEST. LOOP TRANSMITS CHARACTERS ONLY
 3209
 3210  014132  005777  165652          16$:    TST     @DZVCSR         ;IS TRANSMITTER READY?
 3211  014136  100375                          BPL     16$             ;IF NOT, WAIT FOR IT
 3212  014140  110177  165670                  MOVB    R1,@DZVTDR      ;LOAD THE CHARACTER
 3213  014144  104401                          SCOP1                   ;LOOP AGIN IF SW09=1
 3214  014146  000760                          BR      13$             ;OTHERWISE, GO PICK UP THE TEST NORMALLY
 3215
 3216                                          ;************************* TEST 14 *********************************
 3217                                          ;*THIS TEST VERIFIES THAT EACH RECEIVING LINE CAN BE
 3218                                          ;*DISABLED BY SETTING RCVON (BIT12 IN THE LPR REGISTER)
 3219                                          ;*TO ZERO FOR EACH LINE.
 3220                                          ;*THIS TEST ALSO VERIFIES THAT THE SILO CAN BE
 3221                                          ;*EMPTIED BY ISSUING A DEVICE MASTER CLEAR.
 3223                                  ;:*  TEST 14
  (5)                                  ;:**************************************************************
  (4)  014150  000004                  TST14:  SCOPE
  (2)  014152  012737  000014  001246          MOV     #14,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
  (2)  014160  012737  014472  001362          MOV     #TST15,NEXT     ;POINT TO THE START OF THE NEXT TEST
 3224  014166  105037  001425                  CLRB    DONFLG          ;CLEAR TEST CONTROL FLAG
 3225  014172  005037  001374                  CLR     SAVLIN          ;CLEAR LINE INDICATOR
 3226  014176  104417                          DCLASM                  ;ISSUE A DEVICE MASTER CLEAR
 3227                                                                  ;AND SET MAINT BIT IF NECESSARY
 3228  014200  013701  001370                  MOV     PAR,R1          ;SAVE DEFAULT PARAMETERS
 3229  014204  042737  010000  001370          BIC     #RCVON,PAR      ;DISABLE RECEIVER IN DEFAULT PAR.
 3230  014212  104421                  100$:   LPRSET                  ;LOAD PARAMETERS IN LPR REGISTER
 3231  014214  010137  001370                  MOV     R1,PAR          ;RESTORE DEFAULT PARAMETERS
 3232  014220  012701  000252                  MOV     #252,R1         ;LOAD A CHARAC. INTO R1
 3233  014224  013702  001366                  MOV     LINE,R2         ;COPY AN IMAGE OF THE ACTIVE LINES
 3234  014230  010277  165570                  MOV     R2,@DZVTCR      ;SET TCR BITS FOR ALL ACTIVE LINES
 3235  014234  052777  000040  165546          BIS     #MSENAB,@DZVCSR ;SET MASTER SCAN ENABLE
 3236  014242  005005                  1$:     CLR     R5              ;INIT DELAY COUNTER
 3237  014244  005777  165540          2$:     TST     @DZVCSR         ;IS TRANS READY SET?
 3238  014250  100404                          BMI     3$              ;BRANCH IF YES
 3239  014252  104414                          DELAY                   ;WAIT FOR TRDY TO SET
 3240  014254  005205                          INC     R5              ;INCREMENT DELAY COUNTER
 3241  014256  001372                          BNE     2$              ;RETURN TO CHECK TRDY
 3242  014260  104003                          ERROR   3               ;TRDY FAILED TO SET!
 3243  014262  117705  165524          3$:     MOVB    @HDZVCSR,R5     ;MOVE LINE NO. TO R5
 3244  014266  012703  000001                  MOV     #1,R3           ;INIT TCR POINTER
 3245  014272  042705  177774                  BIC     #^C<3>,R5       ;ISOLATE LINE NO.
 3246  014276  001403                          BEQ     31$             ;IF LINE 0 BRANCH
 3247  014300  106303                  30$:    ASLB    R3              ;SHIFT R3 POINTER TO NEXT LINE
 3248  014302  005305                          DEC     R5              ;DECREMENT LINE NO.
```

```
3249  014304  001375                  BNE     30$             ;WHEN R5=0, R3 POINTS TO LINE TCR
3250  014306  030302          31$:    BIT     R3,R2           ;HAS CHARACTER BEEN SENT
3251  014310  001007                  BNE     4$              ;BRANCH IF NO
3252  014312  140377  165506          BICB    R3,@DZVTCR      ;IF YES THEN CLEAR TCR BIT
3253  014316  001351                  BNE     1$              ;IF ALL CHARAC. SENT DROP THROUGH
3254  014320  105737  001425          TSTB    DONFLG          ;IF NO MORE ACTIVE IS THIS SECOND
3255                                                          ;TIME HERE?
3256  014324  001037                  BNE     10$             ;IF YES SKIP TO SECOND PART OF TEST
3257  014326  000404                  BR      5$              ;IF FIRST TIME HERE GO ZERO TCR BITS
3258  014330  110177  165500    4$:   MOVB    R1,@DZVTDR      ;LOAD CHAR. INTO BUFFER
3259  014334  040302                  BIC     R3,R2           ;INDICATE CHARAC. SENT ON THIS LINE
3260  014336  000741                  BR      1$              ;GO BACK AND WAIT FOR TRDY TO SET
3261  014340  005077  165460    5$:   CLR     @DZVTCR         ;CLEAR OUT TCR BITS
3262  014344  005005                  CLR     R5              ;INIT DELAY COUNTER
3263  014346  105777  165436    6$:   TSTB    @DZVCSR         ;IS RECEIV. DONE SET?
3264  014352  100002                  BPL     7$              ;IF NOT THEN WAIT TO SEE IF IT WILL
3265  014354  104020                  ERROR   20              ;REC DONE SHOULD NOT SET!
3266  014356  000403                  BR      8$              ;GO FIND WHICH LINE RECEIVED
3267  014360  104414          7$:     DELAY                   ;STALL FOR RECEIVER
3268  014362  005205                  INC     R5              ;INCREMENT DELAY COUNTER
3269  014364  001370                  BNE     6$              ;IF NOT DONE GO RETEST REC DONE
3270  014366  017704  165422    8$:   MOV     @DZVRBUF,R4     ;READ REC. BUFFER
3271  014372  100007                  BPL     9$              ;IS DVALID SET?
3272  014374  000304                  SWAB    R4              ;IF YES GET LINE NO.
3273  014376  042704  177774          BIC     #^C<3>,R4       ;ISOLATE LINE NO.
3274  014402  010437  001374          MOV     R4,SAVLIN       ;SET UP LINE NO. FOR ERROR REPORT
3275  014406  104017                  ERROR   17              ;DVALID SHOULD NOT BE SET
3276  014410  000766                  BR      8$              ;GO CHECK FOR ANY OTHER CHAR. IN SILO
3277  014412  105237  001425    9$:   INCB    DONFLG          ;INDICATE THAT FIRST PART OF TEST IS DONE
3278  014416  013701  001370          MOV     PAR,R1          ;SAVE DEFAULT LINE PARAM.
3279  014422  000673                  BR      100$            ;NOW GO RELOAD LPR REGISTER TO
3280                                                          ;TURN RECEIVERS ON
3281  014424  005005          10$:    CLR     R5              ;ZERO DELAY COUNTER
3282  014426  104414          11$:    DELAY                   ;WAIT FOR ALL CHARAC. TO BE RECEIVED
3283  014430  005205                  INC     R5              ;INCREASE DELAY COUNT
3284  014432  001375                  BNE     11$             ;CONT. DELAY IF NOT FINISHED
3285  014434  104413                  DEVICE.CLR              ;ISSUE A MASTER CLEAR
3286  014436  000240                  NOP
3287  014440  000240                  NOP
3288  014442  105777  165342          TSTB    @DZVCSR         ;NOW IS RECEIV. DONE SET?
3289  014446  100003                  BPL     12$             ;BRANCH IF NO
3290  014450  005037  001374          CLR     SAVLIN          ;CLEAR LINE NO FOR ERROR REPORT
3291  014454  104020                  ERROR   20              ;REC. DONE SHOULD NOT BE SET!
3292  014456  017704  165332    12$:  MOV     @DZVRBUF,R4     ;READ REC. BUFFER
3293  014462  100003                  BPL     13$             ;IS DVALID SET? IT SHOULDN'T BE
3294  014464  005037  001374          CLR     SAVLIN          ;DEVICE. CLR DID NOT ZERO SILO
3295  014470  104017                  ERROR   17              ;PRINT OUT THE ERROR.(LINE NO. IS IRRELEVANT)
3296  014472                  13$:
3297
3298                                  ;************************* TEST 15 *********************************
3299                                  ;* THIS TEST PROVES THAT THE TRANSMITTER TRANSMITS
3300                                  ;*CHARACTERS (FLAG MODE)AND THE RECEIVER RECEIVES (FLAG MODE)
3301                                  ;*(ONE LINE AT A TIME   BASED UPON VALID LINES)
3302                                  ;*THIS IS THE FIRST TIME THAT ALL DATA IS CHECKED
3304                          ;:*  TEST 15
 (5)                      ;;*************************************************************
```

