DZV11

DZV11 DIAG PRT 2
CVDZBB0

AH-A939B-MC
FICHE 1 OF 1

```
        .REM    &
```

IDENTIFICATION

PRODUCT CODE:	AC-A938B-MC
PRODUCT NAME:	CVDZBB0 DZV11 DIAG PRT2
DATE RELEASED:	17-FEB-82
MAINTAINER:	DIAGNOSTIC ENGINEERING

1. ABSTRACT

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

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

CURRENTLY THERE ARE THREE STANDALONE DIAGNOSTICS (DVDZA,DVDZB,AND DVDZC) ONE SYSTEM MODULE FOR DEC X/11 (DZBA), AND AN OVERLAY FOR ITEP (DVDZD).

DVDZA TOGETHER WITH DVDZB WILL TEST ALL LOGICAL FUNCTIONS OF THE DZV11 INTERFACE MODULE.
DVDZC IS DESIGNED AS A NON-CHAINABLE STANDALONE DIAGNOSTIC PROVIDING THE OPERATOR WITH DIRECT CONTROL OVER THE TESTING OF ALL DZV11 EIA CABLES.

```
**************************************************************************
*                                                                        ;;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 RESIDE. LOCATION 1500 THRU 1740 ARE ESPECIALLY TO BE NOTED AND TO BE UNTOUCHED BY OPERATOR AFTER PARAMETERS HAVE BEEN INPUT FROM CONSOLE (SW00=1); OR AFTER THE 'AUTO SIZING' HAS BEEN DONE. THESE LOCATIONS MAY BE CHANGED IF THE USER UNDERSTANDS THEIR MEANING AND DIFFERENT PARAMETERS ARE REQUIRED.

3. LOADING PROCEEDURE

3.1 METHOD

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

ABSOLUTE LOADER STARTING ADDRESS *500

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

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

4. STARTING PROCEEDURE

A. SET SWR TO ZERO FOR 'AUTO SIZING' OR SET SW00=1 FOR USER PARAMETER INPUT FROM CONSOLE TERMINAL. NOTE: LOC. 000176 IS USED AS A SOFTWARE SWITCH REGISTER IN ALL OF THE DZV11 DIAGNOSTICS. (SEE SEC. 4.1) ON THE FIRST STARTUP OF THE DIAGNOSTIC IF SW07=1 AND SW00=0 THE PROGRAM WILL ASSUME THAT THE STATUS TABLE HAS BEEN ALREADY BUILT FROM A PREVIOUS DZV11 DIAGNOSTIC RUN. NOTE: ANY DZV11 DIAGNOSTIC WILL OVERLAY THE STATUS TABLE WHEN LOADED TO PRESERVE ITS CONTENTS AND THUS WILL NOT ALTER A PREVIOUSLY BUILT TABLE.

B. START THE DIAGNOSTIC AT LOC. 200(8). THE PROGRAM WILL TYPE MAINDEC AND PROGRAM NAMES (IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING: (ON THE FIRST PROGRAM RUN OR IF PARAMETERS WERE CHANGED)

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

THE ABOVE IS ONLY AN EXAMPLE! THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 1500 IN THE PROGRAM. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.

THE PROGRAM WILL TYPE 'RUNNING'' AND PROCEED TO RUN THE DIAGNOSTIC.

4.1 CONTROL SWITCH SETTINGS

NOTE: THIS PROGRAM UTILIZES A SOFTWARE SWITCH REGISTER WHICH MAY BE MODIFIED BY CHANGING LOC. 176 OR BY TYPING CONTROL 'G'' (^G) ON THE CONSOLE TERMINAL WHILE THE PROGRAM IS RUNNING.

```
SW 15   SET:  HALT ON ERROR
SW 14   SET:  LOOP ON CURRENT TEST
SW 13   SET:  INHIBIT ERROR PRINT OUT
SW 12   SET:  INHIBIT **ALL** TYPE OUT/BELL ON ERROR.
SW 11   SET:  INHIBIT ITERATIONS.  (QUICK PASS)
SW 10   SET:  ESCAPE TO NEXT TEST
SW 09   SET:  LOOP WITH CURRENT DATA
SW 08   SET:  CATCH ERROR AND LOOP ON IT
SW 07   SET:  NO AUTO SIZE.  IF 1ST START OF  PROGRAM AFTER  LOADING AND
              IF SW00=0 THEN THE PROGRAM WILL ASSUME THAT THE STATUS MAP
              HAS BEEN BUILT FROM A PREVIOUS DZV11 DIAGNOSTIC RUN.
SW 06   SET:  RESELECT DZV11'S DESIRED ACTIVE
SW 05   SET:  RESERVED
SW 04   SET:  SELECT DELAY PARAMETER (SEE SEC. 4.1.1)
SW 03   SET:  EXTRA PARAMETER INPUT  (SEE SEC. 4.1.1)
SW 02   SET:  LOCK ON SELECTED TEST
SW 01   SET:  RESTART PROGRAM AT SELECTED TEST
SW 00   SET:  GET USERS PARAMETERS FROM CONSOLE
```

4.1.1 SWITCH REGISTER CONTROL OF PARAMETER INPUT FROM CONSOLE

SW 00 GET USERS PARAMETERS FROM CONSOLE. SETTING THIS SWITCH AT START UP TIME ALLOWS THE USER TO INPUT AT THE CONSOLE TERMINAL THE FOLLOWING PARAMETERS: BASE DEVICE ADDRESS, BASE VECTOR ADDRESS, MODE OF OPERATION (EXTERNAL, INTERNAL, OR STAGGERED), AND THE NUMBER OF DZV11'S THAT ARE RUNNING. USING THIS SWITCH ALONE WILL DEFAULT THE FOLLOWING PARAMETERS: ALL 4 LINES ARE SET TO BE TESTED ON EACH DZV11, THE DEFAULT BAUD RATE IS SET AT 19.2 KBAUD AND THE CHARACTER LENGTH FOR THE MAJORITY OF TESTING IS SET AT EIGHT BITS PER CHARACTER WITH TWO STOP BITS.

SW 03 EXTRA PARAMETER INPUT. SETTING THIS SWITCH AT START UP TIME PROVIDES THE USER WITH THE ABILITY TO SET THE LINES ACTIVE FOR TESTING AND TO SET THE DEFAULT BAUD RATE USED FOR THE MAJORITY OF THE DIAGNOSTIC TESTS. THE DELAY PARAMETER IS AUTOMATICALLY ADJUSTED TO THE BAUD RATE GIVEN BY THE USER.

SW 04 SELECT DELAY PARAMETER. THE DELAY PARAMETER THIS SWITCH CONTROLS DETERMINES THE LENGTH OF TIME THE PROGRAM STALLS WAITING FOR A CHARACTER TO BE COMPLETELY TRANSMITTED OR RECEIVED. THIS DELAY COUNT IS AUTOMATICALLY SET TO PROVIDE ENOUGH DELAY TIME FOR THE DEFAULT BAUD RATE SPECIFIED WHEN RUNNING THE PROGRAM ON AN LSI11 WITH MOS MEMORY. WHEN RUNNING THIS PROGRAM ON A PROCESSOR WITH A FASTER MEMORY SPEED THIS DELAY COUNT SHOULD BE ADJUSTED PROPORTIONATELY HIGHER THAN THE FOLLOWING DEFAULTED VALUES:

```
2450        ;TIME FOR   50 BAUD
1560        ;TIME FOR   75 BAUD
1120        ;TIME FOR  110 BAUD
0750        ;TIME FOR  134 BAUD
0660        ;TIME FOR  150 BAUD
0330        ;TIME FOR  300 BAUD
0150        ;TIME FOR  600 BAUD
0060        ;TIME FOR 1200 BAUD
0040        ;TIME FOR 1800 BAUD
0030        ;TIME FOR 2000 BAUD
0020        ;TIME FOR 2400 BAUD
0010        ;TIME FOR 3600 BAUD
0001        ;TIME FOR 4800 BAUD
0001        ;TIME FOR 7200 BAUD
0001        ;TIME FOR 9600 BAUD
0001        ;TIME FOR 19.2 KBAUD
```

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 06 RESELECT DZV11'S DESIRED ACTIVE. A MESSAGE IS TYPED OUT ON THE CONSOLE TERMINAL ASKING THE OPERATOR TO TYPE A BIT MAP OF THE DZV'S DESIRED ACTIVE. USING THIS SWITCH ALLOWS LOCATION DZVACTV TO BE ALTERED (SEE SEC. 8.3 FOR A DESCRIPTION OF THIS LOCATION).
EXAMPLE:
IF THE DEVICES CORRESPONDING TO THE DZV11'S NUMBERED ZERO, TWO, AND FOUR IN THE DZV11 STATUS MAP (LOC. 1500 THROUGH 1740) ARE TO BE TESTED, TYPE IN: 25
THIS WILL SET BITS ZERO, TWO, AND FOUR IN LOCATION DZVACTV. ALL REMAINING DEVICES IN THE STATUS MAP WILL THEN NOT BE TESTED.

SW 01 RESTART PROGRAM AT SELECTED TEST IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A 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 STARTING RESTRICTIONS

SEE SECTION 4.1.2
THE STATUS TABLE SHOULD BE VERIFIED REGARDLESS OF HOW THE PROGRAM WAS STARTED. ALSO IT IS IMPORTANT TO USE THIS LISTING ALONG WITH THE INFORMATION PRINTED ON THE TTY TO COMPLETELY ISOLATE PROBLEMS.

7.2 OPERATING RESTRICTIONS

PARAMETER MUST BE INPUT FROM USER OR APT IF "AUTO SIZING" IS NOT USED.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DZV11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 2 MIN. THIS IS ASSUMING SW11=1 (INHIBIT ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION.

8.2 PASS COMPLETE

NOTE: *EVERY* TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO *HARD* ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTIL ALL DZV11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS DVDZB-A CSR: 160100 VEC: 300 PASSES: 000001 ERRORS: 000000

NOTE: THE NUMBERS FOR CSR AND VEC ARE NOT NECESSARILY THE VALUES FOR THE DEVICE. THEY ARE ONLY FOR THIS EXAMPLE.

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.
1504	000017	THIS IS THE BINARY REPRESENTATION OF WHAT LINES ARE TO BE TESTED.
1506	017470	THIS IS THE PARAMETER LOCATION USED IN MOST OF THE TESTS. IT INDICATES PARAMETERS OF: RX ON, SPEED SELECT 17 (19.2K BAUD) EIGHT BITS PER CHAR, AND TWO STOP BITS. THE USER MAY ALTER THE STOP BITS AND THE SPEED, BUT THE REMAINING PARAMETERS SHOULD BE LEFT ALONE. THIS LOCATION IS USED TO LOAD THE DZV11 LINE PARAMETER REGISTER FOR EACH LINE. THE MEANING OF THE BITS SET IN THIS LOCATION IS THE SAME AS THE FUNCTION OF THE RELATED BITS IN THE DEVICE LINE PARAMETER REGISTER.
1510	000000	THIS LOCATION WILL CONTAIN EITHER ALL ZEROS INDICATING THAT INTERNAL LOOP WAS SELECTED AS MODE OF OPERATION OR IT WILL CONTAIN 100000 INDICATING THAT "STAGGERED MODE" WAS SELECTED OR IT WILL CONTAIN 000200 INDICATING THAT "EXTERNAL" WAS THE MODE SELECTED.

THE ABOVE IS REPEATED FOR EACH DZV11 IN THE SYSTEM. THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT PROGRAM AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND TO SUIT THE SPECIFIC CONFIGURATION.

8.5 *** METHOD OF AUTO SIZING ***

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE PROGRAM WILL START AT ADDRESS 160000 AND START 'REFERENCING' THE ADDRESS IN THE POINTER. IF A NON-EX MEMORY TRAP OCCURES, THE POINTER (HOLDING 160000) IS UPDATED BY 10 AND THE ABOVE IS REPEATED UNTIL ADDRESS 163770 IS REACHED. IF A 'BUS REPLY' RESPONSE WAS ISSUED BY THE DZV11 (OR ANY OTHER DEVICE) (NO NXM TRAP), 'MASTER SCAN ENABLE" IS ATTEMPTED TO BE SET AND THE TCR BITS FOR ALL FOUR LINES ARE SET. "TRDY" IS THEN TESTED TO BE SET AND 'MASTER SCAN ENABLE" IS TESTED TO BE STILL SET. THE DIAGNOSTIC WILL THEN CHECK THAT AT LEAST ONE TCR BIT IS STILL SET. IF ALL OF THE ABOVE WORKED, THIS DEVICE IS ASSUMED TO BE A DZV11. IF ANY OF THE ABOVE FAILED, UPDATING OF THE POINTER IS DONE AND THE 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 (DVDZA, AND DVDZB) CAN BE RUN AS STANDALONE DIAGNOSTICS OR IN EITHER OF THE APT MODES. DVDZC, HOWEVER IS DESIGNED AS A STANDALONE DIAGNOSTIC ONLY AND REQUIRES DIRECT OPERATOR PARTICIPATION.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

THE DIAGNOSTIC USES SEVERAL VARIABLES IN THE REGION SUBTITLED '' APT MAILBOX-ETABLE'. THESE VARIABLES ARE:

$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), JAN 19, 1977.


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

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

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

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

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

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

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

                1072    THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                        AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                        AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                        THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                        SW14=1  LOOP ON TEST
                        SW11=1  INHIBIT ITERATIONS
                        CALL
                                SCOPE           ;;SCOPE=IOT

                1147    ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                        THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                        NOTE1:  $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                        NOTE2:  $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                        NOTE3:  $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

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

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

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

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

```
                2192    ********** TEST 2 **********
                        THIS TEST THAT "SILO ENABLE" WILL INHIBIT
                        RECEIVER INTERRUPTS AND THAT ON THE
                        16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
                        INTERRUPT WITH "RIE" SET.
                        THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
                        ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.

                2264    ********** TEST 3 **********
                        THIS TEST RUNS ALL LINES FULL BORE
                        BASED UPON QUALIFIED LINES
                        ..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
                        TRANSMITTER

                2397    ********** TEST 4 **********
                        DZV11 RELATIVE TIMING TEST.
                        EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
                        AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
                        WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
                        DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
                        THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
                         AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
                        PARAMETERS ARE:
                          EIGHT BITS/PER/CHAR - TWO STOP BITS AT
                                50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
                                2400, 3600, 4800, 7200, 9600 BAUD.
                          19.2 K BAUD - TWO STOP BITS AT
                                SEVEN, SIX, FIVE BITS/PER/CHAR.
                        AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
                        THE NEXT SELECTED LINE IS THEN TESTED.
                        WHEN RUNNING UNDER THE APT MANUFACTURING SYSTEM
                        THIS TEST IS ONLY RUN THE FIRST PASS

                2491    ********** TEST 5 **********
                        THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
                        THAT "PE" (PARITY ERROR) CAN BE FLAGGED BY
                        THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
                        YOU ARE IN "STAGGERED" MODE.
                        40(8) CHARS ARE USED FOR THIS TEST.
                        ALL SELECTED LINES WILL BE ENABLED AT THE SAME TIME.
                        THIS TEST FIRST CHECKS EVEN PARITY FOR ODD LINES AND
                        ODD PARITY FOR EVEN LINES, THEN IT CHECKS THE REVERSE.