```
   (4)  014472  000004                        TST15:  SCOPE
   (2)  014474  012737  000015  001246                MOV     #15,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
   (2)  014502  012737  014762  001362                MOV     #TST16,NEXT     ;POINT TO THE START OF THE NEXT TEST
   (1)  014510  012737  014576  001364                MOV     #5$,LOCK        ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
 3305   014516  104417                                DCLASM                  ;SET DCLR AND SET MNTFLG
 3306   014520  104421                                LPRSET                  ;LOAD LPR REGISTER FOR ALL LINES
 3307   014522  005037  001374                        CLR     SAVLIN          ;INIT FOR FIRST LINE
 3308   014526  104422                                BUFSET                  ;ZERO BUFFER AREA
 3309   014530  105037  001425                        CLRB    DONFLG          ;ZERO TCR BIT HANDLER FLAG
 3310   014534  012702  000001                        MOV     #1,R2           ;LINE POINTER
 3311   014540  052777  000040  165242                BIS     #MSENAB,@DZVCSR ;START SCANNER
 3312   014546  030237  001366          3$:   BIT     R2,LINE         ;VALID LINE ?
 3313   014552  001477                                BEQ     15$             ;NO SET UP NEXT LINE
 3314   014554  010277  165244                        MOV     R2,@DZVTCR      ;SET TCR BIT
 3315   014560  013700  001374                        MOV     SAVLIN,R0       ;ADJUST BUFFER POINTER
 3316   014564  006300                                ASL     R0              ;OFFSET
 3317   014566  105777  165216          4$:   TSTB    @DZVCSR         ;IS REC DONE = 0 ?
 3318   014572  100001                                BPL     5$              ;IF YES, ALLOW TIME FOR TRDY TO SET
 3319   014574  104020                                ERROR   20              ;*REC DONE SHOULD = 0
 3320   014576  005005          5$:   CLR     R5              ;USE R5 AS TIMER WAITING FOR TRDY TO SET
 3321   014600  005777  165204          6$:   TST     @DZVCSR         ;IS THE TRANSMITTER READY?
 3322   014604  100404                                BMI     7$              ;IF SO, GO TRANSMIT A CHARACTER
 3323   014606  104414                                DELAY                   ;WAIT A LITTLE BIT
 3324   014610  005205                                INC     R5              ;UP THE LOCAL COUNTER.TIME EXCEEDED?
 3325   014612  001372                                BNE     6$              ;IF NOT, GO TRY AGAIN
 3326   014614  104003                                ERROR   3               ;*TRDY FAILED TO SET!
 3327   014616  105737  001425          7$:   TSTB    DONFLG          ;ALL CHARAC. TRANS.?
 3328   014622  001047                                BNE     14$             ;IF YES GO ZERO TCR BIT
 3329   014624  116077  001426  165202                MOVB    TDO(R0),@DZVTDR ;LOAD CHARACTER
 3330   014632  013705  001374                        MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
 3331   014636  005737  001372                        TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
   (1)  014642  100006                                BPL     10$             ;IF NOT, SKIP STAGGERED SETUP
   (1)
   (1)                                                ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
   (1)
   (1)  014644  006205                                ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
   (1)  014646  103402                                BCS     8$              ;IF IT IS SET, GO CLEAR IT
   (1)  014650  000261                                SEC                     ;IF IT IS CLEAR SET IT HERE
   (1)  014652  000401                                BR      9$              ;SKIP THE CLEARING
   (1)  014654  000241          8$:   CLC             ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
   (1)  014656  006105          9$:   ROL     R5              ;GET THE NEW BIT BACK INTO R5
 3332   014660  000305          10$:  SWAB    R5              ;MOVE THE LINE NUMBER TO THE UPPER BYTE
 3333   014662  156005  001426                        BISB    TDO(R0),R5      ;ADD CHARACTER
 3334   014666  052705  100000                        BIS     #DVALID,R5      ;ADD DATA VALID
 3335   014672  005003                                CLR     R3
 3336   014674  105777  165110          11$:  TSTB    @DZVCSR         ;REC DONE?
 3337   014700  100404                                BMI     12$             ;IF YES GO CHECK CHAR.
 3338   014702  104414                                DELAY                   ;IF NOT WAIT FOR REC.
 3339   014704  005203                                INC     R3              ;DELAY LOOP TIMER
 3340   014706  001372                                BNE     11$             ;DELAY FINISHED?
 3341   014710  104004                                ERROR   4               ;*RDONE FAILED TO SET!
 3342   014712  017704  165076          12$:  MOV     @DZVRBUF,R4     ;LOAD THE VALUE ACTUALLY RECEIVED
 3343   014716  020405                                CMP     R4,R5           ;COMPARE ACTUAL VS EXPECTED. ARE THEY THE SAME?
 3344   014720  001401                                BEQ     13$             ;IF YES, GO DO THE NEXT LINE
 3345   014722  104006                                ERROR   6               ;*NO DATA/CONTENTS DID NOT COMPARE
 3346   014724  104401          13$:  SCOP1           ;CHECK TO SEE IF SWITCH NINE IS SET
```

```
3347  014726  105260  001426                    INCB    TDO(R0)          ;INCREMENT BINARY PATTERN FOR THIS LINE
3348  014732  001315                            BNE     4$               ;GO 'ROUND AGAIN FOR NEXT CHARACTER
3349  014734  105237  001425                    INCB    DONFLG           ;INDICATE ALL CHAR. SENT
3350  014740  000712                            BR      4$               ;BRANCH TO CLEAR TCR BIT
3351  014742  005077  165056          14$:      CLR     @DZVTCR          ;CLEAR TCR REGISTER
3352  014746  105037  001425                    CLRB    DONFLG           ;INIT FOR NEXT LINE
3353  014752  005237  001374          15$:      INC     SAVLIN           ;INC EXPECTED LINE
3354  014756  104420                            SHIFT                    ;SHIFT THE LINE POINTER. ARE WE ALL DONE?
3355  014760  000672                            BR      3$               ;IF NO, GO AROUND AGAIN FOR NEXT LINE
3356
3357
3358                                            ;************************* TEST 16 *********************************
3359                                            ;*THIS TEST WILL PROVE THAT:
3360                                            ;* 1) THE TRANSMITTER 'BREAK BIT" WORKS
3361                                            ;* 2) THE RECEIVER CAN FLAG 'FRAMING ERRORS"
3362                                            ;* 3) THE RECEIVER CAN FLAG 'PARITY ERRORS"
3363                                            ;*ONLY ONE LINE AT A TIME WILL BE EXERCISED.
3365                                    ;:*  TEST 16
 (5)                                    ;;*****************************************************************
 (4)  014762  000004                    TST16:  SCOPE
 (2)  014764  012737  000016  001246            MOV     #16,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
 (2)  014772  012737  015164  001362            MOV     #TST17,NEXT      ;POINT TO THE START OF THE NEXT TEST
3366  015000  012737  015110  001364            MOV     #5$,LOCK         ;SET FOR LOOP
3367  015006  005037  001374                    CLR     SAVLIN           ;INIT LINE INDIC. FOR ERROR PRINTOUT
3368  015012  012702  000001                    MOV     #1,R2            ;LINE POINTER
3369  015016  030237  001366          1$:       BIT     R2,LINE          ;VALID LINE?
3370  015022  001454                            BEQ     9$               ;IF NOT SET FOR NEXT LINE
3371  015024  104417                            DCLASM                   ;SET DCLR IN CSR AND SET MNTFLG
3372  015026  013701  001370                    MOV     PAR,R1           ;PICK UP PARAMETERS
3373  015032  052737  000300  001370            BIS     #ODDPAR!PARITY,PAR ;FORCE ODD PARITY
3374  015040  104421                            LPRSET                   ;LOAD LPR REGISTER
3375  015042  010137  001370                    MOV     R1,PAR           ;RESET PAR TO ORIGINAL VALUE
3376  015046  052777  000040  164734            BIS     #MSENAB,@DZVCSR  ;START SCANNER
3377  015054  013705  001374                    MOV     SAVLIN,R5        ;MAKE EXPECTED DATA
3378  015060  005737  001372                    TST     MODE             ;IS THIS TEST IN STAGGERED MODE?
 (1)  015064  100006                            BPL     4$               ;IF NOT, SKIP STAGGERED SETUP
 (1)
 (1)                                            ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
 (1)
 (1)  015066  006205                            ASR     R5               ;GET THE LAST BIT INTO THE CARRY BIT
 (1)  015070  103402                            BCS     2$               ;IF IT IS SET, GO CLEAR IT
 (1)  015072  000261                            SEC                      ;IF IT IS CLEAR SET IT HERE
 (1)  015074  000401                            BR      3$               ;SKIP THE CLEARING
 (1)  015076  000241                    2$:     CLC                      ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
 (1)  015100  006105                    3$:     ROL     R5               ;GET THE NEW BIT BACK INTO R5
3379  015102  000305                    4$:     SWAB    R5               ;PUT LINE NUMBER IN UPPER BYTE
3380  015104  052705  130000                    BIS     #DVALID!PARER!FRMERR,R5 ;ADD EXPECTED
3381  015110  005003                    5$:     CLR     R3               ;INIT DELAY ACCUMULATOR
3382  015112  110277  164720                    MOVB    R2,@HDZVTDR      ;SET BREAK BIT
3383  015116  105777  164666          6$:       TSTB    @DZVCSR          ;RECEIVER DONE?
3384  015122  100404                            BMI     7$               ;BRANCH IF YES
3385  015124  104414                            DELAY                    ;WAIT FOR REC DONE TO SET
3386  015126  005203                            INC     R3               ;INC DELAY LOOP
3387  015130  001372                            BNE     6$               ;DELAY FINISHED?
3388  015132  104004                            ERROR   4                ;*RDONE FAILED TO SET!
3389  015134  017704  164654          7$:       MOV     @DZVRBUF,R4      ;ACTUAL
```

```
3390  015140  020405                          CMP     R4,R5           ;CMP ACTUAL VS EXPECTED. DO THEY MATCH?
3391  015142  001401                          BEQ     8$              ;IF YES, GO CLEAN UP
3392  015144  104006                          ERROR   6               ;*DATA/CONTENTS FAILED TO COMPARE
3393  015146  105077  164664          8$:     CLRB    @#DZVTDR        ;CLEAR BREAK BITS
3394  015152  1C4401                          SCOP1                   ;LOOP?
3395  015154  005237  001374          9$:     INC     SAVLIN          ;INC LINE #
3396  015160  104420                          SHIFT                   ;SET R2 TO NEXT LINE
3397  015162  000715                          BR      1$              ;GO BACK AND TEST NEXT LINE
3398                                          ;************************ TEST 17 *********************************
 (1)                                        ;* THIS TEST VERIFIES THAT THE DEVICE DOES NOT INTERRUPT
 (1)                                        ;*WHILE THE PROCESSOR STATUS DOES NOT ALLOW INTERRUPTS
 (1)                                        ;*BUT WILL INTERRUPT IF THE PROCESSOR STATUS
 (1)                                        ;*ALLOWS INTERRUPTS.
 (3)                                    ;::*  TEST 17
 (6)                                    ;::*************************************************************************
 (5)  015164  000004                    TST17:  SCOPE
 (3)  015166  012737  000017  001246            MOV     #17,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
 (3)  015174  012737  015560  001362            MOV     #TST20,NEXT     ;POINT TO THE START OF THE NEXT TEST
 (1)  015202  104417                            DCLASM                  ;SET DCLR IN CSR AND SET MAINT BIT
 (1)                                            ;IF NECESSARY (INTERNAL MODE)
 (1)  015204  104421                            LPRSET                  ;SET UP LPR REGISTER
 (1)  015206  005037  001374                    CLR     SAVLIN          ;INIT LINE INDIC. FOR ERROR
 (1)  015212  105037  001425                    CLRB    DONFLG          ;INIT TCR BIT HANDLER FLAG
 (1)  015216  113777  001366  164600            MOVB    LINE,@#DZVTCR   ;SET ALL VALID TCR BITS
 (1)  015224  106427  000200                    MTPS    #MASK           ;SET CPU STATUS TO DZV11 PRIO,
 (1)  015230  012777  000200  164604            MOV     #MASK,@#DZVRIS  ;SET RECEIVER STATUS
 (1)  015236  012777  000200  164602            MOV     #MASK,@#DZVTIS  ;SET TRANSMITTER STATUS
 (1)  015244                            1$:
 (2)  015244  012777  015332  164572            MOV     #6$,@#DZVTIV    ;SET UP THE TRANSMITTER INTERRUPT VECTOR
 (2)  015252  012777  015354  164560            MOV     #7$,@#DZVRIV    ;SET UP THE RECEIVER INTERRUPT VECTOR
 (2)  015260  012777  000200  164554            MOV     #MASK,@#DZVRIS  ;SET THE INTERRUPT VECTOR STATUS
 (2)  015266  012777  000200  164552            MOV     #MASK,@#DZVTIS  ;SET TRANSMITTER INTERRUPT PRIORITY
 (2)  015274  052777  040040  164506            BIS     #TIE!MSENAB,@#DZVCSR ;ENABLE THE DEVICE
 (1)  015302  005005                            CLR     R5              ;INIT DELAY COUNTER
 (1)  015304  005777  164500          4$:       TST     @#DZVCSR        ;TRDY SET?
 (1)  015310  100003                            BPL     5$              ;IF NOT GO DO DELAY
 (1)  015312  000240                            NOP                     ;WAIT FOR INTERRUPT
 (1)  015314  000240                            NOP
 (1)  015316  000420                            BR      8$              ;GO CLEAR TIE BIT
 (1)  015320  104414                  5$:       DELAY                   ;DELAY ROUTINE CALL
 (1)  015322  005205                            INC     R5              ;INC DELAY COUNTER
 (1)  015324  001367                            BNE     4$              ;DELAY FINISHED?
 (1)  015326  104003                            ERROR   3               ;*TRDY NOT SET!
 (1)  015330  000413                            BR      8$              ;GO CLEAR TIE
 (1)  015332  022626                  6$:       POP2SP                  ;REMOVE THE INTERRUPT FROM THE STACK
 (1)  015334  042777  040000  164446            BIC     #TIE,@#DZVCSR   ;DON'T LET ANY MORE INTERRUPTS OCCUR
 (1)  015342  105737  001425                    TSTB    DONFLG          ;PROCESSOR ALLOWING INTER?
 (1)  015346  001013                            BNE     10$             ;IF YES NO ERROR
 (1)  015350  104010                            ERROR   10              ;IF NOT PRINT ERROR
 (1)  015352  000413                            BR      9$              ;RETURN TO THE NORMAL FLOW
 (1)  015354  104012                  7$:       ERROR   12              ;*RECEIVER SHOULD NOT INTERRUPT
 (1)  015356  022626                            POP2SP                  ;POP FOR FAKE TI
 (1)  015360  042777  040000  164422  8$:       BIC     #TIE,@#DZVCSR   ;RESET TRANSMITTER INTERRUPT ENABLE
 (1)  015366  105737  001425                    TSTB    DONFLG          ;INTERRUPTS ENABLED?
 (1)  015372  001403                            BEQ     9$              ;IF NOT GET OUT
 (1)  015374  104007                            ERROR   7               ;IF YES TRANS FAILED TO INTER.
```