                &
```

```
  2727                          ;;GPA   PRGFRT  ^?MAINDEC-11-DVDZBA/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 OF 2?,DVDZBA
  2728                          ;;GPA   .HEADER <MD-11-DVDZB-A>,1977
   (2)                          .TITLE  CVDZB-B
   (2)                          ;*COPYRIGHT (C) 1977,1981
   (2)                          ;*DIGITAL EQUIPMENT CORP.
   (2)                          ;*MAYNARD, MASS. 01754
   (2)                          ;*
   (2)                          ;*
   (2)                          ;*THIS PROGRAM WAS ASSEMBLED USIN. THE PDP-11 MAINDEC SYSMAC
   (2)                          ;*PACKAGE (MAINDEC-11-DZQAC-C5),  AN, 1981.
   (2)                          ;*
   (2)          000001          $TN=1
   (2)                                  ;STARTING PROCEDURE
   (2)                                  ;LOAD PROGRAM
   (2)                                  ;LOAD ADDRESS 000200
   (2)                                  ;PRESS START
   (2)                                  ;PROGRAM WILL TYPE
   (2)                                  ;"CVDZBB/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 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,SWC
(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  004362                      .SCOPE                  ;SCOPE LOOP HANDLER
(1)     000022  000200                      MASK                    ;HANDLE AT PRIORITY 7
(1)     000024  007372                      $PWRDN                  ;POWER FAIL HANDLER
(1)     000026  000340                      340                     ;SERVICE AT PRIORITY LEVEL 7
(1)     000030  006500                      $ERROR                  ;ERROR HANDLER
(1)     000032  000340                      340                     ;SERVICE AT PRIORITY LEVEL 7
(1)     000034  006272                      .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  004316                      $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>/CVDZBB/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 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)  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 NUMBER 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                     $APTHDR ;;POINT TO APT HEADER BLOCK
(2)          001446                     .=.$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  000132             $TSTM:  .WORD   90.     ;;RUN TIM OF LONGEST TEST
(2)  001454  000137             $PASTM: .WORD   95.     ;;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)
(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  006366                             .ADVANCE
(3)        104401                       SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
(2)  001744  004626                             .SCOP1
(3)        104402                       TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
(2)  001746  004652                             .TYPE
(3)        104403                       INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
(2)  001750  005472                             .INSTR
(3)        104404                       INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
(2)  001752  005576                             .INSTER
(3)        104405                       PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
(2)  001754  005616                             .PARAM
(3)        104406                       SETFLG=TRAP+6           ;CALL TO  SET FLAG ROUTINE
(2)  001756  010230                             .SFTFLG
(3)        104407                       SAV05=TRAP+7            ;CALL TO REGISTER SAVE ROUTINE
(2)  001760  006016                             .SAV05
(3)        104410                       RES05=TRAP+10           ;CALL TO REGISTER RESTORE ROUTINE
(2)  001762  006056                             .RES05
(3)        104411                       CONVRT=TRAP+11          ;CALL TO DATA OUTPUT ROUTINE
(2)  001764  006110                             .CONVRT
(3)        104412                       CNVRT=TRAP+12           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
(2)  001766  006114                             .CNVRT
(3)        104413                       DEVICE.CLR=TRAP+13              ;CALL TO ISSUE A DEVICE CLEAR
(2)  001770  006314                             .DEVICE.CLR
(3)        104414                       DELAY=TRAP+14           ;CALL TO DELAY FOR FAST CPU'S
(2)  001772  006346                             .DELAY
(3)        104415                       PARMD=TRAP+15           ;CONVERT DECIMAL STRING TO OCTAL
(2)  001774  011234                             .PARMD
(3)        104416                       PAWCH=TRAP+16           ;SET FLAG    ECHO OR  C  LE
(2)  001776  010350                             .PAWCH
(3)        104417                       DCLASM=TRAP+17          ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2)  002000  006334                             .DCLASM
(3)        104420                       SHIFT=TRAP+20           ;CALL TO ROTATE LINE POINTER
(2)  002002  006400                             .SHIFT
(3)        104421                       LPRSET=TRAP+21          ;CALL TO SET UP LPR DEVICE REGISTER
(2)  002004  006416                             .LPRSET
(3)        104422                       BUFSET=TRAP+22          ;CALL TO ZERO BUFFER AREA
(2)  002006  006456                             .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)
(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  007372  000024             MOV     #$PWRDN,@#24    ;SET UP POWER FAIL VECTOR
(1)  002136  005037  001126                     CLR     $PASS           ;CLEAR PASS COUNT
(1)  002142  105037  001247                     CLRB    $ERFLG          ;CLEAR ERROR FLAG
(1)  002146  012737  001500  001420             MOV     #DZV.MAP,ACTIVE ;GET MAP POINTER.
(1)  002154  012737  000001  001412             MOV     #1,RUN          ;POINT POINTER TO FIRST DEVICE.
(1)  002162  005037  001256                     CLR     $ERTTL          ;CLEAR ERROR COUNT
(1)  002166  005037  001262                     CLR     $ERRPC          ;CLEAR LAST ERROR POINTER
(1)  002172  005037  001246                     CLR     $TSTNM          ;SET UP FOR TEST 1
(1)  002176  012737  002116  001252             MOV     #.START,$LPADR  ;SET UP FOR POWER FAIL BEFORE
(1)                                                                     ;TESTING STARTS
(1)                                             ;SET UP FOR SMALL 11 SWITCH REGISTER COMPATIBILITY
(1)  002204  012737  000176  001304             MOV     #SWREG,SWR      ;POINT TO SOFTWARE SWR
(1)  002212  012737  000174  001306             MOV     #DISPREG,DISPLAY ;POINT TO SOFTWARE DISPLAY REGISTER
(1)  002220  004737  017126                     CALL    FALCON          ; CHECK FOR FALCON (KXT11)      ;;GPA
(1)  002224  001402                             BEQ     1000$           ; BR IF NOT                     ;;GPA
(1)  002226  004737  000570                     CALL    FALCINI         ; YES, INIT FOR FALCON.         ;;GPA
(1)  002232                                 1000$:                                                      ;;GPA
(1)  002232  105737  001422                     TSTB    INIFLG          ;HAVE WE ALREADY BEEN HERE TODAY?
(1)  002236  001010                             BNE     10$             ;IF SO, SKIP PRINTING THE TITLE
(1)  002240  023727  000042  004316             CMP     @#42,#$ENDAD    ;IF RUNNING UNDER ACT
(1)  002246  001402                             BEQ     1$              ;DON'T PRINT TITLE
(1)  002250  104402  001000                     TYPE    ,MTITLE         ;PRINT THE DIAGNOSTIC'S TITLE
(1)  002254  105337  001422                 1$: DECB    INIFLG          ;SET THE ONCE ONLY FLAG
(1)  002260  105737  001141                 10$: TSTB   $ENVM           ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
(1)  002264  100004                             BPL     15$             ;IF NOT, GO CHECK FOR AUTO-SIZING
(1)  002266  004737  011236                     JSR     PC,SETAPT       ;OTHERWISE, GO DO APT SIZING FROM ETABLE
(1)  002272  000137  003606                     JMP     105$            ;GO PRINT DZV STATUS TABLE
(1)  002276  005737  000042                 15$: TST    @#42            ; CHAINED UNDER XXDP ??         ;;GPA
(1)  002302  001404                             BEQ     16$             ; BR IF NOT                     ;;GPA
(1)  002304  004737  011236                     CALL    SETAPT          ; YES, SET-UP FROM ETABLE       ;;GPA
(1)  002310  000137  003606                     JMP     105$            ; AND PROCEED                   ;;GPA
(1)  002314  004737  007154                 16$: CALL   GETSWR          ; GET AN INITIAL SWR            ;;GPA
(1)  002320  032777  000001  176756             BIT     #SW00,@SWR      ;RESELECT ?
(1)  002326  001002                             BNE     20$             ;IF YES, GO SET UP THE INFORMATION
(1)  002330  000137  002632                     JMP     55$             ;IF NO, SKIP THE INTERROGATION
(1)  002334  012700  001500                 20$: MOV    #DZV.MAP,R0     ;POINT TO THE BEGINNING OF THE MAP TABLE
(1)  002340  105037  001423                     CLRB    HDRFLG          ;MAKE SURE A MAP GETS PRINTED
(1)  002344  005020                         25$: CLR    (R0)+           ;CLEAR A TABLE LOCATION
(1)  002346  020027  001740                     CMP     R0,#DZV.END     ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  002352  001374                             BNE     25$             ;IF NOT ,CLEAR THE NEXT LOCATION IN THE TABLE
(1)  002354  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)             002360                          GETCSR= .               ; POINTER FOR FALCON TWEAKER.   ;;GPA
(2)   002360    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002362    003052                          91$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002364    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002366    160000                          160000                  ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002370    163770                          163770                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002372    001500                          DZCR0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002374       007                          .BYTE   7               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002375       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)   002376    013737  001500  001174          MOV     DZCR0,$BASE     ;COPY BASE ADDRESS TO ETABLE
(1)
(1)                                             ;GET THE BASE VECTOR ADDRESS
(1)             002404                          GETVEC= .               ; POINTER FOR FALCON TWEAKER.   ;;GPA
(2)   002404    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002406    003116                          92$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002410    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002412    000300                          300                     ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002414    000776                          776                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002416    001502                          DZVC0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002420       003                          .BYTE   3               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002421       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)   002422    013737  001502  001170          MOV     DZVC0,$VECT1    ;COPY VECTOR TO ETABLE
(1)
(1)                                             ;GET THE MODE OF OPERATION (E,I,S)
(2)   002430    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002432    003345                          96$                     ;POINTER TO THE MESSAGE TO BE PRINTED
(2)   002434    104406                          SETFLG                  ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
(2)   002436    001510                          MANT0                   ;THIS IS THE FLAG BEING SETUP
(1)
(1)                                             ;GET THE NUMBER OF DZV11'S RUNNING
(2)   002440    104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)   002442    003302                          95$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)   002444    104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)   002446    000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002450    000020                          16.                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)   002452    001344                          $TMP1                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)   002454       000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)   002455       001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)
(1)   002456    012737  000017  001504          MOV     #17,LINE0       ;SET UP DEFAULT LINES
(1)   002464    012737  017470  001506          MOV     #17470,PAR0     ;SET UP DEFAULT LPR PARAMETER
(1)                                             ;RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
(1)   002472    032777  000010  176604          BIT     #SW03,@SWR      ;DO YOU WANT PARAMETERS?
(1)   002500    001402                          BEQ     30$             ;IF NO, SKIP THE PARAMETER CALL
(1)   002502    004737  002662                  JSR     PC,65$          ;GET PARAMETERS
(1)   002506    012737  000001  001410  30$:    MOV     #1,SAVACTV      ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
(1)   002514    113737  001344  001414          MOVB    $TMP1,DZVNUM    ;COPY THE NUMBER OF DEVICES
(1)   002522    005337  001344          35$:    DEC     $TMP1           ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
(1)   002526    001404                          BEQ     40$             ; SELECTED DEVICES
(1)   002530    000261                          SEC                     ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
(1)   002532    006137  001410                  ROL     SAVACTV         ;POINT TO THE NEXT DEVICE
(1)   002536    000771                          BR      35$             ;GO DO THIS PROCEDURE AGAIN
(1)   002540    013737  001410  001346  40$:    MOV     SAVACTV,$TMP2   ;# OF TIMES
(1)   002546    012700  001500                  MOV     #DZCR0,R0       ;SET A POINTER TO THE SPECIFIED INFORMATION
```

```
(1)  002552  012701  001512                    MOV     #DZCR1,R1       ;POINT R1 TO THE REST OF THE MAP TABLE
(1)  002556  012702  001204                    MOV     #$DDW0,R2       ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
(1)  002562  000241                            CLC                     ;INITIALIZE THE "C" BIT FOR A ROTATION
(1)  002564  006037  001346                    ROR     $TMP2           ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
(1)  002570  006237  001346            45$:    ASR     $TMP2           ;ISOLATE A SELECTION FLAG IN THE "C" BIT
(1)  002574  103404                            BCS     50$             ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
(1)  002576  012711  177777                    MOV     #-1,(R1)        ;TERMINATE THE LIST
(1)  002602  000137  003550                    JMP     100$            ;GO TO THE NEXT BLOCK
(1)  002606  012011                    50$:    MOV     (R0)+,(R1)      ;ADDRESS
(1)  002610  062721  000010                    ADD     #10,(R1)+       ;POINT TO THE NEXT DZV11 ADDRESS VALUE
(1)  002614  012011                            MOV     (R0)+,(R1)      ;VECTOR
(1)  002616  062721  000010                    ADD     #10,(R1)+       ;POINT TO THE NEXT VECTOR VALUE
(1)  002622  012021                            MOV     (R0)+,(R1)+     ;LINES
(1)  002624  012021                            MOV     (R0)+,(R1)+     ;PARAMETERS
(1)  002626  012021                            MOV     (R0)+,(R1)+     ;MAINTENANCE MODE
(1)  002630  000757                            BR      45$
(1)  002632  032777  000010  176444    55$:    BIT     #SW03,@SWR      ;ASK PARAMETERS ?
(1)  002640  001002                            BNE     60$             ;IF NO, GO DO AUTO SIZING
(1)  002642  000137  003550                    JMP     100$            ;GO SET UP FOR AUTO SIZING
(1)  002646  004737  002662            60$:    JSR     PC,65$          ;GO ASK PARAMETERS
(1)  002652  105337  001422                    DECB    INIFLG          ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
(1)  002656  000137  003606                    JMP     105$            ;GO TO THE NEXT BLOCK
(1)
(1)                                            ;GET THE ACTIVE LINES PARAMETER
(1)
(1)  002662                            65$:
(2)  002662  104403                            INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002664  003157                            93$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002666  104405                            PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002670  000001                            1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002672  000017                            17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002674  001504                            LINE0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002676     360                            .BYTE   360             ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002677     001                            .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002700  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)  002704  005737  001510                    TST     MANTO           ;IS STAGGERED THE MODE OF OPERATION?
(1)  002710  100021                            BPL     85$             ;IF NOT, SKIP THIS SEGMENT
(1)  002712  013703  001504                    MOV     LINE0,R3        ;GET A SCRATCH COPY OF THE ACTIVE LINES
(1)  002716  006003                    70$:    ROR     R3              ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
(1)  002720  103410                            BCS     80$             ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
(1)  002722  001414                            BEQ     85$             ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
(1)  002724  006203                            ASR     R3              ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
(1)  002726  103373                            BCC     70$             ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
(1)  002730  104402  001356            75$:    TYPE    ,$QUES          ;THIS IS AN INCORRECT PARAMETER
(1)  002734  104402  010154                    TYPE    ,MBADLN         ;LET THE USER KNOW ABOUT IT
(1)  002740  000750                            BR      65$             ;GO GET THE CORRECT PARAMETER
(1)  002742  001772                    80$:    BEQ     75$             ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
(1)  002744  006203                            ASR     R3              ;GET THE NEXT FLAG
(1)  002746  103370                            BCC     75$             ;IF IT ISN'T SET, THERE'S AN ERROR
(1)  002750  000241                            CLC                     ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
(1)  002752  000761                            BR      70$             ;GO TEST THE NEXT PAIR OF FLAGS
(1)
```

```
(1)                                             ;GET THE LINE PARAMETER REGISTER ARGUMENT
(1)
(1)  002754                             85$:
(2)  002754  104403                             INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002756  003232                             94$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002760  104405                             PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002762  000000                             0                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002764  000017                             17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002766  001506                             PAR0                    ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002770     000                             .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002771     001                             .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002772  012702  001504                     MOV     #LINE0,R2       ;POINT TO THE LINE SELECTION PARAMETER
(1)  002776  012703  001506                     MOV     #PAR0,R3        ;POINT TO THE CHOSEN PARAMETERS
(1)  003002  011304                             MOV     (R3),R4         ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
(1)  003004  006304                             ASL     R4              ;ALIGN INDEX ON WORD BOUNDARY
(1)  003006  016437  017066  006364             MOV     DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
(1)  003014  000313                             SWAB    (R3)            ;PLACE IN HIGH BYTE
(1)  003016  052713  010070                     BIS     #10070,(R3)     ;PLACE EXTRA PARAMETERS INTO LOC
(1)  003022  011262  000012             90$:    MOV     (R2),12(R2)     ;LOAD THE LINES
(1)  003026  011363  000012                     MOV     (R3),12(R3)     ;LOAD THE PARAMETERS
(1)  003032  062702  000012                     ADD     #12,R2          ;POINT TO THE NEXT SET
(1)  003036  062703  000012                     ADD     #12,R3          ;.. OF BOTH PARAMETERS
(1)  003042  020327  001734                     CMP     R3,#PAR17       ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  003046  001365                             BNE     90$             ;IF NOT, GO LOAD SOME MORE PARAMETERS
(1)  003050  000207                             RTS     PC              ;RETURN TO CALLING BLOCK
(1)  003052  030600  052123  041440  91$:      .ASCIZ  <200>/1ST CSR ADDRESS (160000:163770):  /
(1)  003116  030600  052123  053040  92$:      .ASCIZ  <200>/1ST VECTOR ADDRESS (300:770):  /
(1)  003157     200  044514  042516  93$:      .ASCIZ  <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:17):  /
(1)  003232  042200  043105  052501  94$:      .ASCIZ  <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17):  /
(1)  003302  021600  047440  020106  95$:      .ASCIZ  <200>/# OF DZV11'S <IN OCTAL> (1:20):  /
(1)  003345     200  040515  047111  96$:      .ASCII  <200>/MAINTENANCE MODE/
(1)  003366  020200  042533  052130             .ASCII  <200>/ [EXTERNAL  <H325>     (E)]/
(1)  003422  020200  044533  052116             .ASCII  <200>/ [INTERNAL  <DZVCSR03=1>(I)]/
(1)  003457     200  055440  052123             .ASCIZ  <200>/ [STAGGERED <H329>     (S)]: /
(1)  003516  042600  052116  051105  97$:      .ASCIZ  <200>/ENTER DELAY PARAMETER: /
(1)          003550                             .EVEN
(1)  003550                             100$:
(1)  003550  122737  000377  001422             CMPB    #377,INIFLG     ;ONLY DO AUTO SIZE ON 1ST START
(1)  003556  001013                             BNE     105$            ;
(1)  003560  032777  000200  175516             BIT     #BIT7,@SWR      ;BIT7=1??
(1)  003566  001007                             BNE     105$            ;BR IF NO AUTO SIZE
(1)  003570  005737  017142                     TST     KXTFLAG         ; KXT11 ??                      ;;GPA
(1)  003574  001402                             BEQ     1001$           ; SKIP NEXT IF NOT              ;;GPA
(1)  003576  000137  002334                     JMP     20$             ; YES, DON'T AUTO-SIZE          ;;GPA
(1)  003602                             1001$:                                                          ;;GPA
(1)  003602  004737  011364                     JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
(1)  003606  105737  001423             105$:   TSTB    HDRFLG          ;HAS THE TABLE BEEN TYPED YET?
(1)  003612  001021                             BNE     120$            ;IF SO, DON'T TYPE IT AGAIN
(1)  003614  105337  001423                     DECB    HDRFLG          ;INDICATE THAT THE TABLE WILL BE TYPED
(1)  003620  104402  010126                     TYPE    ,XHEAD          ;TYPE MAP HEADER
(1)  003624  012700  001500                     MOV     #DZV.MAP,R0     ;SET POINTER
(1)  003630  010037  001344             110$:   MOV     R0,$TMP1        ;POINT TO THE MAP LOCATION
(1)  003634  012037  001346                     MOV     (R0)+,$TMP2     ;SET DATA
(1)  003640  022737  177777  001346             CMP     #-1,$TMP2       ;END OF LIST?
(1)  003646  001403                             BEQ     120$            ;BR IF YES
(1)  003650  104411                     115$:   CONVRT                  ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
```

```
(1)  003652  010216                          XSTATQ                  ;CONVERT THE DATA AT THIS ADDRESS
(1)  003654  000765                          BR      110$            ;GO PRINT THE NEXT PARAMETER
(1)  003656  013737  001410  001406  120$:   MOV     SAVACTV,DZVACTV ;COPY BIT MAP OF ACTIVE DEVICES
(1)  003664  113737  001414  001416          MOVB    DZVNUM,SAVNO    ;COPY NO. OF DEVICES IN THE SYSTEM
(1)  003672  032777  000100  175404          BIT     #SW06,@SWR      ;DESELECT SPECIFIC DEVICES??
(1)  003700  001431                          BEQ     135$            ;BR IF NO.
(1)  003702                          121$:
(2)  003702  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003704  010044                          MNEW                    ;POINTER TO MESSAGE TO BE PRINTED
(2)  003706  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  003710  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003712  177777                          177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003714  001406                          DZVACTV                 ;POINTER TO MAP LOCATION TO BE FILLED
(2)  003716     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  003717     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  003720  023737  001406  001410          CMP     DZVACTV,SAVACTV ;IS VALUE VALID?
(1)  003726  101403                          BLOS    122$            ;IF YES BRANCH
(1)  003730  104402  007716                  TYPE    ,MERR3          ;IF NOT TYPE ERROR
(1)  003734  000762                          BR      121$            ;THEN REASK QUESTION
(1)  003736  105037  001416          122$:   CLRB    SAVNO           ;INITIALIZE NO. OF ACTIVE DEVICES
(1)  003742  013737  001406  001344          MOV     DZVACTV,$TMP1   ;COPY BIT MAP OF ACTIVE DEVICES
(1)  003750  006237  001344          126$:   ASR     $TMP1           ;ROTATE OUT AN ACTIVE BIT
(1)  003754  103002                          BCC     127$            ;IF NOT ACTIVE SKIP RECORDING IT
(1)  003756  105237  001416                  INCB    SAVNO           ;INCREMENT NO. OF ACTIVE DEVICES
(1)  003762  001372                  127$:   BNE     126$            ;IF NOT DONE GO CONTINUE
(1)  003764  032777  000020  175312  135$:   BIT     #SW04,@SWR      ;CHECK TO SEE IF DELAY COUNT CHANGES
(1)  003772  001407                          BEQ     140$            ;IF NOT, GO CLEAR VECTOR AREA
(2)  003774  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003776  003516                          97$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  004000  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  004002  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  004004  177777                          177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  004006  006364                          DLYCNT                  ;POINTER TO MAP LOCATION TO BE FILLED
(2)  004010     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  004011     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  004012  012700  000300          140$:   MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
(1)  004016  012701  000302                  MOV     #302,R1         ;VECTOR AREA. 300-776
(1)  004022  010120                  145$:   MOV     R1,(R0)+        ;START PUTTING 'PC+2 - HALT''
(1)  004024  005021                          CLR     (R1)+           ;IN VECTOR AREA.
(1)  004026  022021                          CMP     (R0)+,(R1)+     ;POP POINTERS
(1)  004030  005737  017142                  TST     KXTFLAG         ; IF FALCON...                  ;;GPA
(1)  004034  001403                          BEQ     1002$           ;                               ;;GPA
(1)  004036  020027  000400                  CMP     R0,#400         ;...QUIT AT 400.                ;;GPA
(1)  004042  000402                          402                     ; SKIP NEXT                     ;;GPA
(1)  004044                          1002$:                          ;                               ;;GPA
(1)  004044  022700  001000                  CMP     #1000,R0        ;ALL DONE??
(1)  004050  001364                          BNE     145$            ;BR IF NO.
(1)
(1)                                  ;TEST START AND RESTART
(1)                                  ;----------------------
(1)
(1)  004052  012706  001120          .BEGIN: MOV     #STACK,SP       ;SET UP STACK
(1)  004056  106427  000200                  MTPS    #MASK           ;LOCK OUT INTERRUPTS
(1)  004062  005737  000042                  TST     @#42            ;IS PROGRAM UNDER MONITOR CONTROL
(1)  004066  001015                          BNE     2$              ;BR IF YES
(1)  004070  032777  000004  175206          BIT     #BIT2,@SWR      ;CHECK FOR LOOP ON TEST
```

```
  (1)  004076  001406                          BEQ     1$              ;BR IF NO LOCK DESIRED.
  (1)  004100  104402  007742                  TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
  (1)  004104  012737  000240  004374          MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
  (1)  004112  000403                          BR      2$              ;CONTINUE ALONG.
  (1)  004114  013737  004622  004374  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
  (1)  004122  012737  010530  001252  2$:     MOV     #CYCLE,$LPADR   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
  (1)  004130  113737  001416  001415          MOVB    SAVNO,SAVNUM    ;COPY NO. OF ACTIVE DEVICES
  (1)  004136  104402  007633                  TYPE    ,MR             ;TYPE "RUNNING"
  (1)  004142  000177  175104                  JMP     @$LPADR         ;START TESTING
 2729                                  ::GPA   PRGEND  DZV11,<END PASS DVDZB-A  >,10.
```

SEQ 0041