```
   (1)  015376  106427  000 )          10$:    MTPS    #CLEAR          ;ALLOW INTERRUPTS
   (1)  015402                          9$:
   (2)  015402  012777  015506  164434          MOV     #11$,@DZVTIV    ;SET UP THE TRANSMITTER INTERRUPT VECTOR
   (2)  015410  012777  015512  164422          MOV     #12$,@DZVRIV    ;SET UP THE RECEIVER INTERRUPT VECTOR
   (2)  015416  012777  000200  164416          MOV     #MASK,@DZVRIS   ;SET THE INTERRUPT VECTOR STATUS
   (2)  015424  012777  000200  164414          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
   (2)  015432  052777  000140  164350          BIS     #RIE!MSENAB,@DZVCSR ;ENABLE THE DEVICE
   (1)  015440  113777  001426  164366          MOVB    TDO,@DZVTDR     ;LOAD BUFFER WITH ANY CHAR.
   (1)  015446  005005                          CLR     R5              ;INIT DELAY ACCUMULATOR
   (1)  015450  105777  164334          13$:    TSTB    @DZVCSR         ;REC. DONE?
   (1)  015454  100003                          BPL     14$             ;IF NOT DELAY
   (1)  015456  000240                          NOP                     ;WAIT FOR INTERRUPT
   (1)  015460  000240                          NOP
   (1)  015462  000404                          BR      18$
   (1)  015464  104414                  14$:    DELAY                   ;DELAY FOR INTERRUPT
   (1)  015466  005205                          INC     R5              ;INCREMENT DELAY COUNTER
   (1)  015470  001367                          BNE     13$             ;DELAY FINISHED?
   (1)  015472  104004                          ERROR   4               ;*NO RX DONE! (NOT SET)
   (1)  015474  105737  001425          18$:    TSTB    DONFLG          ;PROCESSOR ALLOWING INTERRUPTS?
   (1)  015500  001411                          BEQ     15$             ;IF NOT DON'T PRINT ERROR
   (1)  015502  104011                          ERROR   11              ;RECEIVER FAILED TO INTERRUPT
   (1)  015504  000407                          BR      15$             ;CONTINUE TEST
   (1)  015506  104010                  11$:    ERROR   10              ;TRANSMITTER SHOULD NOT INTER.
   (1)  015510  000404                          BR      16$             ;CONT TEST
   (1)  015512  105737  001425          12$:    TSTB    DONFLG          ;PROCESSOR ALLOWING INTERRUPTS?
   (1)  015516  001001                          BNE     16$             ;IF YES DON'T PRINT ERROR
   (1)  015520  104012                          ERROR   12              ;*RECEIVER SHOULD NOT INTERRUPT
   (1)  015522  022626                  16$:    POP2SP                  ;POP FOR FAKE RTI
   (1)  015524  042777  040100  164256  15$:    BIC     #RIE!TIE,@DZVCSR        ;CLEAR INTERRUPTS
   (1)  015532  105737  001425                  TSTB    DONFLG          ;SECOND TIME THROUGH?
   (1)  015536  001005                          BNE     17$             ;IF YES LEAVE TEST
   (1)  015540  105237  001425                  INCB    DONFLG          ;IF NO INDICATE SECOND TEST PASS
   (1)  015544  106427  000000                  MTPS    #CLEAR          ;ALLOW INTERRUPTS
   (1)  015550  000635                          BR      1$              ;RESTART TEST
   (1)  015552  106427  000200          17$:    MTPS    #MASK           ;DON'T ALLOW INTERRUPTS
   (1)  015556  104413                          DEVICE.CLR              ;CLEAR DEVICE, LEAVE TEST
3399
3400                                             ;************************** TEST 20 ********************************
3401                                             ;*THIS TEST VERIFIES THAT THE RECEIVER WILL
3402                                             ;*INTERRUPT BEFORE THE TRANSMITTER EVEN
3403                                             ;*THOUGH THE TRANSMITTER WAS ENABLED
3404                                             ;*FIRST.  SET PS TO HIGH (MASK INTERRUPTS);
3405                                             ;*GET RDONE AND TRDY TO SET;
3406                                             ;*SET TX IE AND RX IE;
3407                                             ;*CLEAR PS AND EXPECT RX TO INTERRUPT FIRST
3409                                     ;:*  TEST 20
   (5)                                   ;:*****************************************************************
   (4)  015560  000004                  TST20:  SCOPE
   (2)  015562  012737  000020  001246          MOV     #20,$TSTNM      ;LOAD THE NUMBER OF THIS TEST
   (1)  015570  012737  004170  001362          MOV     #$EOP,NEXT      ;POINT TO THE END-OF-PASS HANDLER
3410    015576  104417                          DCLASM                  ;SET DCLR IN CSR AND MNTFLG
3411    015600  104421                          LPRSET                  ;LOAD PAR REGISTER FOR ALL LINES
3412    015602  005037  001374                  CLR     SAVLIN          ;INIT. ERROR LINE INDIC.
3413    015606  012777  016016  164224          MOV     #8$,@DZVRIV     ;SETUP INTERRUPT STUFF
3414    015614  012777  000200  164220          MOV     #MASK,@DZVRIS   ;
3415    015622  012777  016104  164214          MOV     #12$,@DZVTIV    ;
```

```
3416     5630  012777  000200  164210          MOV     #MASK,@DZVTIS   ;
3417   015636  052777  000040  164144          BIS     #MSENAB,@DZVCSR
3418   015644  012702  000001                  MOV     #1,R2           ;LINE POINTER
3419   015650  030237  001366          3$:     BIT     R2,LINE         ;VALID LINE ?
3420   015654  001515                          BEC     14$             ;IF NOT GO TO NEXT LINE
3421   015656  106427  000200          4$:     MTPS    #MASK
3422   015662  110277  164136                  MOVB    R2,@DZVTCR      ;SET TCR BIT
3423   015666  005777  164122                  TST     @DZVRBUF        ;VALID DATA?
3424   015672  100001                          BPL     .+4             ;IT BETTER NOT BE SET
3425   015674  104017                          ERROR   17              ;DATA VALID SHOULD NOT BE SET
3426   015676  105777  164106          5$:     TSTB    @DZVCSR         ;RECEIVER DONE ?
3427   015702  100001                          BPL     .+4
3428   015704  104020                          ERROR   20              ;RECEIVER DONE BIT SHOULD NOT BE SET
3429   015706  005005                          CLR     R5
3430   015710  005004                          CLR     R4
3431   015712  005777  164072          99$:    TST     @DZVCSR         ;WAIT FOR TRDY
3432   015716  100404                          BMI     100$            ;BR IF READY
3433   015720  104414                          DELAY                   ;STALL TIME
3434   015722  005204                          INC     R4              ;
3435   015724  001372                          BNE     99$
3436   015726  104003                          ERROR   3               ;TRDY FAILED TO SET
3437   015730  105077  164100          100$:   CLRB    @DZVTDR         ;SEND A ZERO CHARACTER
3438   015734  005004                          CLR     R4
3439   015736  105777  164046          6$:     TSTB    @DZVCSR         ;IS RDONE SET?
3440   015742  100404                          BMI     7$
3441   015744  104414                          DELAY
3442   015746  005204                          INC     R4
3443   015750  001372                          BNE     6$
3444   015752  104004                          ERROR   4               ;*RDONE FAILED TO SET!
3445   015754  005777  164030          7$:     TST     @DZVCSR         ;TRANS DONE BIT = 1 ?
3446   015760  100401                          BMI     .+4             ;YES
3447   015762  104003                          ERROR   3               ;*NO   TRANS DONE FAILED TO SET
3448                                           ;NOW THAT BOTH TRANSMITTER AND RECEIVER DONE BIT =1
3449                                           ;SET INTERRUPT ENABLES
3450   015764  052777  040000  164016          BIS     #TIE,@DZVCSR
3451   015772  052777  000100  164010          BIS     #RIE,@DZVCSR
3452   016000  106427  000000                  MTPS    #CLEAR          ;ALLOW THE INTERRUPTS
3453   016004  000240                          NOP
3454   016006  000240                          NOP
3455   016010  104007                          ERROR   7               ;*TRANSMITTER FAILED TO INTERRUPT
3456   016012  104011                          ERROR   11              ;*RECEIVER FAILED TO INTERRUPT
3457   016014  000435                          BR      14$             ;GET OUT
3458
3459                                           ;RECEIVER INTERRUPT ROUTINE
3460   016016  017704  163772          8$:     MOV     @DZVRBUF,R4     ;ACTUAL
3461   016022  010403                          MOV     R4,R3
3462   016024  000303                          SWAB    R3
3463   016026  042703  177770                  BIC     #^C<7>,R3       ;STRIP JUNK
3464   016032  005737  001372                  TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
  (1)  016036  100006                          BPL     11$             ;IF NOT, SKIP STAGGERED SETUP
  (1)
  (1)                                          ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
  (1)
  (1)  016040  006203                          ASR     R3              ;GET THE LAST BIT INTO THE CARRY BIT
  (1)  016042  103402                          BCS     9$              ;IF IT IS SET, GO CLEAR IT
  (1)  016044  000261                          SEC                     ;IF IT IS CLEAR SET IT HERE
```

```
  (1)  016046  000401                        BR      10$             ;SKIP THE CLEARING
  (1)  016050  000241              9$:       CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
  (1)  016052  006103              10$:      ROL     R3              ;GET THE NEW BIT BACK INTO R3
 3465  016054  020337  001374      11$:      CMP     R3,SAVLIN       ;IS THIS A VALID LINE
 3466  016060  001401                        BEQ     .+4             ;YES
 3467  016062  104015                        ERROR   15              ;*INVALID LINE
 3468  016064  042704  177400              BIC     #^C<377>,R4     ;STRIP JUNK
 3469  016070  120504                        CMPB    R5,R4           ;DATA COMPARE ?
 3470  016072  001401                        BEQ     .+4             ;YES
 3471  016074  104005                        ERROR   5               ;*DATA DOES NOT COMPARE
 3472  016076  040277  163722              BIC     R2,@DZVTCR      ;CLEAR TCR BIT
 3473  016102  000401                        BR      13$             ;GO GET OUT OF INTERRUPT MODE
 3474                                        ;TRANSMITTER INTERRUPT SVC ROUTINE
 3475  016104  104011              12$:      ERROR   11              ;THE RECEIVER INTERRUPT FAILED
 3476                                                                ;TO OVERRIDE THE TRANSMITTER
 3477  016106  022626              13$:      POP2SP                  ;REMOVE THE INTERRUPT VECTOR FROM THE STACK
 3478  016110  005237  001374      14$:      INC     SAVLIN          ;ADJUST FOR NEXT LINE
 3479  016114  104420                        SHIFT                   ;GET THE NEXT POINTER. IF DONE, ADVANCE
 3480  016116  000137  015650              JMP     3$              ;OTHERWISE GO DO THE NEXT LINE
```

  COPYRIGHT 1977,1981 DIGITAL EQUIP. CORP.

```
                                                ;ERROR TABLE
016122  000000                  .ERRTAB:        0       ;ERROR 0
016124  000000                                  0
016126  000000                                  0

016130  016270                                  EM1     ;ERROR
016132  017106                                  DH1
016134  017226                                  DT1

016136  016343                                  EM2     ;ERROR 2
016140  017132                                  DH2
016142  017240                                  DT2

016144  016371                                  EM3     ;ERROR 3
016146  017165                                  DH3
016150  017256                                  DT3

016152  016430                                  EM4     ;ERROR 4
016154  017165                                  DH3
016156  017256                                  DT3

016160  016457                                  EM5     ;ERROR 5
016162  017177                                  DH4
016164  017264                                  DT4

016166  016506                                  EM6     ;ERROR 6
016170  017177                                  DH4
016172  017264                                  DT4

016174  016545                                  EM7     ;ERROR 7
016176  017165                                  DH3
016200  017256                                  DT3

016202  016606                                  EM10    ;ERROR 10
016204  017165                                  DH3
016206  017256                                  DT3

016210  016650                                  EM11    ;ERROR 11
016212  017165                                  DH3
016214  017256                                  DT3

016216  016706                                  EM12    ;ERROR 12
016220  017165                                  DH3
016222  017256                                  DT3

016224  000000                                  0
016226  000000                                  0
016230  000000                                  0

016232  000000                                  0
016234  000000                                  0
016236  000000                                  0

016240  016745                                  EM15    ;ERROR 15
016242  000000                                  0
016244  000000                                  0
```

```
3538
3539  016246  000000                          0
3540  016250  000000                          0
3541  016252  000000                          0
3542
3543  016254  017007                          EM17    ;ERROR 17
3544  016256  017165                          DH3
3545  016260  017256                          DT3
3546
3547  016262  017045                          EM20
3548  016264  017165                          DH3
3549  016266  017256                          DT3
```

```
                                                ;ERROR MESSAGES
016270  047200  020117  052502  EM1:    .ASCIZ  <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
016343    200  042522  044507  EM2:    .ASCIZ  <200>?REGISTER R/W FAILURE?
016371    200  051124  047101  EM3:    .ASCIZ  <200>/TRANSMIT READY (TRDY) NOT SET/
016430  051200  041505  044505  EM4:    .ASCIZ  <200>/RECEIVER DONE NOT SET/
016457    200  040504  040524  EM5:    .ASCIZ  <200>/DATA COMPARISON ERROR/
016506  042200  053132  030461  EM6:    .ASCIZ  <200>/DZV11 *RECEIVER BUFFER* ERROR/
016545    200  051124  047101  EM7:    .ASCIZ  <200>/TRANSMITTER FAILED TO INTERRUPT/
016606  052600  042516  050130  EM10:   .ASCIZ  <200>/UNEXPECTED TRANSMITTER INTERRUPT/
016650  051200  041505  044505  EM11:   .ASCIZ  <200>/RECEIVER FAILED TO INTERRUPT/
016706  052600  042516  050130  EM12:   .ASCIZ  <200>/UNEXPECTED RECEIVER INTERRUPT/
016745    200  041501  044524  EM15:   .ASCIZ  <200>/ACTION DETECTED ON INVALID LINE./
017007    200  040504  040524  EM17:   .ASCIZ  <200>/DATA VALID SHOULD NOT BE SET/
017045    200  042522  042503  EM20:   .ASCIZ  <200>/RECEIVER DONE SHOULD NOT BE SET/

017106  052200  040522  020120  DH1:    .ASCIZ  <200>/TRAP PC  DZV11 REG/
017132  042600  050130  041505  DH2:    .ASCIZ  <200>/EXPECTED  FOUND  REGISTER/
017165    200  044514  042516  DH3:    .ASCIZ  <200>/LINE NO./
017177    200  054105  042520  DH4:    .ASCIZ  <200>/EXPECTED  FOUND  LINE/

                                .EVEN
                                        ;DATA TABLES FOR ERROR MESSAGES
017226  000002                  DT1:    2
017230    006     003                   .BYTE   6,3
017232  001330                          $REG1
017234    006     001                   .BYTE   6,1
017236  001326                          $REG0

                                DT2:    3
017240  000003
017242    006     004                   .BYTE   6,4
017244  001340                          $REG5
017246    006     001                   .BYTE   6,1
017250  001336                          $REG4
017252    006     001                   .BYTE   6,1
017254  001326                          $REG0