```
                        ;*END OF PASS ROUTINE

  2730
   (2)                          ;END 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)  004146          $EOP:
   (5)  004146  000004          SCOPE
   (5)  004150  005037  001262  CLR     $ERRPC          ;CLEAR LAST ERROR PC
   (5)  004154  105037  001247  CLRB    $ERFLG          ;CLEAR ERROR FLAG
   (5)  004160  104402  007607  TYPE    ,MEPASS         ;TYPE END PASS
   (5)  004164  104402  007771  TYPE    ,MCSRX          ;TYPE CSR
   (5)  004170  104412  004332  CNVRT   ,XCSR          ;SHOW IT
   (5)  004174  104402  007777  TYPE    ,MVECX          ;TYPE VECTOR
   (5)  004200  104412  004340  CNVRT   ,XVEC          ;SHOW IT
   (5)  004204  005237  001126  INC     $PASS           ;RAISE PASS COUNT
   (5)  004210  104402  010005  TYPE    ,MPASSX         ;TYPE PASSES
   (5)  004214  104412  004346  CNVRT   ,XPASS         ;SHOW IT
   (5)  004220  005337  001126  DEC     $PASS           ;RESTORE PASS COUNT
   (5)  004224  104402  010016  TYPE    ,MERRX          ;TYPE ERRORS
   (5)  004230  104412  004354  CNVRT   ,XERR          ;SHOW IT
   (5)  004234  005237  001130  INC     $DEVCT          ;INC DEVCNT FOR APT
   (5)  004240  1053??  001415  DECB    SAVNUM          ;ARE ALL DEVICES TESTED?
   (5)  004244  0610??          BNE     $DOAGN          ;BR IF NO.
   (5)  004246  113737  001416  001415  MOVB    SAVNO,SAVNUM    ;RESTORE THE COUNT
   (3)  004254  005037  001354  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
   (3)  004260  005237  001126  INC     $PASS           ;;INCREMENT THE PASS NUMBER
   (3)  004264  042737  100000  001126  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
   (3)  004272  005327          DEC     (PC)+           ;;LOOP?
   (3)  004274  000001  $EOPCT: .WORD   1
   (3)  004276  003013          BGT     $DOAGN          ;;YES
   (3)  004300  012737          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
   (3)  004302  000001  $ENDCT: .WORD   1
   (3)  004304  004274          $EOPCT
   (3)  004306  013700  000042  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
   (3)  004312  001405          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
   (3)  004314  000005          RESET                   ;;CLEAR THE WORLD
   (3)  004316  004710  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
   (3)  004320  000240          NOP                     ;;SAVE ROOM
   (3)  004322  000240          NOP                     ;;FOR
   (3)  004324  000240          NOP                     ;;ACT11
   (3)  004326  $DOAGN:
   (3)  004326  000137          JMP     @(PC)+          ;;RETURN
   (3)  004330  010530  $RTNAD: .WORD   CYCLE
   (2)
   (2)  004332  000001  XCSR:   1
   (2)  004334     006  002             .BYTE   6,2
   (2)  004336  002010          DZVCSR
   (2)  004340  000001  XVEC:   1
   (2)  004342     003  002             .BYTE   3,2
```

```
(2)  004344  002040                          DZVRIV
(2)  004346  000001                  XPASS:  1
(2)  004350     006     002                  .BYTE   6,2
(2)  004352  001126                          $PASS
(2)  004354  000001                  XERR:   1
(2)  004356     006     002                  .BYTE   6,2
(2)  004360  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)  004362                          $SCOPE:
(5)  004362  005037  001262          .SCOPE: CLR     $ERRPC          ;CLEAR LAST ERROR PC.
(5)  004366  022716  012072                  CMP     #TST1+2,(SP)    ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
(5)  004372  001413                          BEQ     $XTSTR          ;IF SO, DON'T LOOP ON IT
(5)  004374  000406                  TTST:   BR      1$              ;GOTO 1$    (IF LOCK SW02=1; THIS LOC =240)
(5)  004376  105777  174706                  TSTB    @$TKS           ;KEYBOARD DONE?
(5)  004402  100067                          BPL     $OVER           ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
(5)  004404  017766  174702  177776          MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
(3)  004412  032777  040000  174664  1$:     BIT     #BIT14,@$WR     ;;LOOP ON PRESENT TEST?
(3)  004420  001060                          BNE     $OVER           ;;YES IF SW14=1
(3)                                  ;#####START OF CODE FOR THE XOR TESTER#####
(3)  004422  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3)  004424  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3)  004430  012737  004450  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(3)  004436  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
(3)  004442  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004446  000436                          BR      $SVLAD          ;;GO TO THE NEXT TEST
(3)  004450  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(3)  004452  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004456  000441                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
(3)  004460                          6$:;#####END OF CODE FOR THE XOR TESTER#####
(3)  004460  105737  001247          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(3)  004464  001404                          BEQ     3$              ;;BR IF NO
(3)  004466  105037  001247          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(3)  004472  005037  001354                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3)  004476  032777  004000  174600  3$:     BIT     #BIT11,@$WR     ;;INHIBIT ITERATIONS?
(3)  004504  001011                          BNE     1$              ;;BR IF YES
(3)  004506  005737  001126                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(3)  004512  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
(3)  004514  005237  001250                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
(3)  004520  023737  001354  001250          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(3)  004526  002015                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
```

```
(3)  004530  012737  000001  001250  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(3)  004536  013737  004624  001354          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(3)  004544  105237  001246          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(3)  004550  113737  001246  001124          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(3)  004556  011637  001252                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(3)  004562  013777  001246  174516  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(3)  004570  013716  001252                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(5)  004574  004737  007126                  JSR     PC,^ERV.G       ;FIND OUT IF ^G WAS TYPED
(5)  004600  105037  001424                  CLRB    MNTFLG          ;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
(5)  004604  005737  001372                  TST     MODE            ;HAS THE MODE BEEN CHANGED?
(5)  004610  001003                          BNE     4$              ;IF NOT INTERNAL , GO DO A TEST
(5)  004612  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
(5)  004620  000002                  4$:     RTI                     ;GO DO THE TEST
(5)  004622  000406                  BRW:    406
(3)  004624  000005                  $MXCNT: 5                       ;;MAX. NUMBER OF ITERATIONS
(1)
(1)                                          ;CHECK FOR FREEZE ON CURRENT DATA
(1)                                          ;------------------------------
(1)
(1)  004626  032777  001000  174450  .SCOP1: BIT     #SW09,@SWR      ;IS SW09=1(SET)?
(1)  004634  001405                          BEQ     1$              ;BR IF NOT SET.
(1)  004636  005737  001364                  TST     LOCK            ;IS THERE A TIGHT LOOP SPECIFIED?
(1)  004642  001402                          BEQ     1$              ;IF NO, RETURN
(1)  004644  013716  001364                  MOV     LOCK,(SP)       ;IF YES, GOTO THE ADDRESS IN LOCK.
(1)  004650  000002                  1$:     RTI                     ;GO BACK.
(1)
(1)  004652  032777  010000  174424  .TYPE:  BIT     #SW12,@SWR      ;INHIBIT ALL PRINTOUT??
(1)  004660  001403                          BEQ     $TYPE           ;IF NOT, GO TYPE
(1)  004662  062716  000002                  ADD     #2,(SP)         ;SKIP OVER MESSAGE POINTER
(1)  004666  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)  004670  105737  001323          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(2)  004674  100002                          BPL     1$              ;;BR IF YES
(2)  004676  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
(2)  004700  000430                          BR      3$              ;;LEAVE
(2)  004702  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
(2)  004704  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(2)  004710  122737  000001  001140          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  0047'4  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(2)  0047c0  132737  000100  001141          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
(2)  004726  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(2)  004730  010037  004740                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(2)  004734  004737  005232                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(2)  004740  000000                61$:    .WORD   0               ;;MESSAGE ADDRESS
(2)  004742  132737  000040  001141  62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(2)  004750  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
(2)  004752  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2)  004754  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(2)  004756  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(2)  004760  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
(2)  004762  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(2)  004766  000002                        RTI                     ;;RETURN
(2)  004770  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(2)  004774  001430                        BEQ     8$
(2)  004776  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(2)  005002  001006                        BNE     5$
(2)  005004  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(2)  005006  104402                        TYPE                    ;;TYPE A CR AND LF
(2)  005010  001357                        $CRLF
(2)  005012  105037  005220                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(2)  005016  000755                        BR      2$              ;;GET NEXT CHARACTER
(2)  005020  004737  005102        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(2)  005024  123726  001322        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(2)  005030  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(2)  005032  013746  001320                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(2)                                                                 ;;AND THE NULL CHAR.
(2)  005036  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(2)  005042  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2)  005044  004737  005102                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(2)  005050  105337  005220                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(2)  005054  000770                        BR      7$              ;;LOOP
(2)
(2)                                 ;HORIZONTAL TAB PROCESSOR
(2)
(2)  005056  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(2)  005062  004737  005102        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(2)  005066  132737  000007  005220        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(2)  005074  001372                        BNE     9$              ;;TAB STOP
(2)  005076  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(2)  005100  000724                        BR      2$              ;;GET NEXT CHARACTER
(2)  005102                         $TYPEC:
(2)  005102  105777  174202                TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
(2)  005106  100022                        BPL     10$             ;;BR IF NOT                     ;MJD001
(2)  005110  017746  174176                MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
(2)  005114  042716  177600                BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
(2)  005120  122716  000023                CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
(2)  005124  001012                        BNE     102$            ;;BR IF NOT                     ;MJD001
(2)  005126                         101$:                                                          ;MJD001
(2)  005126  105777  174156                TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
(2)  005132  100375                        BPL     101$                                            ;MJD001
(2)  005134  117716  174152                MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
(2)  005140  042716  177600                BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
(2)  005144  122716  000021                CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
(2)  005150  001366                        BNE     101$            ;;BR IF NOT                     ;MJD001
(2)  005152                         102$:                                                          ;MJD001
(2)  005152  005726                        TST     (SP)+           ;;FIX STACK                     ;MJD001
```

```
(2)  005154                          10$:                                                          ;MJD001
(2)  005154  105777  174134                 TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(2)  005160  100375                         BPL     10$                                           ;MJD001
(2)  005162  116677  000002  174126         MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)  005170  122766  000015  000002         CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(2)  005176  001003                         BNE     1$              ;;BRANCH IF NO
(2)  005200  105037  005220                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(2)  005204  000406                         BR      $TYPEX          ;;EXIT
(2)  005206  122766  000012  000002  1$:    CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(2)  005214  001402                         BEQ     $TYPEX          ;;BRANCH IF YES
(2)  005216  105227                         INCB    (PC)+           ;;COUNT THE CHARACTER
(2)  005220  000000          $CHARCNT:.WORD 0               ;;CHARACTER COUNT STORAGE
(2)  005222  000207          $TYPEX: RTS    PC
(2)
(2)                          .SBTTL  APT COMMUNICATIONS ROUTINE
(2)
(3)                          ;;*****************************************************************
(2)  005224  112737  000001  005470 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(2)  005232  112737  000001  005466 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(2)  005240  000403                         BR      $ATYC
(2)  005242  112737  000001  005470 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(2)  005250                          $ATYC:
(4)  005250  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(4)  005252  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(2)  005254  105737  005466                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(2)  005260  001450                         BEQ     5$              ;;IF NOT:  BR
(2)  005262  122737  000001  001140         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(2)  005270  001031                         BNE     3$              ;;IF NOT:  BR
(2)  005272  132737  000100  001141         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(2)  005300  001425                         BEQ     3$              ;;IF NOT:  BR
(2)  005302  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(2)  005306  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(2)  005314  005737  001120          1$:    TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(2)  005320  001375                         BNE     1$              ;;IF NOT:  WAIT
(2)  005322  010037  001134                 MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(2)  005326  105720                  2$:    TSTB    (R0)+           ;;FIND END OF MESSAGE
(2)  005330  001376                         BNE     2$
(2)  005332  163700  001134                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(2)  005336  006200                         ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(2)  005340  010037  001136                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(2)  005344  012737  000004  001120         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(2)  005352  000413                         BR      5$
(2)  005354  017637  000004  005400  3$:    MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(2)  005362  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(4)  005370  013746  177776                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(2)  005374  004737  004670                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
(2)  005400  000000                  4$:    .WORD   0
(2)  005402                          5$:
(2)  005402  105737  005470          10$:   TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(2)  005406  001416                         BEQ     12$             ;;IF NOT:  BR
(2)  005410  005737  001140                 TST     $ENV            ;;RUNNING UNDER APT?
(2)  005414  001413                         BEQ     12$             ;;IF NOT:  BR
(2)  005416  005737  001120          11$:   TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(2)  005422  001375                         BNE     11$             ;;IF NOT:  WAIT
(2)  005424  017637  000004  001122         MOV     @4(SP),$FATAL   ;;GET ERROR #
(2)  005432  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
```

```
(2)  005440  005237  001120                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(2)  005444  105037  005470          12$:   CLRB    $FFLG           ;;CLEAR FATAL FLAG
(2)  005450  105037  005467                 CLRB    $LFLG           ;;CLEAR LOG FLAG
(2)  005454  105037  005466                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(4)  005460  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(4)  005462  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(2)  005464  000207                         RTS     PC              ;;RETURN
(2)  005466     000                 $MFLG:  .BYTE   0               ;;MESSG. FLAG
(2)  005467     000                 $LFLG:  .BYTE   0               ;;LOG FLAG
(2)  005470     000                 $FFLG:  .BYTE   0               ;;FATAL FLAG
(2)          005472                         .EVEN
(2)          000200                 APTSIZE=200
(2)          000001                 APTENV=001
(2)          000100                 APTSPOOL=100
(2)          000040                 APTCSUP=040
(1)
(1)                                         ;STRING INPUT ROUTINE
(1)                                         ;--------------------
(1)
(1)  005472  010346                 .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
(1)  005474  010446                         MOV     R4,-(SP)        ;SAVE R4 ON STACK
(1)  005476  017637  000004  005514         MOV     @4(SP),.MSG     ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1)  005504  062766  000002  000004         ADD     #2,4(SP)        ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1)  005512  104402                 .INST1: TYPE                    ;PRINT THE MESSAGE
(1)  005514  000000                 .MSG:   0                       ;MESSAGE IS POINTED TO FROM HERE
(1)  005516  012704  010424                 MOV     #INBUF,R4       ;POINT R4 TO THE INPUT BUFFER
(1)  005522  012703  000007                 MOV     #7,R3           ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1)  005526  105777  173556          1$:    TSTB    @$TKS           ;HAS A CHARACTER BEEN RECEIVED?
(1)  005532  100375                         BPL     1$              ;IF NO, KEEP WAITING FOR IT
  )  005534  117714  173552                 MOVB    @$TKB,(R4)      ;IF YES, SAVE IT IN THE INPUT BUFFER
(1)  005540  142714  000200                 BICB    #200,(R4)       ;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1)  005544  122427  000015                 CMPB    (R4)+,#15       ;IS THIS CHARACTER A LINE FEED?
(1)  005550  001417                         BEQ     INSTR2          ;IF SO, TERMINATE THE INPUT SEQUENCE
(1)  005552  105777  173536          2$:    TSTB    @$TPS           ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1)  005556  100375                         BPL     2$              ;IF WE CAN'T, WAIT UNTIL WE CAN
(1)  005560  017777  173526  173530         MOV     @$TKB,@$TPB     ;ECHO THE CHARACTER BACK
(1)  005566  005303                         DEC     R3              ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
(1)  005570  001356                         BNE     1$              ;IF WE DON'T HAVE 7, GO GET SOME MORE
(1)  005572  012604                         MOV     (SP)+,R4        ;IF WE HAVE 7, RESTORE R4
(1)  005574  012603                         MOV     (SP)+,R3        ;RESTORE R3
(1)  005576  010346                 .INSTE: MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
(1)  005600  010446                         MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
(1)  005602  104402  001356                 TYPE    ,$QUES          ;PRINT A QUESTION MARK... WHAT'S GOING ON?
(1)  005606  000741                         BR      .INST1          ;GO PRINT THE MESSAGE AGAIN
(1)  005610  012604                 INSTR2: MOV     (SP)+,R4        ;RESTORE R4
(1)  005612  012603                         MOV     (SP)+,R3        ;RESTORE R3
(1)  005614  000002                         RTI                     ;RETURN TO THE MAIN PROCEDURE
(1)
(1)                                         ;CONVERT ASCII STRING TO OCTAL
(1)                                         ;----------------------------
(1)
(1)  005616  010546                 .PARAM: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
(1)  005620  010446                         MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
(1)  005622  016605  000004                 MOV     4(SP),R5        ;GET THE SETUP INFORMATION POINTER
(1)  005626  012537  006006                 MOV     (R5)+,LOLIM     ;SET THE LOW LIMIT FOR THE INPUT
(1)  005632  012537  006010                 MOV     (R5)+,HILIM     ;SET THE HIGH LIMIT FOR THE INPUT
```

```
(1)  005636  012537  006012                PARAM1: MOV     (R5)+,DEVADR    ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
(1)  005642  112537  006014                        MOVB    (R5)+,LOBITS    ;GET THE MASK OF THE INCORRECT BITS
(1)  005646  112537  006015                        MOVB    (R5)+,ADRCNT    ;GET THE COUNT OF ITEMS TO BE STORED
(1)  005652  010566  000004                        MOV     R5,4(SP)        ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
(1)  005656  005005                                CLR     R5              ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
(1)  005660  012704  010424                        MOV     #INBUF,R4       ;POINT TO THE INPUT BUFFER
(1)  005664  122714  000015                        CMPB    #15,(R4)        ;IS THIS CHARACTER A CARRIAGE RETURN?
(1)  005670  001420                                BEQ     PARERR          ;IF SO, PRINT THE MESSAGE AGAIN
(1)  005672  121427  000060                1$:     CMPB    (R4),#60        ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
(1)  005676  002415                                BLT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)  005700  121427  000067                        CMPB    (R4),#67        ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
(1)  005704  003012                                BGT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)  005706  142714  000060                        BICB    #60,(R4)        ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
(1)  005712  152405                                BISB    (R4)+,R5        ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
(1)  005714  122714  000015                        CMPB    #15,(R4)        ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
(1)  005720  001406                                BEQ     LIMITS          ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
(1)  005722  006305                                ASL     R5              ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
(1)  005724  006305                                ASL     R5              ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
(1)  005726  006305                                ASL     R5              ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
(1)                                                                        ;NEXT THREE BITS
(1)  005730  000760                                BR      1$              ;GO GET THE NEXT CHARACTER
(1)  005732  104404                PARERR: INSTER                          ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
(1)  005734  000750                                BR      PARAM1          ;TRY GETTING THE PARAMETERS AGAIN
(1)
(1)                                                ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(1)                                                ;--------------------------------------
(1)
(1)  005736  020537  006010          LIMITS: CMP     R5,HILIM        ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
(1)  005742  101373                          BHI     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005744  020537  006006                  CMP     R5,LOLIM        ;IS THE RESULT LOWER THAN ALLOWED?
(1)  005750  103770                          BLO     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005752  133705  006014                  BITB    LOBITS,R5       ;ARE ANY INCORRECT BITS SET IN THE RESULT?
(1)  005756  001365                          BNE     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)
(1)                                                ;STORE NUMBER AT SPECIFIED ADDRESS
(1)
(1)  005760  013704  006012                  MOV     DEVADR,R4       ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1)  005764  010524                  1$:     MOV     R5,(R4)+        ;STORE THE RESULT
(1)  005766  062705  000002                  ADD     #2,R5           ;CALCULATE THE NEXT DATUM
(1)  005772  105337  006015                  DECB    ADRCNT          ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1)  005776  001372                          BNE     1$              ;IF NOT, GO STORE THE NEXT DATUM
(1)  006000  012604                          MOV     (SP)+,R4        ;RESTORE R4
(1)  006002  012605                          MOV     (SP)+,R5        ;RESTORE R5
(1)  006004  000002                          RTI                     ;RETURN TO THE MAIN PROGRAM
(1)
(1)  006006  000000                  LOLIM:  0                       ;LOWEST ACCEPTABLE VALUE
(1)  006010  000000                  HILIM:  0                       ;HIGHEST ACCEPTABLE
(1)  006012  000000                  DEVADR: 0                       ;LOCATION WHERE RESULT WILL BE STORED
(1)  006014     000                  LOBITS: .BYTE   0               ;INCORRECT BITS MASK
(1)  006015     000                  ADRCNT: .BYTE   0               ;COUNT OF ITEMS TO BE STORED
(1)
(1)                                                ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1)                                                ;-------------------------------------
(1)
(1)  006016  016637  000004  001404  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
(1)
```

```
(1)                                             ;SAVE R0-R5
(1)
(1)  006024  010537  001340        SV05:  MOV     R5,$REG5        ;SAVE R5
(1)  006030  010437  001336               MOV     R4,$REG4        ;SAVE R4
(1)  006034  010337  001334               MOV     R3,$REG3        ;SAVE R3
(1)  006040  010237  001332               MOV     R2,$REG2        ;SAVE R2
(1)  006044  010137  001330               MOV     R1,$REG1        ;SAVE R1
(1)  006050  010037  001326               MOV     R0,$REG0        ;SAVE R0
(1)  006054  000002                       RTI                     ;LEAVE.
(1)
(1)                                             ;RESTORE R0-R5
(1)
(1)  006056  013700  001326        .RES05: MOV     $REG0,R0        ;RESTORE R0
(1)  006062  013701  001330               MOV     $REG1,R1        ;RESTORE R1
(1)  006066  013702  001332               MOV     $REG2,R2        ;RESTORE R2
(1)  006072  013703  001334               MOV     $REG3,R3        ;RESTORE R3
(1)  006076  013704  001336               MOV     $REG4,R4        ;RESTORE R4
(1)  006102  013705  001340               MOV     $REG5,R5        ;RESTORE R5
(1)  006106  000002                       RTI                     ;LEAVE
(1)
(1)                                             ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(1)                                             ;-----------------------------------------------------
(1)
(1)  006110  104402  001357        .CONVR: TYPE    ,$CRLF          ;PRINT A CARRIAGE RETURN
(1)  006114  010046                .CNVRT: MOV     R0,-(SP)        ;SAVE R0
(1)  006116  010146                        MOV     R1,-(SP)        ;SAVE R1
(1)  006120  010346                        MOV     R3,-(SP)        ;SAVE R3
(1)  006122  010446                        MOV     R4,-(SP)        ;SAVE R4
(1)  006124  010546                        MOV     R5,-(SP)        ;SAVE R5
(1)  006126  017601  000012                MOV     @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
(1)  006132  062766  000002  000012        ADD     #2,12(SP)       ;POINT TO WHERE MAIN PROGRAM WILL RESUME
(1)  006140  012137  006264                MOV     (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
(1)  006144  112105                1$:     MOVB    (R1)+,R5        ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
(1)  006146  112100                        MOVB    (R1)+,R0        ;GET THE NUMBER OF SPACES TO PRINT
(1)  006150  013104                        MOV     @(R1)+,R4       ;COPY THE WORD TO BE CONVERTED
(1)  006152  110537  006266                MOVB    R5,CHRCNT       ;COPY THE CHARACTER COUNT
(1)  006156  010403                3$:     MOV     R4,R3           ;COPY THE ARGUMENT WORD AGAIN
(1)  006160  042703  177770                BIC     #^C<7>,R3       ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
(1)  006164  062703  000060                ADD     #060,R3         ;MAKE AN ASCII CHARACTER OUT OF THEM
(1)  006170  110346                        MOVB    R3,-(SP)        ;SAVE THAT CHARACTER
(1)  006172  006004                        ROR     R4              ;MOVE THE NEXT THREE BITS INTO PLACE
(1)  006174  006204                        ASR     R4              ;MOVE THEM AGAIN
(1)  006176  006204                        ASR     R4              ;AND FINALLY A THIRD TIME
(1)  006200  005305                        DEC     R5              ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
(1)                                                                 ;BUILT?
(1)  006202  001365                        BNE     3$              ;IF NO, GO BUILD THE NEXT ONE.
(1)  006204  012703  010466                MOV     #MDATA,R3       ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1)  006210  112623                4$:     MOVB    (SP)+,(R3)+     ;STORE THE CHARACTER, STARTING WITH THE MOST
(1)  006212  105337  006266                DECB    CHRCNT          ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1)  006216  001374                        BNE     4$              ;IF NO, GO TRANSFER ANOTHER
(1)  006220  105700                        TSTB    R0              ;ARE ANY SPACES TO BE PRINTED?
(1)  006222  001404                        BEQ     6$              ;IF NO, DON'T SET UP ANY
(1)  006224  112723  000040        5$:     MOVB    #040,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
(1)  006230  105300                        DECB    R0              ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1)  006232  001374                        BNE     5$              ;IF YES, GO ADD ANOTHER SPACE
(1)  006234  105013                6$:     CLRB    (R3)            ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
```

```
(1)  006236  104402  010466                TYPE    ,MDATA          ;PRINT THE STRING WE JUST BUILT
(1)  006242  005337  006264                DEC     WRDCNT          ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1)  006246  001336                        BNE     1$              ;IF YES, GO CONVERT THEM
(1)  006250  012605                        MOV     (SP)+,R5        ;RESTORE R5
(1)  006252  012604                        MOV     (SP)+,R4        ;RESTORE R4
(1)  006254  012603                        MOV     (SP)+,R3        ;RESTORE R3
(1)  006256  012601                        MOV     (SP)+,R1        ;RESTORE R1
(1)  006260  012600                        MOV     (SP)+,R0        ;RESTORE R0
(1)  006262  000002                        RTI                     ;RETURN TO THE MAIN PROGRAM
(1)  006264  000000              WRDCNT:   0
(1)  006266     000              CHRCNT:   .BYTE                   ;NUMBER OF CHARACTERS TO PRINT
(1)  006267     000              SPACNT:   .BYTE   0               ;NUMBER OF SPACES TO PRINT
(1)
(1)  006270  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)  006272  010046              .TRPSR:   MOV     R0,-(SP)        ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
(1)  006274  016600  000002                MOV     2(SP),R0        ;GET TRAP ADDRESS
(1)  006300  005740                        TST     -(R0)           ;GET TRAP
(1)  006302  111000                        MOVB    (R0),R0         ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
(1)  006304  006300                        ASL     R0              ;POSITION OFFSET FOR TABLE INDEXING
(1)  006306  016000  001742                MOV     .TRPTAB(R0),R0  ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1)  006312  000200                        RTS     R0              ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                                        ;DEVICE CLEAR ROUTINE
(1)                                        ;ISSUE A DEVICE CLEAR
(1)                                        ;--------------------
(1)  006314                      .DEVICE.CLR:
(1)  006314  052777  000020  173466        BIS     #DCLR,@DZVCSR   ;SET DCLR
(1)  006322  032777  000020  173460  1$:   BIT     #DCLR,@DZVCSR   ;DID IT CLEAR?
(1)  006330  001374                        BNE     1$              ;BR IF NO
(1)  006332  000002                        RTI                     ;EXIT ROUTINE
(1)
(1)                                        ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                                        ;-----------------------------------------------------------
(1)  006334  104413              .DCLASM:  DEVICE.CLR              ;ISSUE A DEVICE CLEAR
(1)  006336  153777  001424  173444        BISB    MNTFLG,@DZVCSR  ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1)  006344  000002                        RTI                     ;RETURN TO CALLING ROUTINE
(1)
(1)  006346                      .DELAY:
(1)  006346  010046                        MOV     R0,-(SP)        ;SAVE R0
(1)  006350  013700  006364                MOV     DLYCNT,R0       ;SET COUNT
(1)  006354  005300              1$:       DEC     R0              ;DELAY
(1)  006356  001376                        BNE     1$              ;
(1)  006360  012600                        MOV     (SP)+,R0        ;RESTORE R0
(1)  006362  000002                        RTI                     ;LEAVE ROUTINE
(1)  006364  000001              DLYCNT:   .WORD     1             ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                                        ;ADVANCE TO NEXT TEST HANDLER
(1)                                        ;---------------------------
(1)
```

```
    (1)  006366  013716  001362          .ADVANCE:MOV    NEXT,(SP)       ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
    (1)  006372  005037  001364                   CLR     LOCK            ;RESET TIGHT LOOP ADDRESS
    (1)  006376  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)  006400  106302                 .SHIFT:   ASLB    R2              ;POINT TO THE NEXT LINE
    (1)  006402  032702  000020                   BIT     #BIT4,R2        ;HAVE WE PASSED ALL LINE POINTERS?
    (1)  006406  001402                           BEQ     1$              ;IF NOT, RETURN TO THE TEST
    (1)  006410  022626                           POP2SP                  ;REMOVE THE TRAP CALL FROM THE STACK
    (1)  006412  104400                           ADVANCE                 ;GO TO THE NEXT TEST
    (1)  006414  000002                 1$:       RTI                     ;RETURN TO THE PRESENT TEST
    (1)
```

```
(1)                                             ;LINE PARAMETER REGISTER SETUP ROUTINE
(1)
(1)  006416  010146                 .LPRSET:MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
(1)  006420  010246                         MOV     R2,-(SP)        ;SAVE CONTENTS OF R2
(1)  006422  013701  001370                 MOV     PAR,R1          ;MOVE DEFAULT PARAM. INTO R1
(1)  006426  012702  000001                 MOV     #1,R2   ;INIT. FOR LINE 1
(1)  006432  010177  173362         1$:     MOV     R1,@DZVLPR      ;LOAD PARAM. REGISTER
(1)  006436  005201                         INC     R1              ;SET R1 FOR NEXT LINE
(1)  006440  106302                         ASLB    R2              ;SET R2 FOR NEXT LINE
(1)  006442  032702  000020                 BIT     #BIT4,R2        ;ALL LINES DONE?
(1)  006446  001771                         BEQ     1$              ;IF NG LOAD NEXT LINE
(1)  006450  012602                         MOV     (SP)+,R2        ;RELOAD R2
(1)  006452  012601                         MOV     (SP)+,R1        ;RELOAD R1
(1)  006454  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ROUTINE TO ZERO DATA BUFFER
(1)
(1)  006456  010046                 .BUFSET:MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
(1)  006460  012700  001426                 MOV     #TDO,R0         ;SET R0 TO TOP OF BUFFER
(1)  006464  005020                 1$:     CLR     (R0)+           ;CLEAR BUFFER LOCATION
(1)  006466  022700  001446                 CMP     #STOP,R0        ;IS BUFFER ALL CLEARED
(1)  006472  001374                         BNE     1$              ;IF NOT CLEAR NEXT LOCATION
(1)  006474  012600                         MOV     (SP)+,R0        ;RELOAD R0
(1)  006476  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ERROR HANDLER
(1)                                             ;-------------
(1)
(1)  006500  004737  007126         $ERROR: JSR     PC,SERV.G       ;FIND OUT IF <^G> WAS HIT
(1)  006504  032777  010000  172572         BIT     #SW12,@SWR      ;BELL ON ERROR?
(1)  006512  001406                         BEQ     XBX             ;BR IF NO BELL
(1)  006514  105777  172574                 TSTB    @$TPS           ;TTY READY.
(1)  006520  100003                         BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
(1)  006522  112777  000207  172566         MOVB    #207,@$TPB      ;PUSH A BELL AT THE TTY.
(1)  006530  032777  020000  172546 XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
(1)  006536  001113                         BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
(1)  006540  021637  001262                 CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
(1)  006544  001404                         BEQ     1$              ;BR IF YES
(1)  006546  011637  001262                 MOV     (SP),$ERRPC     ;RECORD BEING HERE
(1)  006552  105037  001247                 CLRB    $ERFLG          ;PREPARE HEADER
(1)  006556  104407                 1$:     SAV05                   ;SAVE ALL PROC REGISTERS
(1)  006560  011605                         MOV     (SP),R5         ;GET THE PC OF ERROR
(1)  006562  162705  000002                 SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
(1)  006566  011504                         MOV     (R5),R4         ;GET ERROR INSTRUCTION
(1)  006570  110437  001260                 MOVB    R4,$ITEMB       ;COPY TEST NUMBER FOR APT HANDLING
(1)  006574  006304                         ASL     R4              ;MULT BY TWO
(1)  006576  061504                         ADD     (R5),R4         ;DOUBLE IT
(1)  006600  006304                         ASL     R4              ;MULT AGAIN
(1)  006602  042704  177001                 BIC     #177001,R4      ;CLEAR JUNK
(1)  006606  062704  015216                 ADD     #.ERRTAB,R4     ;GET POINTER
(1)  006612  012437  006736                 MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
(1)  006616  012437  006750                 MOV     (R4)+,DATAHD    ;GET DATA HEADRER
(1)  006622  011437  006762                 MOV     (R4),DATABP     ;GET DATA TABLE
(1)  006626  105737  001247                 TSTB    $ERFLG          ;TYPE HEADER
(1)  006632  001403                         BEQ     TYPMSG          ;BR IF YES
(1)  006634  005737  006762                 TST     DATABP          ;DOES DATA TABLE EXIST?
```

```
(1) 006640 001044                        BNE     TYPDAT          ;BR IF YES.
(1) 006642 104402 001357       TYPMSG:   TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN
(1) 006646 104402 001357                 TYPE    ,$CRLF          ;AND TYPE ANOTHER
(1) 006652 005737 001364                 TST     LOCK
(1) 006656 001402                        BEQ     1$
(1) 006660 104402 010041                 TYPE    ,MASTEK
(1) 006664 104402 010027       1$:       TYPE    ,MTSTN
(1) 006670 104412 007120                 CNVRT   ,XTSTN          ;SHOW IT
(1) 006674 104402 010121                 TYPE    ,MERRPC         ;TYPE PC.
(1) 006700 104412 007112                 CNVRT   ,ERTABO         ;SHOW IT
(1) 006704 104402 007771                 TYPE    ,MCSRX
(1) 006710 104412 004332                 CNVRT   ,XCSR
(1) 006714 104402 001357                 TYPE    ,$CRLF          ;GIVE A CR/LF
(1) 006720 112737 177777 001247          MOVB    #-1,SERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
(1) 006726 005737 006736                 TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
(1) 006732 001402                        BEQ     WTBS.FM         ;BR IF NO.
(1) 006734 104402                        TYPE                    ;TYPE
(1) 006736 000000              ERRMSG:   0                       ;    ERROR MESSAGE
(1) 006740                     WTBS.FM:                          ;
(1) 006740 005737 006750                 TST     DATAHD          ;DATA HEADER?
(1) 006744 001402                        BEQ     TYPDAT          ;BR IF NO
(1) 006746 104402                        TYPE                    ;TYPE
(1) 006750 000000              DATAHD:   0                       ;    DATA HEADER
(1) 006752 005737 006762       TYPDAT:   TST     DATABP          ;DATA TABLE?
(1) 006756 001402                        BEQ     RESREG          ;BR IF NO.
(1) 006760 104411                        CONVRT                  ;SHOW
(1) 006762 000000              DATABP:   0                       ;    DATA TABLE
(1) 006764 104410              RESREG:   RES05                   ;RESTORE PROC REGISTERS
(1) 006766 122737 000001 001140 HALTS:   CMPB    #APTENV,$ENV    ;IS APT RUNNING?
(1) 006774 001007                        BNE     15$             ;SKIP APT CALL IF NOT
(1) 006776 113737 001260 007010          MOVB    $ITEMB,5$       ;COPY ERROR NUMBER
(1) 007004 004737 005242                 JSR     PC,$ATY4        ;CALL APT SERVICE
(1) 007010 000000              5$:       .WORD   0               ;ERROR NUMBER STUCK HERE
(1) 007012 000777              10$:      BR      10$             ;LOCK UP HERE
(1) 007014 022737 004316 000042 15$:     CMP     #$ENDAD,@#42    ;CHECK TO SEE IF IN ACT-11 MODE
(1) 007022 001403                        BEQ     20$             ;IF SO, HANDLE ACCORDINGLY
(1) 007024 005777 172254                 TST     @SWR            ;HALT ON ERROR?
(1) 007030 100004                        BPL     EXITER          ;BR IF NO HALT ON ERROR
(1) 007032 016677 000002 172246 20$:     MOV     2(SP),@DISPLAY  ;SHOW ERROR PC IN DATA DISPLAY
(1) 007040 000000                        HALT                    ;HALT
(1) 007042 005237 001256       EXITER:   INC     $ERTTL          ;UPDATE ERROR COUNT
(1) 007046 004737 007126                 JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
(1) 007052 032777 000400 172224          BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
(1) 007060 001007                        BNE     1$              ;BR IF YES
(1) 007062 032777 002000 172214          BIT     #SW10,@SWR      ;GOTO NEXT TEST?
(1) 007070 001407                        BEQ     2$              ;BR IF NO
(1) 007072 013737 001362 001252          MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
(1) 007100 012706 001120       1$:       MOV     #STACK,SP       ;RESET SP
(1) 007104 000177 172142                 JMP     @$LPADR         ;GOTO SPECIFIED TEST
(1) 007110 000002              2$:       RTI                     ;RETURN
(1) 007112 000001              ERTABO:   1
(1) 007114     006     002               .BYTE   6,2
(1) 007116 001404                        $AVPC
(1) 007120 000001              XTSTN:    1
(1) 007122     002     002               .BYTE   2,2
(1) 007124 001246                        $TSTNM
```

```
(1)  007126  017746  172160          SERV.G: MOV     @$TKB,-(SP)     ;OTHERWISE, GET THE LAST CHARACTER TYPED
(1)  007132  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY(EIGHTH) BIT
(1)  007136  122726  000007                  CMPB    #7,(SP)+        ;IS IT ^G?
(1)  007142  001076                          BNE     6$              ;IF NOT, IGNORE INPUT
(1)  007144  032777  004000  172136          BIT     #4000,@$TKS     ;RX BUSY?
(1)  007152  001365                          BNE     SERV.G          ;BR IF YES
(1)          007154                  GETSWR= .                                       ;;GPA
(1)  007154  017737  172124  007362          MOV     @SWR,90$        ;SAVE (SWR).
(1)  007162  104402  007342          1$:     TYPE    ,89$            ;TYPE HEADER FOR OLD SWITCH REGISTER
(1)  007166  104412  007354                  CNVRT   ,88$            ;TYPE THE NUMBER ITSELF
(1)  007172  104402  007364                  TYPE    ,91$            ;AFTER HAVING CONVERTED IT TO ASCII
(1)  007176  105037  007370                  CLRB    92$             ;CLEAR SWR CHANGE FLAG
(1)  007202  005077  172076                  CLR     @SWR            ;CLEAR THE SOFTWARE SWITCH REGISTER
(1)  007206  105777  172076          3$:     TSTB    @$TKS           ;WAIT FOR DONE.
(1)  007212  100375                          BPL     3$              ;CONTINUE WAITING FOR IT
(1)  007214  017746  172072                  MOV     @$TKB,-(SP)     ;PUT THE CHARACTER ON THE STACK
(1)  007220  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY B.T
(1)  007224  122726  000015                  CMPB    #15,(SP)+       ;IS IT THE CARRIAGE RETURN CHAR?
(1)  007230  001433                          BEQ     4$              ;IF SO, GO PRINT CRLF
(1)  007232  105777  172056          2$:     TSTB    @$TPS           ;IS THE OUTPUT BUFFER AVAILABLE
(1)  007236  100375                          BPL     2$              ;IF NOT, WAIT FOR IT TO BE READY
(1)  007240  105237  007370                  INCB    92$             ;INDICATE THAT THE SWR WAS CHANGED
(1)  007244  014677  172046                  MOV     -(SP),@$TPB     ;PLACE THE CHARACTER THERE(ECHO BACK)
(1)  007250  000241                          CLC                     ;GET READY TO ROTATE
(1)  007252  006177  172026                  ROL     @SWR            ;MOVE THE EXISTING BITS OVER
(1)  007256  006177  172022                  ROL     @SWR            ;TO MAKE ROOM FOR THE INCOMING
(1)  007262  006177  172016                  ROL     @SWR            ;THREE BITS FROM THIS CHARACTER
(1)  007266  103735                          BCS     1$              ;ERROR
(1)  007270  022627  000060                  CMP     (SP)+,#60       ;IS IT LOWER THAN 0?
(1)  007274  002732                          BLT     1$              ;IF SO, GO ASK AGAIN
(1)  007276  026627  177776  000067          CMP     -2(SP),#67      ;IS IT HIGHER THAN 7?
(1)  007304  003326                          BGT     1$              ;IF SO, GO ASK AGAIN
(1)  007306  042746  177770                  BIC     #^C<7>,-(SP)    ;ISOLATE INFORMATION BITS
(1)  007312  052677  171766                  BIS     (SP)+,@SWR      ;ADD THEM TO THE SWITCH REGISTER
(1)  007316  000733                          BR      3$              ;GO CHECK FOR THE NEXT CHARACTER
(1)  007320  105737  007370          4$:     TSTB    92$             ;HAS THE SWR BEEN CHANGED?
(1)  007324  001003                          BNE     5$              ;IF YES GO TYPE CRLF
(1)  007326  013777  007362  171750          MOV     90$,@SWR        ;IF NOT RESTORE SWR
(1)  007334  104402  001357          5$:     TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
(1)  007340  000207                  6$:     RTS     PC              ;RETURN TO CALLING PROCEDURE
(1)
(1)  007342  020200  051450  051127  89$:     .ASCIZ  <200>? (SWR)=/?
(1)                                  .EVEN
(1)  007354  000001                  88$:     1
(1)  007356     006     000                  .BYTE   6,0
(1)  007360  007362                          90$
(1)  007362  000000                  90$:     .WORD   0
(1)  007364  036457  000057          91$:     .ASCIZ  ?/=/?
(1)  007370     000                  92$:     .BYTE   0
(1)          007372                  .EVEN
(2)                                  .SBTTL  POWER DOWN AND UP ROUTINES
(2)
(2)                                  ;;******************************************************************************
(2)                                  ;POWER DOWN ROUTINE
(2)  007372  012737  007536  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(2)  007400  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
```