017256  000001                  DT3:    1
017260    003     001                   .BYTE   3,1
017262  001374                          SAVLIN

017264  000003                  DT4:    3
017266    006     004                   .BYTE   6,4
017270  001340                          $REG5
017272    006     001                   .BYTE   6,1
017274  001336                          $REG4
017276    003     001                   .BYTE   3,1
017300  001374                          SAVLIN

                                        ;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES
                                        ;----------------------------------------------

017302  002450                  DLYTBL: 2450                    ;TIME FOR   50 BAUD
017304  001560                          1560                    ;TIME FOR   75 BAUD
017306  001120                          1120                    ;TIME FOR  110 BAUD
017310  000750                           750                    ;TIME FOR  134 BAUD
017312  000660                           660                    ;TIME FOR  150 BAUD
```

```
3620  017314  000330                    330                    ;TIME FOR  300 BAUD
3621  017316  000150                    150                    ;TIME FOR  600 BAUD
3622  017320  000060                     60                    ;TIME FOR 1200 BAUD
3623  017322  000040                     40                    ;TIME FOR 1800 BAUD
3624  017324  000030                     30                    ;TIME FOR 2000 BAUD
3625  017326  000020                     20                    ;TIME FOR 2400 BAUD
3626  017330  000010                     10                    ;TIME FOR 3600 BAUD
3627  017332  000001                      1                    ;TIME FOR 4800 BAUD
3628  017334  000001                      1                    ;TIME FOR 7200 BAUD
3629  017336  000001                      1                    ;TIME FOR 9600 BAUD
3630  017340  000001                      1                    ;TIME OF DELAY FOR 19200 BAUD
3631
3632                                    ;DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
3633                                    ;FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.
```

```
                                .SBTTL  FALCON (KXT-11) UPGRADE ROUTINES.                       ;;GPA
                                ;
                                ; THE FOLLOWING ROUTINES HAVE BEEN ADDED TO ALLOW DIAGNOSTIC(S)
                                ; TO RUN ON A FALCON (KXT-11) BASED SYSTEM.
                                ; TO DETERMINE WHETHER WE'RE A FALCON OR NOT, WE'LL SIZE THE 1ST 3/4 OF
                                ; THE I/O PAGE (28K TO 31K).  FALCON HAS 2KW LOCAL RAM AT 28K(+4) TO 30K
                                ; AND A MACRO-ODT AT 30K TO 31K.  CONSEQUENTLY, ALL I/O DEVICES MUST
                                ; BE PLACED BETWEEN 174000 AND 177776.  ADDITIONALLY, WE'LL STRAP THE
                                ; EMT AND TRAP SERVICE LEVEL TO PRI6, AND SET THE HALT VECTOR SO THAT
                                ; WE CAN STOP THE SUCKER !!
                                ;
                                ; TO MINIMIZE THE IMPACT OF THESE CHANGES ON FINAL PROGRAM SIZE, THE
                                ; BULK OF THIS CODE IS PLACED IN THE FLOATING VECTOR SPACE (400-776).
                                ; IF THE CPU AT HAND IS A FALCON (KXT11), IT STAYS THERE (NO HARM DONE).
                                ; OTHERWISE, THE AREA IS RESTORED TO ITS ORIGINAL "TRAP-CATCHER" STATE.
                                ;
017342  005227  177777        FALCON: INC     #-1             ; ONCE-ONLY !!!                 ;;GPA
017346  001002                        BNE     1$                                              ;;GPA
017350  004737  000400                CALL    KXTCHK          ; EXECUTE FALCON CHECK          ;;GPA
017354  005727                1$:     TST     (PC)+           ; TEST FALCON FLAG...           ;;GPA
017356  000000                KXTFLAG: 0                      ;...NZ = FALCON...              ;;GPA
017360  000207                        RETURN                  ;...AND RETURN TO CALLER...     ;;GPA
                                
                                        $SVPC= .                                            ;;GPA
000400                                  .= 400                  ; RESTORE FROM 374:376 AT END   ;;GPA
000400  005037  017356        KXTCHK: CLR     KXTFLAG         ; ASSUME NOT FALCON.            ;;GPA
000404  013746  000004                MOV     @#4,-(SP)       ; SAVE ERROR VECTOR.            ;;GPA
000410  012737  000504  000004        MOV     #2$,@#4         ; SET A TRAP CATCHER.           ;;GPA
000416  012700  160010                MOV     #160010,R0      ; FALCON RAM STARTS AT 28K+4.   ;;GPA
000422  005720                1$:     TST     (R0)+           ;                               ;;GPA
000424  000240                        240                                                     ;;GPA
000426  020027  174000                CMP     R0,#174000      ; SIZE TO 31K.                  ;;GPA
000432  103773                        BLO     1$                                              ;;GPA
000434  010037  017356                MOV     R0,KXTFLAG      ; MUST BE FALCON, SET THE FLAG  ;;GPA
000440  012700  000040                MOV     #40,R0          ; GET PRI1 BIT...               ;;GPA
000444  040037  000006                BIC     R0,@#6          ;...AND LOWER BUS-ERROR...      ;;GPA
000450  040037  000016                BIC     R0,@#16         ;...BPT...                      ;;GPA
000454  040037  000022                BIC     R0,@#22         ;...IOT...                      ;;GPA
000460  040037  000032                BIC     R0,@#32         ;...EMT...                      ;;GPA
000464  040037  000036                BIC     R0,@#36         ;...AND TRAP SERVICE TO PRI6    ;;GPA
000470  012737  170000  000140        MOV     #170000,@#140   ; ENABLE "BREAK" HALT.          ;;GPA
000476  012637  000004                MOV     (SP)+,@#4       ; RESTORE ERROR VECTOR...       ;;GPA
000502  000207                        RETURN                  ;...AND RETURN.                 ;;GPA

000504  012716  000512        2$:     MOV     #3$,(SP)          TRAP -- NOT A FALCON...       ;;GPA
000510  000002                        RTI                     ;...CONTINUE.                   ;;GPA
000512  012637  000004        3$:     MOV     (SP)+,@#4       ; RESET ERROR VECTOR            ;;GPA
000516  012700  000402                MOV     #402,R0         ; SET-UP TO RESTORE FLOATING... ;;GPA
000522  013701  000376                MOV     @#376,R1        ;...VECTORS (400 - 776).        ;;GPA
000526  010602                        MOV     SP,R2           ; SAVE STACK POINTER IN R2      ;;GPA
000530  012704  000570                MOV     #6$,R4
000534  014446                4$:     MOV     -(R4),-(SP)     ; PUSH THE RESTORE CODE...      ;;GPA
000536  020427  000546                CMP     R4,#5$          ;...ONTO THE STACK.             ;;GPA
000542  101374                        BHI     4$                                              ;;GPA
000544  010607                        MOV     SP,PC           ; AND EXECUTE IT.               ;;GPA
```