```
(4)  007406  010046                        MOV     R0,-(SP)          ;;PUSH R0 ON STACK
(4)  007410  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
(4)  007412  010246                        MOV     R2,-(SP)          ;;PUSH R2 ON STACK
(4)  007414  010346                        MOV     R3,-(SP)          ;;PUSH R3 ON STACK
(4)  007416  010446                        MOV     R4,-(SP)          ;;PUSH R4 ON STACK
(4)  007420  010546                        MOV     R5,-(SP)          ;;PUSH R5 ON STACK
(4)  007422  017746  171656                MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
(2)  007426  010637  007542                MOV     SP,$SAVR6         ;;SAVE SP
(2)  007432  012737  007444  000024        MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
(2)  007440  000000                        HALT
(2)  007442  000776                        BR      .-2               ;;HANG UP
(2)
(3)                                        ;;*****************************************************************
(2)                                        ;POWER UP ROUTINE
(2)  007444  012737  007536  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
(2)  007452  013706  007542                MOV     $SAVR6,SP         ;;GET SP
(2)  007456  005037  007542                CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
(2)  007462  005237  007542          1$:   INC     $SAVR6            ;;WAIT FOR THE INC
(2)  007466  001375                        BNE     1$                ;;OF  WORD
(4)  007470  012677  171610                MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
(4)  007474  012605                        MOV     (SP)+,R5          ;;POP STACK INTO R5
(4)  007476  012604                        MOV     (SP)+,R4          ;;POP STACK INTO R4
(4)  007500  012603                        MOV     (SP)+,R3          ;;POP STACK INTO R3
(4)  007502  012602                        MOV     (SP)+,R2          ;;POP STACK INTO R2
(4)  007504  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
(4)  007506  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
(2)  007510  012737  007372  000024        MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
(2)  007516  012737  000340  000026        MOV     #340,@#PWRVEC+2   ;;PRIO:7
(2)  007524  104402                        TYPE                      ;;REPORT THE POWER FAILURE
(2)  007526  007544                $PWRMG: .WORD   MPFAIL            ;;POWER FAIL MESSAGE POINTER
(2)  007530  012716                        MOV     (PC)+,(SP)        ;;RESTART AT RESTART
(2)  007532  011070                $PWRAD: .WORD   RESTART           ;;RESTART ADDRESS
(2)  007534  000002                        RTI
(2)  007536  000000                $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
(2)  007540  000776                        BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
(2)  007542  000000                $SAVR6: 0                         ;;PUT THE SP HERE
(2)  007544  050200  051127  043040  MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT LAST TEST /
(2)  007607     200  047105  020104  MEPASS: .ASCIZ  <200>/END PASS CVDZB-B  /
(2)  007633     200  052522  047116  MR:     .ASCIZ  <200>/RUNNING   /
(2)  007647     200  051120  043517  MERR2:  .ASCIZ  <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2)  007716  044600  051516  043125  MERR3:  .ASCIZ  <200>/INSUFFICIENT DATA!/
(2)  007742  046200  041517  020113  MLOCK:  .ASCIZ  <200>/LOCK ON SELECTED TEST/
(2)  007771     103  051123  020072  MCSRX:  .ASCIZ   /CSR: /
(2)  007777     126  041505  020072  MVECX:  .ASCIZ  /VEC: /
(2)  010005     120  051501  042523  MPASSX: .ASCIZ  /PASSES: /
(2)  010016  051105  047522  051522  MERRX:  .ASCIZ  /ERRORS: /
(2)  010027     124  051505  020124  MTSTN:  .ASCIZ  /TEST NO: /
(2)  010041     052  000040          MASTEK: .ASCIZ  /* /
(2)  010044  052200  050131  020105  MNEW:   .ASCIZ  <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE:  /
(2)  010121     120  035103  000040  MERRPC: .ASCIZ  /PC: /
(2)  010126  046600  050101  047440  XHEAD:  .ASCIZ  <200>/MAP OF DZV11 STATUS/<200>
(2)  010154  044600  046114  043505  MBADLN: .ASCIZ  <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)                                        .EVEN
(2)  010216  000002                        XSTATQ: 2
(2)  010220     006     003                .BYTE   6,3
(2)  010222  001344                        $TMP1
```

```
(2)  010224  006     002                    .BYTE   6,2
(2)  010226  001346                         $TMP2
(1)                                  .EVEN
(2)                                  ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
(2)                                  ;-------------------------------------------------------------
(2)                                  ;
(2)                                  ;E=EXTERNAL LOOP BACK
(2)                                  ;I=INTERNAL LOOP BACK
(2)                                  ;S=STAGGERED LOOP BACK
(2)  010230  017605  000000          .SETFLG:MOV     @(SP),R5        ;PICK UP ADDRESS OF TAG
(2)  010234  042737  000040  010424          BIC     #40,INBUF       ;STRIP LOWER CASE
(2)  010242  122737  000105  010424          CMPB    #'E,INBUF       ;IS IT EXTERNAL LOOP BACK ?
(2)  010250  001005                          BNE     4$              ;NO
(2)  010252  013715  010342                  MOV     1$,(R5)         ;YES STORE INFO
(2)  010256  105037  001424                  CLRB    MNTFLG          ;SET MAINT BIT =0
(2)  010262  000422                          BR      7$              ;GET OUT
(2)  010264  122737  000111  010424  4$:     CMPB    #'I,INBUF       ;IS IT INTERNAL LOOP BACK ?
(2)  010272  001006                          BNE     5$              ;NO
(2)  010274  013715  010344                  MOV     2$,(R5)         ;YES STORE INFO
(2)  010300  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;SET UP THE MAINTENANCE FLAG LOADER
(2)  010306  000410                          BR      7$              ;GET OUT
(2)  010310  122737  000123  010424  5$:     CMPB    #'S,INBUF       ;IS IT STAGGERED LOOP BACK ?
(2)  010316  001007                          BNE     6$              ;WHAT ?
(2)  010320  013715  010346                  MOV     3$,(R5)         ;YES STORE INFO
(2)  010324  105037  001424                  CLRB    MNTFLG          ;ZERO BITS
(2)  010330  062716  000002          7$:     ADD     #2,(SP)         ;POP AROUND
(2)  010334  000002                          RTI
(2)  010336  104404                  6$:     INSTER                  ;RETRY
(2)  010340  000733                          BR      .SETFLG         ;DITTO
(2)  010342  000200                  1$:     .WORD   200             ;EXTERNAL =  E
(2)  010344  000000                  2$:     .WORD   0               ;INTERNAL = I
(2)  010346  100000                  3$:     .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)  010350  017605  000000          .PAWCH:MOV     @(SP),R5
(2)  010354  142737  000040  010424          BICB    #40,INBUF       ;SET FOR LOWER CASE INPUT
(2)  010362  122737  000105  010424          CMPB    #'E,INBUF       ;IS IT 'E' ?
(2)  010370  001002                          BNE     1$
(2)  010372  105015                          CLRB    (R5)            ;000
(2)  010374  000406                          BR      2$
(2)  010376  122737  000103  010424  1$:     CMPB    #'C,INBUF       ;IS IT "C" ?
(2)  010404  001005                          BNE     3$
(2)  010406  112715  177777                  MOVB    #-1,(R5)        ;3177
(2)  010412  062716  000002          2$:     ADD     #2,(SP)
(2)  010416  000002                          RTI
(2)  010420  104404                  3$:     INSTER                  ;RETRY
(2)  010422  000752                          BR      .PAWCH
(2)
(2)                                          ;BUFFERS FOR INPUT-OUTPUT
(2)
(2)  010424  000000                  INBUF:  0
(2)          010466                  .=.+40
(2)                                  ; TEMP: 0               ; TEMP AREA UNUSED.                     ;;GPA
(2)                                  ; .=.+40                ; DELETED TO CONSERVE SPACE.            ;;GPA
(2)  010466  000000                  MDATA:  0
(2)          010530                  .=.+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)  010530  005737  001406          CYCLE:  TST     DZVACTV         ;ARE ANY DZV11'S TO BE TESTED?
(2)  010534  001004                          BNE     1$              ;BR IF OK.
(2)  010536  104402  007647                  TYPE    ,MERR2          ;NO DZV11'S SELECTED!!
(2)  010542  000000                          HALT                    ;STOP THE SHOW.
(2)  010544  000776                          BR      .-2             ;DISQUALIFY CONT. SW.
(2)  010546  013737  004624  001354  1$:     MOV     $MXCNT,$TIMES   ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
(2)  010554  033737  001412  001406          BIT     RUN,DZVACTV     ;IS THIS ONE "ACTIVE"
(2)  010562  001017                          BNE     2$              ;BR IF GOOD ONE FOUND.
(2)  010564  006137  001412                  ROL     RUN             ;UPDATE POINTER
(2)  010570  005537  001412                  ADC     RUN             ;CATCH CARRY FROM RUN
(2)  010574  062737  000012  001420          ADD     #12,ACTIVE      ;UPDATE ADDRESS POINTER.
(2)  010602  022737  001740  001420          CMP     #DZV.END,ACTIVE ;HAVE WE PASSED THE END OF THE MAP?
(2)  010610  001356                          BNE     1$              ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
(2)  010612  012737  001500  001420          MOV     #DZV.MAP,ACTIVE ;RESET ADDRESS POINTER.
(2)  010620  000752                          BR      1$              ;KEEP LOOKING FOR ACTIVE DZV11
(2)  010622  006137  001412          2$:     ROL     RUN             ;UPDATE POINTER.
(2)  010626  005537  001412                  ADC     RUN             ;CATCH CARRY.
(2)  010632  013700  001420                  MOV     ACTIVE,R0       ;GET ADDRESS POINTER.
(2)  010636  062737  000012  001420          ADD     #12,ACTIVE      ;UPDATE.
(2)  010644  022737  001740  001420          CMP     #DZV.END,ACTIVE
(2)                                                                  ;ALL DONE?
(2)  010652  001003                          BNE     3$              ;BR IF NO.
(2)  010654  012737  001500  001420          MOV     #DZV.MAP,ACTIVE ;RESTORE POINTER.
(2)  010662  012037  001174          3$:     MOV     (R0)+,$BASE     ;LOAD SYSTEM CTRL. REG
(2)  010666  012037  002040                  MOV     (R0)+,DZVRIV    ;LOAD VECTOR
(2)  010672  012037  001366                  MOV     (R0)+,LINE      ;SET UP DZV LINES ACTIVE
(2)  010676  012037  001370                  MOV     (R0)+,PAR       ;SET UP PARAMETERIZATION
(2)  010702  012037  001372                  MOV     (R0)+,MODE      ;SET UP MAINTENANCE MODE
(2)  010706  105037  001424                  CLRB    MNTFLG  ;RESET MAINT. FLAG IF
(2)  010712  005737  001372                  TST     MODE            ;RUNNING TESTS
(2)  010716  001003                          BNE     9$              ;IN
(2)  010720  112737  000010  001424          MOVB    #MAINT,MNTFLG   ;INTERNAL MAINT. MODE
(2)  010726  004737  011074          9$:     JSR     PC,DZVLEV       ;SET UP
(2)  010732  005737  000042                  TST     @#42            ;ARE WE UNDER MONITOR CONTROL?
(2)  010736  001051                          BNE     7$              ;IF YES, SKIP THIS SETUP
(2)  010740  032777  000002  170336          BIT     #SW01,@SWR      ;IF SW01=1, GET STARTING TEST #
(2)  010746  001445                          BEQ     7$              ;BR IF NO TEST IS TO BE INPUTTED
(2)  010750  104402  001357          4$:     TYPE    ,$CRLF
(3)  010754  104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(3)  010756  010027                          MTSTN                   ;POINTER TO MESSAGE TO BE PRINTED
(3)  010760  104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(3)  010762  000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  010764  001000                          1000                    ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  010766  001246                          $TSTNM                  ;POINTER TO MAP LOCATION TO BE FILLED
(3)  010770     000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(3)  010771     001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(2)  010772  012700  012070                  MOV     #TST1,R0
```

```
(2)  011776  022710  000004          5$:     CMP     #4,(R0)
(2)  012002  001020                          BNE     6$
(2)  012004  022760  012737  000002          CMP     #12737,2(R0)
(2)  012012  001014                          BNE     6$
(2)  012014  023760  001246  000004          CMP     $TSTNM,4(R0)    ;IS THIS THE TEST ?
(2)  012022  001010                          BNE     6$              ;IF NOT, DON'T PROCESS NUMBER
(2)  012024  010037  001252                  MOV     R0,$LPADR       ;SAVE PC
(2)  012030  062737  000002  001252          ADD     #2,$LPADR       ;POP OVER PREVIOUS SCOPE
(2)  012036  104402  001357                  TYPE    ,$CRLF
(2)  012042  000412                          BR      8$
(2)  012044  005720                  6$:     TST     (R0)+
(2)  012046  020027  014222                  CMP     R0,#TLAST+10
(2)  012052  001351                          BNE     5$
(2)  012054  104402  001356                  TYPE    ,$QUES
(2)  012060  000733                          BR      4$
(2)  012062  012737  012070  001252  7$:     MOV     #TST1,$LPADR    ;PREPARE TEST ADDRESS
(2)  012070                          8$:
(2)  012070  000177  170156          RESTART:JMP     @$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)  012074  013700  002040          DZVLEV: MOV     DZVRIV,R0       ;PLACE THE BASE VECTOR ADDRESS IN R0
(2)  012100  062700  000002                  ADD     #2,R0           ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
(2)  012104  010037  002042                  MOV     R0,DZVRIS       ;STORE IT HERE
(2)  012110  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
(2)  012114  010037  002044                  MOV     R0,DZVTIV       ;STORE IT HERE
(2)  012120  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
(2)  012124  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)  012130  013700  001174                  MOV     $BASE,R0        ;COPY THE ADDRESS BEING LOADED
(2)  012134  010037  002010                  MOV     R0,DZVCSR       ;XXX0
(2)  012140  005200                          INC     R0
(2)  012142  010037  002012                  MOV     R0,HDZVCSR      ;XXX1
(2)  012146  005200                          INC     R0
(2)  012150  010037  002014                  MOV     R0,DZVRBUF      ;XXX2
(2)  012154  010037  002020                  MOV     R0,DZVLPR       ;XXX2
(2)  012160  005200                          INC     R0
(2)  012162  010037  002016                  MOV     R0,HDZVRBUF     ;XXX3
(2)  012166  010037  002022                  MOV     R0,HDZVLPR      ;XXX3
(2)  012172  005200                          INC     R0
(2)  012174  010037  002024                  MOV     R0,DZVTCR       ;XXX4
(2)  012200  005200                          INC     R0
(2)  012202  010037  002026                  MOV     R0,HDZVTCR      ;XXX5
(2)  012206  005200                          INC     R0
(2)  012210  010037  002030                  MOV     R0,DZVMSR       ;XXX6
(2)  012214  010037  002034                  MOV     R0,DZVTDR       ;XXX6
(2)  012220  005200                          INC     R0
(2)  012222  010037  002032                  MOV     R0,HDZVMSR      ;XXX7
(2)  012226  010037  002036                  MOV     R0,HDZVTDR      ;XXX7
(2)  012232  000207                          RTS     PC
```

```
(2)                                     ;CONVERT DECIMAL ASCII STRING TO OCTAL
(2)  011234  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
```

POWER DOWN AND UP ROUTINES

```
(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)  011236  012700  001500          SETAPT: MOV     #DZV.MAP,R0     ;POINT TO THE DEVICE MAP TABLE
(2)  011242  013701  001174                  MOV     $BASE,R1        ;BUILD DEVICE ADDRESSES IN R1
(2)  011246  013702  001170                  MOV     $VECT1,R2       ;BUILD DEVICE VECTORS IN R2
(2)  011252  042702  177007                  BIC     #^C<770>,R2     ;STRIP AWAY OTHER INFORMATION
(2)  011256  012704  001204                  MOV     #$DDW0,R4       ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
(2)  011262  013705  001176                  MOV     $DEVM,R5        ;GET THE MAP OF ACTIVE DEVICES
(2)  011266  105037  001414                  CLRB    DZVNUM          ;INITIALIZE THE NO. OF ACTIVE DEVICES
(2)  011272  005037  001410                  CLR     SAVACTV         ;CLEAR THE ACTIVE BIT MAP
(2)  011276  006005                  1$:     ROR     R5              ;GET A DEVICE SELECTION BIT
(2)  011300  103407                          BCS     3$              ;IF IT IS SELECTED, GO SET UP A MAP
(2)  011302  001422                          BEQ     5$              ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
(2)  011304  005724                          TST     (R4)+           ;POINT TO NEXT DEVICE DESCRIPTOR
(2)  011306  062701  000010          2$:     ADD     #10,R1          ;SET UP THE NEXT ADDRESS
(2)  011312  062702  000010                  ADD     #10,R2          ;SET UP THE NEXT VECTOR GROUP
(2)  011316  000767                          BR      1$              ;GO SEE IF MORE DEVICES REMAIN
(2)  011320  006137  001410          3$:     ROL     SAVACTV         ;SET BIT IN ACTIVE DEVICE MAP
(2)  011324  105237  001414                  INCB    DZVNUM          ;INCREMENT NO. OF ACTIVE DEVICES
(2)  011330  010120                          MOV     R1,(R0)+        ;LOAD DEVICE ADDRESS
(2)  011332  010220                          MOV     R2,(R0)+        ;LOAD THE VECTOR ADDRESS
(2)  011334  013720  001200                  MOV     $CDW1,(R0)+     ;GET THE NUMBER OF LINES IN OPERATION
(2)  011340  012420                          MOV     (R4)+,(R0)+     ;LOAD DEVICE PARAMETERS
(2)  011342  013720  001202                  MOV     $CDW2,(R0)+     ;LOAD DEFAULT TESTING MODE
(2)  011346  000757                          BR      2$              ;GO BUILD THE NEXT ADDRESS
(2)  011350  012710  177777          5$:     MOV     #-1,(R0)        ;TERMINATE THE DEVICE MAP
(2)  011354  012737  001142  001304          MOV     #$SWREG,SWR     ;SET TO SOFTWARE APT SWITCH REGISTER
(2)  011362  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)  011364                          AUTO.SIZE:
(2)  011364  000005                          RESET                   ;INSURE A BUS INIT.
(2)  011366  105337  001422                  DECB    INIFLG          ;SHOW THAT I WAS HERE
(2)  011372  012702  001500          CSRMAP: MOV     #DZV.MAP,R2     ;LOAD MAP POINTER.
(2)  011376  012703  001204                  MOV     #$DDW0,R3       ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
(2)  011402  005022                  1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
(2)  011404  022702  001740                  CMP     #DZV.END,R2     ;ALL DONE?
(2)  011410  001374                          BNE     1$              ;BR IF NO
(2)  011412  105037  001414                  CLRB    DZVNUM          ;SET OCTAL NUMBER OF DZV11'S TO 0
(2)  011416  012702  001500                  MOV     #DZV.MAP,R2
(2)  011422  012701  160000                  MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
(2)  011426  012737  011672  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTENT DEVICE TIME OUT
(2)  011434  052711  000040          2$:     BIS     #BIT5,(R1)      ;TRY TO SET MASTER SCAN ENABLE
(2)  011440  052761  000017  000004          BIS     #17,4(R1)       ;TRY TO TRANSMIT ON ANY LINE
(2)  011446  005000                          CLR     R0              ;USE R0 AS A COUNTER
(2)  011450  005711                  7$:     TST     (R1)            ;HAS TRANSMITTER READY COME UP?
```

```
(2)  011452  100403                          BMI     8$              ;IF SO, GO GET A FINAL CHECK
(2)  011454  005300                          DEC     R0              ;REDUCE COUNT. TIME UP?
(2)  011456  001374                          BNE     7$              ;IF NOT, KEEP WAITING
(2)  011460  000437                          BR      3$              ;ASSUME IT'S NOT A DZV11
(2)  011462  032761  000017  000004  8$:     BIT     #17,4(R1)       ;ARE ANY TCR BITS  STILL SET? THEY SHOULD BE
(2)  011470  001433                          BEQ     3$              ;IF IT'S NOT, ASSUME IT'S NOT A DZV11
(2)  011472  032711  000040                  BIT     #BIT5,(R1)      ;IS MASTER SCAN ENABLE STILL SET?
(2)  011476  001430                          BEQ     3$              ;IF NOT, ASSUME IT'S NOT A DZV11
(2)  011500  052711  000020                  BIS     #20,(R1)        ;SET DEVICE CLEAR
(2)  011504  000240                          NOP
(2)  011506  032711  000040                  BIT     #40,(R1)        ;DID SCANNER CLEAR
(2)  011512  001022                          BNE     3$              ;IF NOT ASSUME IT IS NOT DZV
(2)  011514  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)  011520  010122                          MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
(2)  011522  005722                          TST     (R2)+           ;POP OVER VECTOR STORE AREA
(2)  011524  012722  000017                  MOV     #17,(R2)+       ;SET THE DEFAULT LINE SELECTION PARAMETER
(2)  011530  012712  017470                  MOV     #17470,(R2)     ;SET THE DEFAULT PARAMETERS
(2)  011534  012223                          MOV     (R2)+,(R3)+     ;COPY PARAMETERS INTO ETABLE DESCRIPTOR
(2)  011536  005022                          CLR     (R2)+           ;SET THE DEFAULT MODE OF OPERATION
(2)  011540  012712  177777                  MOV     #-1,(R2)        ;TERMINATE LIST
(2)  011544  105237  001414                  INCB    DZVNUM          ;UPDATE DEVICE COUNTER
(2)  011550  122737  000020  001414          CMPB    #20,DZVNUM      ;ARE MAX. NO. OF DEV FOUND?
(2)  011556  001405                          BEQ     100$            ;YES DON'T LOOK FOR ANY MORE.
(2)  011560  062701  000010          3$:     ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
(2)  011564  022701  164000                  CMP     #164000,R1
(2)  011570  001321                          BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
(2)  011572                        100$:
(2)  011572  105737  001414                  TSTB    DZVNUM          ;WERE ANY DZV11'S FOUND AT ALL?
(2)  011576  001430                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
(2)  011600  113701  001414                  MOVB    DZVNUM,R1
(2)  011604  012737  000001  001410          MOV     #1,SAVACTV      ;CREATE A BIT MAP OF
(2)  011612  005301                  4$:     DEC     R1              ;THE DEVICES IN THE SYSTEM
(2)  011614  001404                          BEQ     98$
(2)  011616  000261                          SEC
(2)  011620  006137  001410                  ROL     SAVACTV
(2)  011624  000772                          BR      4$
(2)  011626  013737  001500  001174  98$:     MOV     DZCR0,$BASE     ;POINT TO THE ADDRESS OF FIRST DEVICE
(2)  011634  013737  001510  001202          MOV     MAN10,$CDW2     ;INDICATE TO ETABLE WHAT MODE IS BEING USED
(2)  011642  012737  000006  000004  99$:     MOV     #6,a#4          ;RESTORE TRAP VECTOR
(2)  011650  013737  001410  001176          MOV     SAVACTV,$DEVM   ;SAVE ACTIVE REGISTER
(2)  011656  000410                          BR      VECMAP          ;GO FIND THE VECTOR NOW.
(2)  011660  104402  007647          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO DZV11'S FOUND.
(2)  011664  005000                          CLR     R0              ;MAKE DATA DISPLAY ZERO
(2)  011666  000000                          HALT                    ;STOP THE SHOW
(2)  011670  000776                          BR      .-2             ;DISABLE CONT. SW.
(2)  011672  012716  011560          6$:     MOV     #3$,(SP)        ;ENTERED BY NON-EXISTENT TIME-OUT
(2)  011676  000002                          RTI                     ;RETURN TO MAINSTREAM
(2)
(2)  011700  012737  000200  000022  VECMAP: MOV     #MASK,a#22      ;SET IOT TRAP PRIORITY
(2)  011706  012737  012022  000020          MOV     #4$,a#20        ;SET IOT TRAP VECTOR
(2)  011714  012702  001500                  MOV     #DZV.MAP,R2     ;SET SOFTWARE POINTER
(2)  011720  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
(2)  011724  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
(2)  011730  010120                  1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
(2)  011732  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
```

```
(2)  011736  022021                        CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
(2)  011740  020127  001000                CMP     R1,#1000        ;HAS THE VECTOR AREA BEEN EXCEEDED?
(2)  011744  101771                        BLOS    1$              ;BR IF MORE TO FILL
(2)  011746  013704  001410                MOV     SAVACTV,R4      ;STORE TEMPORARILY
(2)  011752  006004                 2$:    ROR     R4              ;BRING OUT A BIT
(2)  011754  103036                        BCC     5$              ;BR IF ALL DONE
(2)  011756  106427  000000                MTPS    #0              ;ZERO CPU PRIO
(2)  011762  012772  040040  000000        MOV     #BIT14+BIT5,@(R2)       ;SET TIE AND MAS SCAN
(2)  011770  011201                        MOV     (R2),R1         ;GET CSR
(2)  011772  112761  000017  000004        MOVB    #17,4(R1)       ;SET THE TCR BITS FOR ALL LINES
(2)                                                                 ;ATTEMPT TO FORCE AN INTERRUPT
(2)  012000  005200                        INC     R0              ;STALL
(2)  012002  001376                        BNE     .-2             ;       FOR TIME TO INTERRUPT
(2)  012004  012762  000300  000002        MOV     #300,2(R2)      ;NO INTERRUPT ASSUME 300 AND FIX DZV11 LATER
(2)  012012  000005                        RESET                   ;INIT
(2)  012014  062702  000012          3$:   ADD     #12,R2          ;POP SOFTWARE POINTER
(2)  012020  000754                        BR      2$              ;KEEP GOING
(2)  012022  011662  000002          4$:   MOV     (SP),2(R2)      ;GET VECTOR ADDRESS
(2)  012026  162762  000010  000002        SUB     #10,2(R2)       ;POINT BACK TO THE CORRECT VECTOR
(2)  012034  042762  000007  000002        BIC     #7,2(R2)        ;CLEAR JUNK
(2)  012042  022626                        POP2SP                  ;POP IOT JUNK OFF STACK
(2)  012044  012716  012014                MOV     #3$,(SP)        ;SET FOR RETURN
(2)  012050  000002                        RTI
(2)  012052  013737  001502  001170  5$:   MOV     DZVC0,$VECT1    ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
(2)  012060  012737  004362  000020        MOV     #.SCOPE,IOTVEC  ;RESTORE THE SCOPE TRAP
(2)  012066  000207                        RTS     PC              ;ALL DONE WITH "AUTO SIZING"
(2)
```

```
 2734
 2735                                        ;************************* TEST 1 *********************************
 2736                                        ;*THIS TEST VERIFIES OVERRUN AND SILO ALARM
 2737                                        ;*ONE LINE AT A TIME  - BASED UPON VALID LINES
 2738                                        ;*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
 2739                                        ;*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
 2740                                        ;*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
 2741                                        ;*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
 2742                                        ;*CHAR PULLED OUT OF THE SILO.
 2743                                        ;*ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
 2744                                        ;*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
 2745                                        ;*ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
 2746                                        ;*USED TO SCOPE SILO ALARM PULSES, ETC.
 2748                                  ;:*  TEST 1
  (5)                                  ;;*****************************************************************
  (4)  012070  000004                  TST1:  SCOPE
  (2)  012072  012737  000001  001246         MOV    #1,$TSTNM          ;LOAD THE NUMBER OF THIS TEST
  (2)  012100  012737  012532  001362         MOV    #TST2,NEXT         ;POINT TO THE START OF THE NEXT TEST
 2749  012106  012737  012446  001364         MOV    #18$,LOCK          ;SET FOR LOOP
 2750  012114  104417                        DCLASM                    ;SET DCLR IN CSR AND SET MNTFLG
 2751  012116  104421                        LPRSET                    ;LOAD LINE PARAMETERS
 2752  012120  005037  001374               CLR    SAVLIN             ;INIT LINE INDICATOR
 2753  012124  104422                        BUFSET                    ;ZERO DATA BUFFER
 2754  012126  012702  000001               MOV    #1,R2              ;LINE POINTER
 2755  012132  052777  010040  167650       BIS    #MSENAB!SILOEN,@DZVCSR  ;START SCANNER & SET SILO ENABLE
 2756  012140  030237  001366         3$:   BIT    R2,LINE            ;VALID LINE?
 2757  012144  001533                        BEQ    21$                ;IF NOT GO TO NEXT LINE
 2758  012146  013700  001374               MOV    SAVLIN,R0          ;MAKE OFFSET
 2759  012152  006300                        ASL    R0                 ;MAKE POWER OF TWO
 2760  012154  010277  167644               MOV    R2,@DZVTCR         ;SET TCR BIT
 2761  012160  105777  167624         4$:   TSTB   @DZVCSR            ;REC DONE = 1 ?
 2762  012164  100001                        BPL    .+4
 2763  012166  104020                        ERROR  20                 ;REC DONE SHOULD NOT = 1
 2764  012170  005003                        CLR    R3                 ;SET CHARACTER COUNT
 2765  012172  005004               5$:     CLR    R4
 2766  012174  005777  167610         6$:   TST    @DZVCSR            ;IS TRDY SET?
 2767  012200  100404                        BMI    7$                 ;IF YES, LOAD CHAR.
 2768  012202  104414                        DELAY                     ;WAIT FOR TRDY TO SET
 2769  012204  005204                        INC    R4                 ;INC DELAY COUNTER
 2770  012206  001372                        BNE    6$
 2771  012210  104003                        ERROR  3                  ;*TRDY FAILED TO SET
 2772  012212  116077  001426  167614  7$:   MOVB   TDO(R0),@DZVTDR    ;LOAD A CHARACTER
 2773  012220  005260  001426               INC    TDO(R0)            ;SET UP NEXT CHARACTER
 2774  012224  020327  000017               CMP    R3,#15.            ;16 CHARACTERS ?
 2775  012230  103006                        BHIS   8$
 2776  012232  032777  020000  167550       BIT    #SILOAL,@DZVCSR    ;SILO ALARM = 0 ?
 2777  012240  001413                        BEQ    10$                ;YES
 2778  012242  104013                        ERROR  13                 ;*SILO ALARM SHOULD NOT = 1
 2779                                                                  ;UNTIL 16. DATA CHARACTERS
 2780  012244  000411                        BR     10$
 2781  012246  005004               8$:     CLR    R4
 2782  012250  032777  020000  167532  9$:   BIT    #SILOAL,@DZVCSR
 2783  012256  001004                        BNE    10$
 2784  012260  104414                        DELAY
 2785  012262  005204                        INC    R4
 2786  012264  001371                        BNE    9$
```

```
                                                ERROR   14              ;*SILO ALARM FAILED TO SET!
                                                                        ;SILO ALARM SHOULD =1 AFTER 16.
                                                                        ;DATA CHARACTERS
                                        10$:    INC     R3              ;INC CHAR COUNT
012272  022703  000102                        CMP     #66.,R3         ;FINISHED SENDING CHARACTERS ?
012276  001335                                BNE     5$              ;NO
012300  005004                                CLR     R4
012302  104414                                DELAY
012304  105204                                INCB    R4
012306  001375                                BNE     .-4
                                                ;NOW LETS READ THE SILO
012310  013705  001374                        MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
012314  005737  001372                        TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
012320  100006                                BPL     13$             ;IF NOT, SKIP STAGGERED SETUP
                                                ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
012322  006205                                ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
012324  103402                                BCS     11$             ;IF IT IS SET, GO CLEAR IT
012326  000261                                SEC                     ;IF IT IS CLEAR SET IT HERE
012330  000401                                BR      12$             ;SKIP THE CLEARING
012332  000241                        11$:    CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
012334  006105                        12$:    ROL     R5              ;GET THE NEW BIT BACK INTO R5
012336  000305                        13$:    SWAB    R5              ;PUT IN UPPER BYTE
012340  052705  100000                        BIS     #DVALID,R5      ;ADD DATA VALID
012344  017704  167444                14$:    MOV     @DZVRBUF,R4     ;ACTUAL
012350  020405                                CMP     R4,R5           ;ACTUAL VS. EXPECTED
012352  001401                                BEQ     15$             ;YES
012354  104006                                ERROR   6               ;*DATA/CONTENTS DID NOT COMPARE
012356  032777  020000  167424        15$:    BIT     #SILOAL,@DZVCSR ;SILO ALARM= 0 ?
012364  001401                                BEQ     16$             ;YES
012366  104016                                ERROR   16              ;READING DZVRBUF DID NOT CLEAR SILO ALARM
012370  005205                        16$:    INC     R5              ;UP CHARACTER
012372  120527  000077                        CMPB    R5,#63.         ;LAST SILO CHAR ?....64TH CHAR
012376  101762                                BLOS    14$
012400  005205                                INC     R5              ;ADD 1 MORE FOR THE CLOBBERED CHAR
012402  052705  040000                        BIS     #OVRRUN,R5      ;ADD OVERRUN TO EXPECTED
012406  120527  000101                        CMPB    R5,#65.         ;LAST CHARACTER ?
012412  001754                                BEQ     14$
012414  017704  167374                        MOV     @DZVRBUF,R4     ;FOR GOOD MEASURE
012420  005704                                TST     R4              ;DATA VALID SHOULD = 0
012422  100001                                BPL     17$             ;YES
012424  104017                                ERROR   17              ;DATA VALID SHOULD = 0
012426  040277  167372                17$:    BIC     R2,@DZVTCR      ;CLR TCR BIT
012432  104401                                SCOP1                   ;LOOP?
012434  005237  001374                21$:    INC     SAVLIN          ;INC EXPECTED LINE
012440  104420                                SHIFT                   ;NEXT LINE
012442  000137  012140                        JMP     3$              ;YES

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

012446  052777  010040  167334  18$:    BIS     #MSENAB!SILOEN,@DZVCSR  ;SETUP DEVICE
012454  012777  012522  167362                MOV     #2V$,@DZVTIV    ;SETUP TRANSMITTER VECTOR
012462  012701  000024                        MOV     #20.,R1         ;TEMPORARY COUNT OF CHARACTER BURST
```

```
 2833  012466  050277  167332                BIS     R2,@DZVTCR      ;ENABLE LINE
 2834  012472  052777  040000  167310        BIS     #TIE,@DZVCSR    ;ENABLE INTERRUPTS
 2835  012500  106427  000000                MTPS    #0              ;LOWER PRIORITY
 2836  012504  000001                19$:    WAIT                    ;ALLOW INTERRUPTS
 2837  012506  077102                        SOB     R1,19$          ;RED  E COUNT. ALL CHARACTERS SENT?
 2838  012510  042777  050040  167272        BIC     #SILOEN!MSENAB!TIE,@DZVCSR ;RESET SILO COUNTER, CLEAR STROBE
 2839  012516  104401                        SCOP1                   ;LOOP AGAIN?
 2840  012520  000742                        BR      17$             ;IF NOT, RETURN TO WHERE YOU LEFT OFF
 2841  012522  112777  000252  167304  20$:  MOVB    #252,@DZVTDR    ;SEND A CHARACTER
 2842  012530  000002                        RTI                     ;ALLOW MORE CHARACTERS TO COME
 2843                                        ;************************* TEST 2 *********************************
 2844                                        ;*THIS TEST THAT "SILO ENABLE" WILL INHIBIT
 2845                                        ;*RECEIVER INTERRUPTS AND THAT ON THE
 2846                                        ;*16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
 2847                                        ;*INTERRUPT WITH "RIE" SET.
 2848                                        ;*THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
 2849                                        ;*ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
 2851                                ;:*  TEST 2
  (5)                               ;:*****************************************************************************
  (4)   012532  000004                  TST2:   SCOPE
  (2)   012534  012737  000002  001246          MOV     #2,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
  (2)   012542  012737  013032  001362          MOV     #TST3,NEXT      ;POINT TO THE START OF THE NEXT TEST
 2852   012550  012737  012574  001364          MOV     #3$,LOCK        ;SET FOR LOOP
 2853   012556  104417                        DCLASM                  ;SET DCLR IN CSR AND SET MNTFLG
 2854   012560  104421                        LPRSET                  ;LOAD LINE PARAMETERS
 2855   012562  005037  001374                CLR     SAVLIN          ;INIT LINE INDICATOR
 2856   012566  104422                        BUFSET                  ;ZERO DATA BUFFER
 2857   012570  012702  000001                MOV     #1,R2           ;LINE POINTER
 2858   012574  012777  013004  167236  3$:   MOV     #11$,@DZVRIV    ;SET FOR UNEXPECTED INTER.
 2859   012602  012777  000200  167232        MOV     #MASK,@DZVRIS   ;SET PRIO.
 2860   012610  052777  010140  167172        BIS     #MSENAB!SILOEN!RIE,@DZVCSR
 2861                                                                  ;START SCANNER & SET SILO ENABLE
 2862   012616  030237  001366                BIT     R2,LINE         ;VALID LINE?
 2863   012622  001477                        BEQ     18$             ;IF NOT GO TO NEXT LINE
 2864   012624  005777  167164                TST     @DZVRBUF        ;EMPTY THE SILO
 2865   012630  100775                        BMI     .-4             ;BR IF DATA VALID IS SET!
 2866   012632  106427  000000                MTPS    #0              ;SET PROCESSOR PRIORITY TO 0
 2867   012636  013700  001374                MOV     SAVLIN,R0       ;MAKE OFFSET
 2868   012642  006300                        ASL     R0              ;MAKE POWER OF TWO
 2869   012644  010277  167154                MOV     R2,@DZVTCR      ;SET TCR BIT
 2870   012650  005004                5$:     CLR     R4
 2871   012652  005777  167132        6$:     TST     @DZVCSR
 2872   012656  100404                        BMI     7$
 2873   012660  104414                        DELAY
 2874   012662  005204                        INC     R4
 2875   012664  001372                        BNE     6$
 2876   012666  104003                        ERROR   3               ;*TRDY FAILED TO SET
 2877   012670  116077  001426  167136  7$:   MOVB    TDO(R0),@DZVTDR ;LOAD A CHARACTER
 2878   012676  005260  001426                INC     TDO(R0)         ;SET UP NEXT CHARACTER
 2879   012702  022760  000017  001426        CMP     #15.,TDO(R0)    ;15 CHARS YET?
 2880   012710  001406                        BEQ     8$
 2881   012712  032777  020000  167070        BIT     #SILOAL,@DZVCSR ;SILO ALARM = 0 ?
 2882   012720  001401                        BEQ     .+4             ;YES
 2883   012722  104013                        ERROR   13              ;*SILO ALARM SHOULD NOT = 1
 2884                                                                  ;UNTIL 16. DATA CHARACTERS
 2885   012724  000752                        BR      6$
```