```
3691                                    ;
3692                                    ; THIS CODE IS RELOCATED TO AND EXECUTED IN THE STACK AREA.
3693                                    ;
3694  000546  010060  177776           5$:     MOV     R0,-2(R0)       ; RESTORE .+2...                    ;:GPA
3695  000552  010110                           MOV     R1,(R0)         ;...HALT (OR IOT).                  ;:GPA
3696  000554  022020                           CMP     (R0)+,(R0)+                                         ;:GPA
3697  000556  020027  000776                   CMP     R0,#776                                             ;:GPA
3698  000562  101771                           BLOS    5$              ; LOOP 'TIL DONE                    ;:GPA
3699  000564  010206                           MOV     R2,SP           ; THEN RESTORE SP...                ;:GPA
3700  000566  000207                           RETURN                  ;...AND RETURN TO CALLER            ;:GPA
3701  000570                                 6$:                                                          ;:GPA
3702                                    ;
3703                                    ; IF FALCON, THIS AREA IS FREE FOR ANY PROGRAM UNIQUE
3704                                    ; CHANGES OR DATA STRUCTURES.
3705                                    ; BE SURE THAT IT DOESN'T GET SCREWED UP !!
3706                                    ;
3707                                    ; INIT $BASE AND $VECT1 AND TWEAK THE "$GETPAR" CALLING
3708                                    ; SEQUENCE TO ACCEPT THE VALID FALCON RANGE.
3709                                    ;
3710  000570  023727  001174  160010   FALCINI: CMP    $BASE,#ABASE    ; IS $BASE VIRGIN ??                ;:GPA
3711  000576  001003                           BNE     1$              ; SKIP NEXT IF NOT                  ;:GPA
3712  000600  012737  174040  001174           MOV     #174040,$BASE   ; YES, SET ENGINEERING DEFAULT      ;:GPA
3713  000606  023727  001170  000300   1$:     CMP     $VECT1,#AVECT1  ; IS $VECT1 VIRGIN ??               ;:GPA
3714  000614  001003                           BNE     2$              ; SKIP NEXT IF NOT                  ;:GPA
3715  000616  012737  000370  001170           MOV     #370,$VECT1     ; YES, SET ENGINEERING DEFAULT      ;:GPA
3716  000624  012737  000670  002404   2$:     MOV     #3$,GETCSR+2    ; SUBSTITUE CSR TEXT...             ;:GPA
3717  000632  012737  174000  002410           MOV     #174000,GETCSR+6                                    ;:GPA
3718  000640  012737  177770  002412           MOV     #177770,GETCSR+10 ;...AND VALID RANGE.              ;:GPA
3719  000646  012737  000732  002430           MOV     #4$,GETVEC+2    ; SUBSTITUTE VECTOR TEXT...         ;:GPA
3720  000654  005037  002434                   CLR     GETVEC+6                                            ;:GPA
3721  000660  012737  000370  002436           MOV     #370,GETVEC+10  ;...AND VALID RANGE.                ;:GPA
3722  000666  000207                           RETURN                  ; RETURN TO CALLER.                 ;:GPA
3723
3724  000670  030600  052123  041440   3$:     .ASCIZ  <200>'1ST CSR ADDRESS (174000:177770) '             ;:GPA
3725  000732  030600  052123  053040   4$:     .ASCIZ  <200>'1ST VECTOR ADDRESS (000:370)    '             ;:GPA
3726                                           .EVEN                                                       ;:GPA
3727
3728          000002                   $FREE= <1000-.>/2               ; FREE WORDS LEFT.                  ;:GPA
3732
3733          017362                           .=$SVPC                                                     ;:GPA
3734  017362                           CORMAX:
3738          000001                   .END
```

```
ABASE = 160010
ACDW1 = 000017
ACDW2 = 000000
ACPUOP= 000000
ACTIVE  001420
ADDW0 = 017470
ADDW1 = 017470
ADDW10= 017470
ADDW11= 017470
ADDW12= 017470
ADDW13= 017470
ADDW14= 017470
ADDW15= 017470
ADDW2 = 017470
ADDW3 = 017470
ADDW4 = 017470
ADDW5 = 017470
ADDW6 = 017470
ADDW7 = 017470
ADDW8 = 017470
ADDW9 = 017470
ADEVCT= 000000
ADEVM = 000001
ADRCNT  006037
ADVANC= 104400
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AUTO.S  011406
AVECT1= 000300
AVECT2= 000000
BINWRD  006312
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BPTVEC= 000014
BRK0  = 000400
BRK1  = 001000
BRK2  = 002000
BRK3  = 004000
BRW     004644
BUFSET= 104422
CHRCNT  006310
CLEAR = 000000
CNVRT = 104412
CONVRT= 104411
CORMAX  017362
CO0   = 000400
CO1   = 001000
CO2   = 002000
CO3   = 004000
CR    = 000015
CRLF  = 000200
CSRMAP  011414
CYCLE   010552
DATABP  007004
DATAHD  006772
DCLASM= 104417
DCLR  = 000020
DDISP = 177570
DELAY = 104414
DEVADR  006034
DEVICE= 104413
DH1     017106
DH2     017132
DH3     017165
DH4     017177
DISPLA  001306
DISPRE  000174
DLYCNT  006406
DLYTBL  017302
DONFLG  001425
DSWR  = 177570
DTR0  = 000400
DTR1  = 001000
DTR2  = 002000
DTR3  = 004000
DT1     017226
DT2     017240
DT3     017256
DT4     017264
DVALID= 100000
DZCR0   001500
DZCR1   001512
DZCR10  001620
DZCR11  001632
DZCR12  001644
DZCR13  001656
DZCR14  001670
DZCR15  001702
DZCR16  001714
DZCR17  001726
DZCR2   001524
DZCR3   001536
DZCR4   001550
DZCR5   001562
DZCR6   001574
DZCR7   001606
DZVACT  001406
DZVCSR  002010
DZVC0   001502
DZVC1   001514
DZVC10  001622
DZVC11  001634
DZVC12  001646
DZVC13  001660
DZVC14  001672
DZVC15  001704
DZVC16  001716
DZVC17  001730
DZVC2   001526
DZVC3   001540
DZVC4   001552
DZVC5   001564
DZVC6   001576
DZVC7   001610
DZVLEV  011116
DZVLPR  002020
DZVMSR  002030
DZVNUM  001414
DZVRBU  002014
DZVRIS  002042
DZVRIV  002040
DZVTCR  002024
DZVTDR  002034
DZVTIS  002046
DZVTIV  002044
DZV.EN  001740
DZV.MA  001500
EIGHT = 000030
EIGHTS= 000070
EMTVEC= 000030
EM1     016270
EM10    016606
EM11    016650
EM12    016706
EM15    016745
EM17    017007
EM2     016343
EM20    017045
EM3     016371
EM4     016430
EM5     016457
EM6     016506
EM7     016545
ERRMSG  006760
ERRVEC= 000004
ERTAB0  007134
EVEPAR= 000000
EXITER  007064
FALCIN  000570
FALCON  017342
FIVE  = 000000
FIVES = 000040
FRMERR= 020000
GETCSR= 002402
GETSWR= 007176
GETVEC= 002426
HALTS   007010
HDRFLG  001423
HDZVCS  002012
HDZVLP  002022
HDZVMS  002032
HDZVRB  002016
HDZVTC  002026
HDZVTD  002036
HILIM   006032
HT    = 000011
INBUF   010446
INIFLG  001422
INSTER= 104404
INSTR = 104403
INSTR2  005632
IOTVEC= 000020
KXTCHK  000400
KXTFLA  017356
LF    = 000012
LIMITS  005760
LINE    001366
LINE0   001504
LINE1   001516
LINE10  001624
LINE11  001636
LINE12  001650
LINE13  001662
LINE14  001674
LINE15  001706
LINE16  001720
LINE17  001732
LINE2   001530
LINE3   001542
LINE4   001554
LINE5   001566
LINE6   001600
LINE7   001612
LOBITS  006036
LOCK    001364
LOLIM   006030
LPRSET= 104421
LP0   = 000000
LP1   = 000001
LP2   = 000002
LP3   = 000003
MAINT = 000010
MANT0   001510
MANT1   001522
MANT10  001630
MANT11  001642
MANT12  001654
MANT13  001666
MANT14  001700
MANT15  001712
MANT16  001724
MANT17  001736
MANT2   001534
MANT3   001546
MANT4   001560
```