```
 2886  012726  012777  013012  167104  8$:     MOV     #12$,@DZVRIV    ;SET NEW VECTOR
 2887  012734  005777  167050                  TST     @DZVCSR         ;READY FOR 16TH CHAR
 2888  012740  100375                          BPL     .-4
 2889  012742  016077  001426  167064          MOV     TDO(R0),@DZVTDR ;LOAD THE 16TH CHAR.
 2890  012750  005004                          CLR     R4
 2891  012752  032777  020000  167030  9$:     BIT     #SILOAL,@DZVCSR
 2892  012760  001005                          BNE     10$
 2893  012762  104414                          DELAY
 2894  012764  005204                          INC     R4
 2895  012766  001371                          BNE     9$
 2896  012770  104014                          ERROR   14              ;*SILO ALARM FAILED TO SET!
 2897  012772  000410                          BR      17$             ;SILO ALARM SHOULD =1 AFTER 16.
 2898                                                                  ;DATA CHARACTERS
 2899  012774  000240                  10$:    NOP                     ;STALL
 2900  012776  000240                          NOP
 2901  013000  104027                          ERROR   27              ;SILO ALARM NOT INTERRUPTING.
 2902  013002  000404                          BR      17$             ;CONTINUE TEST.
 2903  013004  022626                  11$:    POP2SP                  ;FAKE RTI
 2904  013006  104012                          ERROR   12              ;RX SHOULD NOT INTERRUPT
 2905  013010  000401                          BR      17$             ;CONTINUE
 2906  013012  022626                  12$:    POP2SP                  ;GOOD INTERRUPT TO HERE.
 2907  013014  040277  167004          17$:    BIC     R2,@DZVTCR      ;CLR TCR BIT
 2908  013020  104401                          SCOP1                   ;LOOP?
 2909  013022  005237  001374          18$:    INC     SAVLIN          ;INC EXPECTED LINE
 2910  013026  104420                          SHIFT                   ;NEXT LINE
 2911  013030  000661                          BR      3$              ;YES
```

```
                                                ;************************** TEST 3 ********************************
                                                ;*THIS TEST RUNS ALL LINES FULL BORE
                                                ;*BASED UPON QUALIFIED LINES
                                                ;*..THIS IS AN INTERRUPT TEST ON THE RECEIVER AND
                                                ;*TRANSMITTER
                                        ;:*  TEST 3
                                        ;:*************************************************************************
                                        TST3:   SCOPE
013032  000004                                 MOV     #3,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
013034  012737  000003  001246                  MOV     #TST4,NEXT      ;POINT TO THE START OF THE NEXT TEST
013042  012737  013574  001362                  DCLASM                  ;SET DCLR IN CSR AND SET MNTFLG
013050  104417                                  MOV     LINE,RXTCR      ;SET IMAGE OF TCR BITS
013052  013737  001366  013572                  MOV     LINE,TXTCR      ;SET IMAGE OF TCR BITS
013060  013737  001366  013312                  LPRSET                  ;LOAD LINE PARAMETERS
013066  104421                                  BUFSET                  ;ZERO DATA BUFFER
013070  104422                                  MOV     #RXSVC,@DZVRIV  ;SET UP REC INTR VECTOR
013072  012777  013314  166740                  MOV     #MASK,@DZVRIS   ;STATUS
013100  012777  000200  166734                  MOV     #TXSVC,@DZVTIV  ;SET UP TRANS INTR VECTOR
013106  012777  013200  166730                  MOV     #MASK,@DZVTIS   ;STATUS
013114  012777  000200  166724                  BIS     #MSENAB!RIE!TIE,@DZVCSR ;SET MASTER SCAN ENABLE
013122  052777  040140  166660                  MOVB    LINE,@DZVTCR    ;SET TCR BITS
013130  113777  001366  166666                  MTPS    #CLEAR          ;ALLOW INTERRUPTS
013136  106427  000000
                                        SNAP:   CLR     4$              ;CLEAR DELAY COUNTER
013142  005037  013176                  2$:     DELAY                   ;WAIT FOR RECEIVERS TO FINISH
013144  104414                                  TSTB    RXTCR           ;WAIT FOR ALL RECIEVERS TO FINISH
013150  105737  013572                          BNE     3$              ;
013154  001002                                  JMP     OUT
013156  000137  013520                  3$:     INC     4$              ;INCREMENT DELAY COUNTER
013162  005237  013176                          BNE     2$              ;DELAY FINISHED?
013166  001367                                  ERROR   7               ;*TRANSMITTER FAILED TO INTERRUPT
013170  104007                                  ERROR   11              ;*RECEIVER FAILED TO INTERRUPT
013172  104011                                  ADVANCE                 ;LEAVE THIS TEST
013174  104400                          4$:     0
013176  000000
                                        ;TRANS INTR SVC ROUTINE
013200  117703  166606                  TXSVC:  MOVB    @HDZVCSR,R3     ;FIND LINE NO.
013204  042703  177774                          BIC     #^C<3>,R3       ;ISOLATE LINE NO.
013210  010304                                  MOV     R3,R4           ;SAVE LINE NO.
013212  010337  001374                          MOV     R3,SAVLIN       ;SAVE LINE NO.
013216  005777  166566                          TST     @DZVCSR         ;TRANS READY SET ?
013222  100401                                  BMI     .+4
013224  104003                                  ERROR   3               ;*TRANSMITTER FAILED
013226  012702  000001                          MOV     #1,R2           ;SET UP POSITION POINTER
013232  105303                          3$:     DECB    R3              ;IS IT THIS LINE ?
013234  100402                                  BMI     4$              ;YES
013236  006302                                  ASL     R2              ;UP THE LINE #
013240  000774                                  BR      3$              ;GO 'ROUND AGAIN
013242  030237  001366                  4$:     BIT     R2,LINE         ;VALID LINE?
013246  001001                                  BNE     .+4             ;YES
013250  104015                                  ERROR   15              ;NO,INVALID LINE!!!!
013252  030237  013312                          BIT     R2,TXTCR        ;DATA FINISHED?
013256  001003                                  BNE     6$              ;IF NOT SEND CHAR.
013260  040277  166540                          BIC     R2,@DZVTCR      ;CLEAR TCR BIT
013264  000411                                  BR      5$              ;RETURN
```

```
2966  013266  006304                        6$:     ASL     R4                      ;MAKE POWER OF 2
2967  013270  116477  001426  166536                MOVB    TDO(R4),@DZVTDR         ;LOAD CHARACTER
2968  013276  105264  001426                        INCB    TDO(R4)                 ;SET UP NEXT CHARACTER
2969  013302  001002                                BNE     5$                      ;LAST CHARACTER ?
2970  013304  040237  013312                        BIC     R2,TXTCR                ;INDICAT LINE FINISHED
2971  013310  000002                        5$:     RTI
2972
2973  013312  000000                        TXTCR:  0
2974
2975                                        ;REC INTR SVC ROUTINE
2976  013314  105777  166470                RXSVC:  TSTB    @DZVCSR                 ;REC DONE ?
2977  013320  100401                                BMI     .+4                     ;YES
2978  013322  104004                                ERROR   4                       ;FALSE INTERRUPT
2979  013324  032777  020000  166456                BIT     #SILOAL,@DZVCSR         ;SILO ALARM?
2980  013332  001401                                BEQ     .+4                     ;NO
2981  013334  104013                                ERROR   13                      ;SILO ALARM SHOULD NOT =1
2982  013336  017704  166452                        MOV     @DZVRBUF,R4             ;SAVE IT
2983  013342  010403                                MOV     R4,R3
2984  013344  000303                                SWAB    R3
2985  013346  042703  177774                        BIC     #^C<3>,R3               ;STRIP JUNK
2986  013352  010337  001374                        MOV     R3,SAVLIN               ;SAVE LINE NUMBER
2987  013356  005704                                TST     R4                      ;DATA VALID?
2988  013360  100401                                BMI     4$                      ;IF YES SKIP ERROR PRINTOUT
2989  013362  104023                                ERROR   23                      ;YOU LOSE  ...DATA VALID WAS'NT SET
2990  013364  032704  040000                4$:     BIT     #OVRRUN,R4              ;TEST FOR OVERRUN
2991  013370  001401                                BEQ     1$                      ;IF NO OVERRUN SKIP ERROR
2992  013372  104024                                ERROR   24                      ;DATA OVERRUN
2993  013374  032704  020000                1$:     BIT     #FRMERR,R4              ;DATA FRAMING ERROR
2994  013400  001401                                BEQ     2$                      ;IF NO FRAMING ERROR CONTINUE
2995  013402  104025                                ERROR   25                      ;FRAMING ERROR
2996  013404  032704  010000                2$:     BIT     #PARER,R4               ;TEST FOR PARITY ERROR
2997  013410  001401                                BEQ     3$                      ;BRANCH IF NO ERROR
2998  013412  104026                                ERROR   26                      ;TYPE OUT PARITY ERROR
2999  013414  012702  000001                3$:     MOV     #1,R2                   ;SET UP POSITION POINTER
3000  013420  105303                        5$:     DECB    R3
3001  013422  100402                                BMI     6$
3002  013424  006302                                ASL     R2                      ;RE POSITION POINTER
3003  013426  000774                                BR      5$                      ;GO 'ROUND AGAIN
3004  013430  030237  001366                6$:     BIT     R2,LINE                 ;LINE VALID ?
3005  013434  001001                                BNE     .+4                     ;YES
3006  013436  104015                                ERROR   15                      ;INVALID LINE #
3007  013440  013703  001374                        MOV     SAVLIN,R3               ;GET THE LINE NUMBER AGAIN
3008  013444  006303                                ASL     R3                      ;USE R3 AS A POINTER IN THE DATA TABLE
3009  013446  126304  001436                        CMPB    TRO(R3),R4              ;DOES THE DATA CHARACTER COMPARE ?
3010  013452  001410                                BEQ     7$                      ;YES
3011  013454  013705  001374                        MOV     SAVLIN,R5               ;MOVE LINE NO INTO EXPECTED
3012  013460  000305                                SWAB    R5                      ;ADJUST TO HIGH BYTE
3013  013462  052705  100000                        BIS     #DVALID,R5              ;SET DVALID IN EXPECTED
3014  013466  056305  001436                        BIS     TRO(R3),R5              ;SET DATA IN EXPECTED
3015  013472  104005                                ERROR   5                       ;*NO, DATA DOES NOT COMPARE
3016  013474  005263  001436                7$:     INC     TRO(R3)                 ;SET UP FOR NEXT CHARACTER
3017  013500  105763  001436                        TSTB    TRO(R3)                 ;ALL CHARS DONE?
3018  013504  001002                                BNE     .+6
3019  013506  040237  013572                        BIC     R2,RXTCR                ;ZERO LINE DONE INDICATOR.
3020  013512  012716  013142                        MOV     #SNAP,(SP)              ;RESET THE BACKGROUND TIMING LOOP
3021  013516  000002                                RTI
```

```
3022
3023
3024                                            ;FINISH UP ROUTINE
3025  013520  106427  000200          OUT:      MTPS    #MASK           ;STOP ALL INTERRUPTS
3026  013524  104413                           DEVICE.CLR              ;CLEAR ALL INTERRUPTS AWAY
3027  013526  005003                           CLR     R3
3028  013530  005037  001374                   CLR     SAVLIN
3029  013534  012702  000001                   MOV     #1,R2
3030  013540  030237  001366          1$:       BIT     R2,LINE         ;VALID LINE ?
3031  013544  001405                           BEQ     2$              ;NO
3032  013546  022763  000400  001436           CMP     #400,TRO(R3)    ;RECEIVED A BINARY COUNT PATTERN ?
3033  013554  001401                           BEQ     .+4             ;YES
3034  013556  104030                           ERROR   30              ;THE LINE FAILED TO RECEIVE A FULL
3035                                                                   ;BINARY COUNT PATTERN
3036  013560  005237  001374          2$:       INC     SAVLIN          ;SET UP FOR NEXT LINE
3037  013564  005723                           TST     (R3)+           ;ADD 2
3038  013566  104420                           SHIFT                   ;SET UP NEXT LINE POINTER
3039  013570  000763                           BR      1$              ;FINISHED ?
3040  013572  000000                   RXTCR:  0                       ;RX IMAGE OF TCR BITS
3041
3042
3043                                            ;************************* TEST 4 *********************************
3044                                            ;*DZV11 RELATIVE TIMING TEST.
3045                                            ;*EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
3046                                            ;*AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
3047                                            ;*WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
3048                                            ;*DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
3049                                            ;*THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
3050                                            ;* AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
3051                                            ;*PARAMETERS ARE:
3052                                            ;*  EIGHT BITS/PER/CHAR - TWO STOP BITS AT
3053                                            ;*      50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
3054                                            ;*      2400, 3600, 4800, 7200, 9600 BAUD.
3055                                            ;*  19.2 K BAUD - TWO STOP BITS AT
3056                                            ;*      SEVEN, SIX, FIVE BITS/PER/CHAR.
3057                                            ;*AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
3058                                            ;*THE NEXT SELECTED LINE IS THEN TESTED.
3059                                            ;*WHEN RUNNING UNDER THE APT MANUFACTURING SYSTEM
3060                                            ;*THIS TEST IS ONLY RUN THE FIRST PASS
3062                                    ;:*  TEST 4
 (5)                                    ;:*************************************************************************
 (4)  013574  000004                    TST4:   SCOPE
 (2)  013576  012737  000004  001246            MOV     #4,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
 (2)  013604  012737  014212  001362            MOV     #TST5,NEXT      ;POINT TO THE START OF THE NEXT TEST
 (1)  013612  012737  013732  001364            MOV     #3$,LOCK        ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
3063  013620  132737  000001  001140            BITB    #1,$ENV         ;RUNNING UNDER APT?
3064  013626  001405                           BEQ     10$             ;IF NOT CONTINUE WITH TEST
3065  013630  005737  001126                   TST     $PASS           ;IF YES IS THIS FIRST PASS
3066  013634  001402                           BEQ     10$             ;IF NOT 1ST PASS SKIP TEST
3067  013636  000177  165520                   JMP     @NEXT
3068  013642  012737  000002  001354  10$:     MOV     #2,$TIMES       ;SET UP FOR 2 ITERATIONS
3069  013650  005037  015214                   CLR     OFFSET          ;RESET THIS VARIABLE
3070  013654  005037  001374                   CLR     SAVLIN          ;RESET LINE NUMBER INDICATOR
3071  013660  005037  001376                   CLR     XMTLIN          ;USE THIS WORD TO TELL WHAT LINE TRANSMITTED
3072  013664  012702  000001                   MOV     #1,R2           ;USE R2 AS A BIT POINTER
3073  013670  012703  010070                   MOV     #RCVON!S50!EIGHT!TWOSTOP,R3 ;BUILD TEMPORARY PARAMETERS
```

```
 3074  013674  030237  001366          1$:     BIT     R2,LINE         ;IS THIS LINE ACTIVE?
 3075  013700  001014                          BNE     3$              ;IF SO, GO GET STARTED
 3076  013702  012703  010070          2$:     MOV     #RCVON!S50!EIGHT!TWOSTOP,R3 ;LOAD PARAMETERS TEMPORARILY
 3077  013706  005237  001376                  INC     XMTLIN          ;POINT TO THE NEXT LINE TO TRANSMIT
 3078  013712  042703  000007                  BIC     #7,R3           ;MAKE SURE TEMPORARY PARAMETERS POINT TO 0
 3079  013716  053703  001376                  BIS     XMTLIN,R3       ;ADD DESIRED LINE NUMBER
 3080  013722  005037  015214                  CLR     OFFSET
 3081  013726  104420                          SHIFT                   ;POINT TO THE NEXT LINE
 3082  013730  000761                          BR      1$              ;PROCESS THE NEXT LINE
 3083  013732                          3$:
   (1) 013732  104417                          DCLASR                  ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
 3084  013734  042703  010000                  BIC     #RCVON,R3       ;ZERO PARAMTERS FOR TX LINE
 3085  013740  010377  166054                  MOV     R3,@DZVLPR      ;LOAD PARAMTERS FOR TX
 3086  013744  005737  001372                  TST     MODE            ;STAGGERED?
 3087  013750  100007                          BPL     100$            ;BR IF NO
 3088  013752  000241                          CLC                     ;SET UP LINE
 3089  013754  006003                          ROR     R3              ;
 3090  013756  103002                          BCC     98$             ;BR IF LINE WAS EVEN
 3091  013760  000241                          CLC                     ;PREPARE TO MKE LINE EVEN
 3092  013762  000401                          BR      99$             ;CONTINUE
 3093  013764  000261                  98$:    SEC                     ;PREPARE TO MAKE LINE ODD
 3094  013766  006103                  99$:    ROL     R3              ;SET ALTERED LINE
 3095  013770  052703  010000          100$:   BIS     #RCVON,R3       ;SET RX ON
 3096  013774  010377  166020                  MOV     R3,@DZVLPR      ;LOAD RX PARAMETERS
 3097  014000  010337  001374                  MOV     R3,SAVLIN       ;SET FOR RECEIV. LINE
 3098  014004  042737  177774  001374          BIC     #^C<3>,SAVLIN   ;ISOLATE LINE NO.
 3099  014012  042703  000003                  BIC     #3,R3           ;CLEAR OLD LINE #
 3100  014016  053703  001376                  BIS     XMTLIN,R3       ;SET LINE UP AGAIN
 3101  014022  010337  001402                  MOV     R3,REGIST       ;SAVE PARAMETERS FOR PRINTOUT
 3102  014026  104422                          BUFSET                  ;ZERO DATA BUFFER
 3103  014030  005037  001342                  CLR     $TMP0           ;USE $TMP0 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
 3104  014034  005037  001344                  CLR     $TMP1           ;INITIALIZE THE TIMER
 3105  014040  005037  001350                  CLR     $TMP3           ;INITIALIZE THESE BITS ALSO
 3106  014044  012737  000020  001400          MOV     #20,XMTCNT      ;SET HOW MANY CHARACTERS TO TRANSMIT
 3107  014052  012777  014642  165764          MOV     #XMTSRV,@DZVTIV
 3108  014060  012777  015012  165752          MOV     #RXISR1,@DZVRIV
 3109  014066  012777  000200  165746          MOV     #MASK,@DZVRIS
 3110  014074  012777  000200  165744          MOV     #MASK,@DZVTIS
 3111  014102  110277  165716                  MOVB    R2,@DZVTCR      ;START THE VALID LINE
 3112  014106  052777  040140  165674          BIS     #TIE!RIE!MSENAB,@DZVCSR
 3113  014114  106427  000000                  MTPS    #0              ;LOWER THE PRIORITY TO ALLOW INTERRUPTS
 3114  014120  032777  000100  165662  4$:     BIT     #RIE,@DZVCSR    ;IS ROUTINE DONE?
 3115  014126  001407                          BEQ     5$              ;WHEN ALL IS DONE RX IE IS CLEARED IN ISR.
 3116  014130  005237  001344                  INC     $TMP1           ;INCREMENT TIMER
 3117  014134  001371                          BNE     4$              ;WHEN IT OVERFLOWS
 3118  014136  005237  001350                  INC     $TMP3           ;CATCH CARRY
 3119  014142  001366                          BNE     4$              ;CONTINUE TEST
 3120  014144  104011                          ERROR   11              ;INTERRUPTS NOT FINISHED
 3121  014146  004737  007126          5$:     JSR     PC,SERV.G       ;<^G>?
 3122  014152  104401                          SCOP1                   ;LOOP?
 3123  014154  062737  000002  015214          ADD     #2,OFFSET
 3124  014162  022703  017400                  CMP     #17400,R3
 3125  014166  003006                          BGT     6$
 3126  014170  032703  000030                  BIT     #BIT4+BIT3,R3   ;IS  CHARACTER SIZE DONE?
 3127  014174  001642                          BEQ     2$
 3128  014176  162703  000010                  SUB     #BIT3,R3
```