```
MANT5   001572
MANT6   001604
MANT7   001616
MASK  = 000200
MASTEK  010063
MBADLN  010176
MCSRX   010013
MDATA   010510
MEPASS  007631
MERRPC  010143
MERRX   010040
MERR2   007671
MERR3   007740
MLOCK   007764
MNEW    010066
MNTFLG  001424
MODE    001372
MPASSX  010027
MPFAIL  007566
MR      007655
MSENAB= 000040
MTITLE  001000
MTSTN   010051
MVECX   010021
NEXT    001362
ODDPAR= 000200
ONESTO= 000000
OVRRUN= 040000
PAR     001370
PARAM = 104405
PARAM1  005700
PARER = 010000
PARERR  005754
PARITY= 000100
PARMD = 104415
PAR0    001506
PAR1    001520
PAR10   001626
PAR11   001640
PAR12   001652
PAR13   001664
PAR14   001676
PAR15   001710
PAR16   001722
PAR17   001734
PAR2    001532
PAR3    001544
PAR4    001556
PAR5    001570
PAR6    001602
PAR7    001614
PAWCH = 104416
PIRQ  = 177772
PIRQVE= 000240
POPR0 = 012600
POP1SP= 005726
POP2SP= 022626
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PUSHR0= 010046
PUSH1S= 005746
PUSH2S= 024646
PWRVEC= 000024
RCVON = 010000
RDONE = 000200
REGIST  001402
RESREG  007006
RESTAR  011112
RESVEC= 000010
RES05 = 104410
RIE   = 000100
RING0 = 000001
RING1 = 000002
RING2 = 000004
RING3 = 000010
RL0   = 000000
RL1   = 000400
RL2   = 001000
RL3   = 001400
RUN     001412
R6    =%000006
R7    =%000007
SAVACT  001410
SAVLIN  001374
SAVNO   001416
SAVNUM  001415
SAVPC   001404
SAV05 = 104407
SCOP1 = 104401
SERV.G  007150
SETAPT  011260
SETFLG= 104406
SEVEN = 000020
SEVENS= 000060
SHIFT = 104420
SILOAL= 020000
SILOEN= 010000
SIX   = 000010
SIXS  = 000050
SPACNT  006311
STACK = 001120
STKLMT= 177774
STOP    001446
SV05    006046
SWR     001304
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
S110  = 001000
S1200 = 003400
S134  = 001400
S150  = 002000
S1800 = 004000
S19200= 007400
S2000 = 004400
S2400 = 005000
S300  = 002400
S3600 = 005400
S4800 = 006000
S50   = 000000
S600  = 003000
S7200 = 006400
S75   = 000400
S9600 = 007000
TBITVE= 000014
TCP0  = 000001
TCK1  = 000002
TCR2  = 000004
TCR3  = 000010
TD0     001426
TD1     001430
TD2     001432
TD3     001434
TEIGHT  002106
TFIVE   002114
TIE   = 040000
TKVEC = 000060
TLAST = 015560
TL0   = 000000
TL1   = 000400
TL2   = 001000
TL3   = 001400
TMTBL   002050
TPVEC = 000064
TRAPVE= 000034
TRDY  = 100000
TRTVEC= 000014
TR0     001436
TR1     001440
TR2     001442
TR3     001444
TSEVEN  002110
TSIX    002112
TST1    012112
TST10   013246
TST11   013332
TST12   013526
TST13   013660
TST14   014150
TST15   014472
TST16   014762
TST17   015164
TST2    012302
TST20   015560
TST3    012346
TST4    012524
TST5    012730
TST6    013032
TST7    013162
TTST    004416
TWOSTO= 000040
TYPDAT  006774
TYPE  = 104402
TYPMSG  006664
T110    002054
T1200   002066
T134    002056
T150    002060
T1800   002070
T2000   002072
T2400   002074
T300    002062
T3600   002076
T4800   002100
T50     002050
T600    002064
T7200   002102
T75     002052
T9600   00210?
VECMAP  011722
WRDCNT  006306
WTBS.F  006762
XBX     006552
XCSR    004354
XERR    004376
XHEAD   010150
XMTCNT  001400
XMTLIN  001376
XPASS   004370
XSTATQ  010240
XTSTN   007142
XVEC    004362
XX    = 160210
YY    = 000500
ZZ    = 000020
$APTHD  001446
$ATYC   005272
$ATY1   005246
$ATY3   005254
$ATY4   005264
$AUTOB  001300
$BASE   001174
$BDADR  001266
$BDDAT  001272
$CDW1   001200
$CDW2   001202
$CHARC  005242
$CMTAG  001244
$CM1  = 000006
$CM2  = 000014
$CM3  = 000006
$CM4  = 000005
$CPUOP  001146
$CRLF   001357
$DDW0   001204
$DDW1   001206
$DDW10  001230
$DDW11  001232
$DDW12  001234
$DDW13  001236
$DDW14  001240
$DDW15  001242
$DDW2   001210
```

```
$DDW3   001212          $FREE = 000002          $MTYP3  001161          $TESTN  001124          .BEGIN  004074
$DDW4   001214          $GDADR  001264          $MTYP4  001165          $TIMES  001354          .BUFSE  006500
$DDW5   001216          $GDDAT  001270          $MXCNT  004646          $TKB    001312          .CNVRT  006136
$DDW6   001220          $GET42  004330          $N    = 000020          $TKS    001310          .CONVR  006132
$DDW7   001222          $HD   = 000001          $NULL   001320          $TMP0   001342          .DCLAS  006356
$DDW8   001224          $HIBTS  001446          $NWTST= 000000          $TMP1   001344          .DELAY  006370
$DDW9   001226          $ICNT   001250          $OVER   004604          $TMP2   001346          .DEVIC  006336
$DEVCT  001130          $ILLUP  007560          $PASS   001126          $TMP3   001350          .ERRTA  016122
$DEVM   001176          $INTAG  001301          $PASTM  001454          $TMP4   001352          .INSTE  005620
$DOAGN  004350          $ITEMB  001260          $PWRAD  007554          $TN   = 000021          .INSTR  005514
$E    = 000022          $LF     001360          $PWRDN  007414          $TPB    001316          .INST1  005534
$ENDAD  004340          $LFLG   005511          $PWRMG  007550          $TPFLG  001323          .LPRSE  006440
$ENDCT  004324          $LPADR  001252          $PWRUP  007466          $TPS    001314          .MSG    005536
$ENV    001140          $LPERR  001254          $QUES   001356          $TSTM   001452          .PARAM  005640
$ENVM   001141          $MADR1  001152          $REGAD  001324          $TSTNM  001246          .PARMD  011256
$EOP    004170          $MADR2  001156          $REG0   001326          $TYPE   004712          .PAWCH  010372
$EOPCT  004316          $MADR3  001162          $REG1   001330          $TYPEC  005124          .RES05  006100
$ERFLG  001247          $MADR4  001166          $REG2   001332          $TYPEX  005244          .SAV05  006040
$ERMAX  001261          $MAIL   001120          $REG3   001334          $UNIT   001132          .SCOPE  004404
$ERROR  006522          $MAMS1  001150          $REG4   001336          $UNITM  001456          .SCOP1  004650
$ERRPC  001262          $MAMS2  001154          $REG5   001340          $USWR   001144          .SETFL  010252
$ERRTB  001362          $MAMS3  001160          $RTNAD  004352          $VECT1  001170          .SHIFT  006422
$ERTTL  001256          $MAMS4  001164          $SAVR6  007564          $VECT2  001172          .START  002116
$ETABL  001140          $MBADR  001450          $SCOPE  004404          $XOFF = 000023          .TRPSR  006314
$ETEND  001244          $MFLG   005510          $SETUP= 000000          $XON  = 000021          .TRPTA  001742
$FATAL  001122          $MSGAD  001134          $SVLAD  004566          $XTSTR  004444          .TYPE   004674
$FFLG   005512          $MSGLG  001136          $SVPC = 017362          $Y    = 000023          .$X   = 001446
$FILLC  001322          $MSGTY  001120          $SWR  = 164000          $$GET4= 000000
$FILLS  001321          $MTYP1  001151          $SWREG  001142          .     = 017362
$FLIP = 177777          $MTYP2  001155          $SWRMK= 000000          .ADVAN  006410

. ABS.  017362     000        CON   RO    REL   GBL   D

 ERRORS DETECTED:  0

 CVDZAC,CVDZAC=CVDZAC
 RUN-TIME: 21 14 .6 SECONDS
 RUN-TIME RATIO: 191/37=5.1
 CORE USED:  37K  (73 PAGES)
```