```
 3129  014202  000653                          BR      3$
 3130  014204  062703  000400          6$:     ADD     #400,R3
 3131  014210  000650                          BR      3$
 3132                                          ;************************** TEST 5 ********************************
  (1)                                          ;*THE MAIN FUNCTION OF THIS TEST IS TO VERIFY
  (1)                                          ;*THAT 'PE' (PARITY ERROR) CAN BE FLAGGED BY
  (1)                                          ;*THE UARTS. THIS TEST WILL NOT BE DONE UNLESS
  (1)                                          ;*YOU ARE IN 'STAGGERED" MODE.
  (1)                                          ;*40(8) CHARS ARE USED FOR THIS TEST.
  (1)                                          ;*ALL SELECTED LINES WILL BE ENABLED AT THE SAME TIME.
  (1)                                          ;*THIS TEST FIRST CHECKS EVEN PARITY FOR ODD LINES AND
  (1)                                          ;*ODD PARITY FOR EVEN LINES, THEN IT CHECKS THE REVERSE.
  (3)                                  ;:*  TEST 5
  (6)                                  ;:*************************************************************************
  (5)  014212  000004                  TST5:   SCOPE
  (3)  014214  012737  000005  001246          MOV     #5,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
  (2)  014222  012737  004146  001362          MOV     #$EOP,NEXT      ;POINT TO THE END-OF-PASS HANDLER
  (1)  014230  005737  001372                  TST     MODE            ;IS THIS STAGGERED MODE?
  (1)  014234  100131                          BPL     17$             ;IF NOT, DON'T DO THIS TEST
  (1)  014236  105037  001425                  CLRB    DONFLG          ;SET UP FOR FIRST TEST PASS
  (1)  014242  104413                  14$:    DEVICE.CLR              ;SET DCLR IN CSR
  (1)  014244  013701  001370                  MOV     PAR,R1          ;USE R1 TO BUILD PARAMETERS TO BE LOADED
  (1)  014250  042701  000200                  BIC     #ODDPAR,R1      ;MAKE SURE ODD PARITY ISN'T SET
  (1)  014254  052701  000100                  BIS     #PARITY,R1      ;MAKE SURE PARITY IS TURNED ON
  (1)  014260  012702  000001                  MOV     #1,R2           ;USE R2 AS A LINE POINTER
  (1)  014264  030237  001366          1$:     BIT     R2,LINE         ;IS THIS A VALID LINE?
  (1)  014270  001420                          BEQ     3$              ;IF NOT, SKIP TO THE NEXT LINE
  (1)  014272  105737  001425                  TSTB    DONFLG          ;FIRST PASS THROUGH TEST?
  (1)  014276  001004                          BNE     15$             ;IF NO BRANCH
  (1)  014300  032701  000001                  BIT     #BIT0,R1        ;IS THIS LINE AN ODD LINE?
  (1)  014304  001006                          BNE     2$              ;IF IT'S ODD, USE EVEN PARITY
  (1)  014306  000403                          BR      16$             ;IF EVEN SET FOR ODD PARITY
  (1)  014310  032701  000001          15$:    BIT     #BIT0,R1        ;IF THE LINE  IS EVEN SET FOR EVEN PAR.
  (1)  014314  001402                          BEQ     2$              ;GO LOAD PARAMETER
  (1)  014316  052701  000200          16$:    BIS     #ODDPAR,R1      ;IF IT'S ODD, USE ODD PARITY
  (1)  014322  010177  165472          2$:     MOV     R1,@DZVLPR      ;LOAD THE LINE PARAMETER REGISTER
  (1)  014326  042701  000200                  BIC     #ODDPAR,R1      ;SET UP THE NEXT PARITY TO EVEN
  (1)  014332  005201                  3$:     INC     R1              ;POINT TO THE NEXT LINE
  (1)  014334  006302                          ASL     R2              ;
  (1)  014336  032702  000020                  BIT     #BIT4,R2        ;ALL LINES DONE?
  (1)  014342  001750                          BEQ     1$              ;IF NOT, GO CHECK THE NEXT LINE
  (1)  014344  005037  001374                  CLR     SAVLIN          ;CLEAR THE LINE NUMBER INDICATOR
  (1)  014350  005037  001342                  CLR     $TMP0           ;USE $TMP0 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
  (1)  014354  005003                          CLR     R3              ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
  (1)  014356  012737  000040  001400          MOV     #40,XMTCNT      ;TRANSMIT A BINARY COUNT PATTERN(00-40)
  (1)  014364  104422                          BUFSET                  ;ZERO BUFFER AREA
  (2)  014366  012777  014642  165450          MOV     #XMTSRV,@DZVTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
  (2)  014374  012777  014522  165436          MOV     #9$,@DZVRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
  (2)  014402  012777  000200  165432          MOV     #MASK,@DZVRIS   ;SET THE INTERRUPT VECTOR STATUS
  (2)  014410  012777  000200  165430          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
  (2)  014416  052777  040140  165364          BIS     #RIE!TIE!MSENAB,@DZVCSR ;ENABLE THE DEVICE
  (1)  014424  113777  001366  165372          MOVB    LINE,@DZVTCR    ;ENABLE ALL SELECTED LINES
  (1)  014432  106427  000000                  MTPS    #0              ;ALLOW INTERRUPTS
  (1)  014436  005037  014514          4$:     CLR     7$
  (1)  014442  005037  014516                  CLR     8$
  (1)  014446  032777  000100  165334  5$:     BIT     #RIE,@DZVCSR    ;WHEN RX DONE; RIE WILL =0
```

```
(1)  014454  001407                    BEQ     6$              ;BR IF ALL DONE
(1)  014456  005237  014514            INC     7$
(1)  014462  001371                    BNE     5$
(1)  014464  105237  014516            INCB    8$
(1)  014470  100366                    BPL     5$
(1)  014472  104011                    ERROR   11              ;*RX FAILED TO FINISH (INTERRUPT)
(1)  014474  106427  000200    6$:     MTPS    #MASK           ;SHUT OFF INTERRUPTS
(1)  014500  105737  001425            TSTB    DONFLG          ;IS THIS SECOND TEST PASS
(1)  014504  001005                    BNE     17$             ;IF SO GET OUT
(1)  014506  105237  001425            INCB    DONFLG          ;INDICATE FIRST TEST PASS DONE
(1)  014512  000653                    BR      14$             ;START OVER
(1)  014514  000000    7$:     0
(1)  014516  000000    8$:     0
(1)  014520  104400    17$:    ADVANCE
(1)
(1)
(1)                                    ;RECEIVER SERVICE ROUTINE
(1)
(1)  014522  017704  165266    9$:     MOV     @DZVRBUF,R4     ;GET THE CHARACTER
(1)  014526  010401                    MOV     R4,R1           ;COPY THE RECEIVED INFORMATION
(1)  014530  000301                    SWAB    R1              ;GET THE LINE NUMBER IN THE LOWER BYTE
(1)  014532  042701  177774            BIC     #^C<3>,R1       ;ISOLATE THE LINE NUMBER
(1)  014536  010137  001374            MOV     R1,SAVLIN       ;SET LINE INDIC. TO RECEIVING LINE
(1)  014542  005704                    TST     R4              ;IS DATA VALID SET?
(1)  014544  100401                    BMI     10$             ;IF YES DON'T PRINT ERROR
(1)  014546  104023                    ERROR   23              ;DATA VALID NOT SET
(1)  014550  010105    10$:    MOV     R1,R5           ;BUILD LINE NO. FOR
(1)  014552  000305                    SWAB    R5              ;EXPECTED DATA IN RECEIVER BUFFER
(1)  014554  006301                    ASL     R1              ;ADJUST R1 FOR OFFSET
(1)  014556  156105  001436            BISB    TRO(R1),R5      ;LOAD CHARACTER IN EXPECTED
(1)  014562  052705  110000            BIS     #DVALID!PARER,R5 ;BUILD WHAT WAS EXPECTED
(1)  014566  020405                    CMP     R4,R5           ;DOES RECEIVED=EXPECTED
(1)  014570  001401                    BEQ     12$             ;IF YES DON'T PRINT ERROR
(1)  014572  104006                    ERROR   6               ;*ERROR- DID NOT GET CORRECT INFORMATION
(1)  014574  005261  001436    12$:    INC     TRO(R1)         ;SET UP THE NEXT CHARACTER
(1)  014600  005203                    INC     R3              ;ADD TO THE TOTAL RECEIVED COUNT
(1)  014602  032777  040000  165200    BIT     #TIE,@DZVCSR    ;ARE TRANSMISSIONS DONE?
(1)  014610  001011                    BNE     13$             ;IF NO, GO RECEIVE SOME MORE
(1)  014612  023703  001342            CMP     $TMP0,R3        ;ARE ALL CHARACTERS RECEIVED?
(1)  014616  001006                    BNE     13$             ;IF NO, GO RECEIVE SOME MORE
(1)  014620  042777  000100  165162    BIC     #RIE,@DZVCSR    ;DISABLE RECEIVER INTERRUPTS
(1)  014626  012716  014474            MOV     #6$,(SP)        ;CRUNCH THE STACK
(1)  014632  000002                    RTI                     ;RETURN AND FINISH
(1)  014634  012716  014436    13$:    MOV     #4$,(SP)        ;CRUNCH THE STACK
(1)  014640  000002                    RTI                     ;GO BACK TO RECEIVER WAIT LOOP
```

```
3134
3135                                            ;TRANSMITTER INTERRUPT SERVICE
3136                                            ;----------------------------
3137
3138  014642  117701  165144          XMTSRV: MOVB    @#DZVCSR,R1     ;GET THE LINE NUMBER.
3139  014646  042701  177774                  BIC     #^C<3>,R1       ;CLEAR JUNK
3140  014652  013705  001374                  MOV     SAVLIN,R5       ;SAVE REC. LINE NO.
3141  014656  010137  001374                  MOV     R1,SAVLIN       ;LOAD TRANS LINE NO FOR ERROR PRINTOUT
3142  014662  006301                          ASL     R1              ;ADJUST R1 FOR OFFSET
3143  014664  023761  001400  001426          CMP     XMTCNT,TDO(R1)  ;HAVE ALL CHAR. BEEN SENT
3144  014672  003414                          BLE     6$              ;IF YES GO CLEAR TCR
3145  014674  005777  165110                  TST     @#DZVCSR        ;TRDY SET?
3146  014700  100401                          BMI     2$              ;IF YES GO LOAD CHAR.
3147  014702  104003                          ERROR   3               ;*TRANSMITTER NOT READY- FALSE INTERRUPT
3148  014704  116177  001426  165122  2$:     MOVB    TDO(R1),@#DZVTDR ;LOAD THE CURRENT CHARACTER FOR THIS LINE
3149  014712  005261  001426                  INC     TDO(R1)         ;SET UP NEXT CHARACTER FOR THIS LINE
3150  014716  005237  001342                  INC     $TMP0           ;UP THE NUMBER OF TRANSMISSIONS
3151  014722  000415                          BR      7$              ;GO RETURN
3152  014724  012700  000001          6$:     MOV     #1,R0           ;SET UP A DESELECTION POINTER
3153  014730  006201                          ASR     R1              ;GET LINE NO. AGAIN
3154  014732  005301                  12$:    DEC     R1              ;REDUCE THE COUNT. WAS THIS THE LINE?
3155  014734  100402                          BMI     3$              ;IF SO, GO DISABLE THE ENABLE BIT FOR IT
3156  014736  006300                          ASL     R0              ;MOVE THE POINTER TO THE NEXT LINE
3157  014740  000773                          BR      12$             ;GO CHECK THE NEXT LINE
3158  014742  140077  165056          3$:     BICB    R0,@#DZVTCR     ;DISABLE THE LINE POINTED TO BY R0
3159  014746  001003                          BNE     7$              ;IF MORE LINES ARE ACTIVE , GO CONTINUE TRANSMIT
3160  014750  042777  040000  165032          BIC     #TIE,@#DZVCSR   ;IF NOT, DISABLE TRANSMITTER INTERRUPTS
3161  014756  010537  001374          7$:     MOV     R5,SAVLIN       ;RESTORE RECEIV. LINE
3162  014762  000002                          RTI                     ;RETURN TO THE TIMING LOOP
3163
3164                                            ; RELATIVE TIME BUILDING ROUTINE
3165                                            ; -----------------------------
3166
3167  014764  012737  000002  001346  BUILD:  MOV     #2,$TMP2        ;ROTATE 2 BITS BACK INTO $TMP1
3168  014772  006037  001350          1$:     ROR     $TMP3           ;GET THE BITS FROM $TMP3, THE HIGH BYTE
3169  014776  006037  001344                  ROR     $TMP1           ;OF THE RELATIVE TIME COUNTER. PUT THEM BACK
3170  015002  005337  001346                  DEC     $TMP2           ;INTO $TMP1 USING THE CARRY BIT WITH
3171                                                                  ;ROTATE INSTRUCTIONS
3172  015006  001371                          BNE     1$              ;REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
3173  015010  000207                          RTS     PC              ;RETURN TO CALLING TEST
3174
```

```
                                                ;RECEIVER SERVICE ROUTINE
                                        RXISR1: TSTB    @DZVCSR         ;IS THE RECEIVER REALLY READY?
                                                BMI     1$              ;IF SO, GO SERVICE IT
                                                ERROR   4               ;*ERROR- RECEIVER DONE FLAG ISN'T SET
                                        1$:     MOV     @DZVRBUF,R4     ;SAVE THE RECEIVER INFORMATION
                                                BMI     2$              ;IF IT WAS VALID, GO PROCESS IT
                                                ERROR   23              ;ERROR- DATA VALID WASN'T SET
                                        2$:     BIT     #OVRRUN,R4      ;OVERRUN ERROR FLAG SET?
                                                BEQ     6$              ;IF NOT DON'T TYPE ERROR
                                                ERROR   24              ;OVERRUN ERROR
                                        6$:     BIT     #FRMERR,R4      ;FRAMING ERROR FLAG SET?
                                                BEQ     9$              ;IF NOT DON'T TYPE ERROR
                                                ERROR   25              ;FRAMING ERROR
                                        9$:     BIT     #PARER,R4       ;PARITY ERROR FLAG SET?
                                                BEQ     3$              ;IF NOT, GO CONTINUE PROCESSING
                                                ERROR   26              ;ERROR- RECEIVER ERROR FLAG SET
                                        3$:     MOV     SAVLIN,R1       ;CALCULATE THE DATA OFFSET
                                                ASL     R1              ;ALIGN IT ON A WORD BOUNDARY
                                                CMPB    R4,TRO(R1)      ;IS THE CHARACTER WHAT IT SHOULD BE?
                                                BEQ     4$              ;IF SO,GO CONTINUE PROCESSING
                                                MOVB    TRO(R1),R5      ;GET WHAT WAS EXPECTED FOR ERROR REPORTING
                                                BIC     #^C<377>,R5     ;ELIMINATE PROPAGATED SIGN
                                                BIC     #^C<377>,R4     ;ISOLATE THE ACTUAL CHARACTER
                                                ERROR   5               ;*DATA ERROR
                                        4$:     INC     TRO(R1)         ;SET UP THE NEXT EXPECTED CHARACTER
                                                CMPB    #20,TRO(R1)     ;HAVE ALL CHARACTERS BEEN RECEIVED?
                                                BNE     8$              ;IF NOT RETURN
                                                CMPB    TRO(R1),$TMPO   ;ALL CHARAC. RECEIVED?
                                                BNE     8$              ;IF SO,GO DETERMINE THE TIMING
                                                JSR     PC,BUILD        ;GET THE RELATIVE TIME (SIGNIFICANT BITS)
                                                MOV     OFFSET,RO       ;GET POINTER
                                                MOV     $TMP1,TMTBL(RO) ; SAVE THIS TEST'S TIME
                                                TST     OFFSET          ;FIRST TEST?
                                                BEQ     7$              ;IF NOT, GO CHECK THE TIME
                                                TST     -(RO)           ;POINT TO THE PREVIOUS  TIME TAKEN
                                                CMP     TMTBL(RO),$TMP1 ;IS THIS TIME WHAT IT SHOULD BE?
                                                BHI     7$              ;IF SO, GO TO THE NEXT TEST
                                                MOV     TMTBL(RO),R5    ;PLACE WHAT WAS EXPECTED IN R5
                                                ERROR   21              ;TIMING ERROR
                                        7$:     BIC     #RIE!MSENAB,@DZVCSR ;DISABLE THE DEVICE
                                        8$:     RTI                     ;RETURN TO THE PROGRAM
                                        OFFSET: 0
```

```
                                        ;ERROR TABLE
015216  000000          .ERRTAB:        0       ;ERROR 0
015220  000000                          0
015222  000000                          0

015224  015444                          EM1     ;ERROR
015226  016570                          DH1
015230  016770                          DT1

015232  015517                          EM2     ;ERROR 2
015234  016614                          DH2
015236  017002                          DT2

015240  015545                          EM3     ;ERROR 3
015242  016647                          DH3
015244  017020                          DT3

015246  015604                          EM4     ;ERROR 4
015250  016647                          DH3
015252  017020                          DT3

015254  015633                          EM5     ;ERROR 5
015256  016661                          DH4
015260  017026                          DT4

015262  015662                          EM6     ;ERROR 6
015264  016661                          DH4
015266  017026                          DT4

015270  000000                          0
015272  000000                          0
015274  000000                          0

015276  000000                          0
015300  000000                          0
015302  000000                          0

015304  015721                          EM11    ;ERROR 11
015306  016647                          DH3
015310  017020                          DT3

015312  000000                          0
015314  000000                          0
015316  000000                          0

015320  015757                          EM13    ;ERROR 13
015322  016647                          DH3
015324  017020                          DT3

015326  016010                          EM14    ;ERROR 14
015330  016647                          DH3
015332  017020                          DT3

015334  016042                          EM15    ;ERROR 15
015336  000000                          0
015340  000000                          0
```

```
3276
3277  015342  016104                          EM16
3278  015344  016647                          DH3
3279  015346  017020                          DT3
3280
3281  015350  016156                          EM17    ;ERROR 17
3282  015352  016647                          DH3
3283  015354  017020                          DT3
3284
3285  015356  016214                          EM20
3286  015360  016647                          DH3
3287  015362  017020                          DT3
3288
3289  015364  016255                          EM21    ;ERROR 21
3290  015366  016710                          DH5
3291  015370  017044                          DT5
3292
3293  015372  000000                          0
3294  015374  000000                          0
3295  015376  000000                          0
3296
3297  015400  016305                          EM23    ;ERROR 23
3298  015402  016647                          DH3
3299  015404  017020                          DT3
3300
3301  015406  016335                          EM24
3302  015410  016647                          DH3
3303  015412  017020                          DT3
3304
3305  015414  016363                          EM25
3306  015416  016647                          DH3
3307  015420  017020                          DT3
3308
3309  015422  016413                          EM26
3310  015424  016647                          DH3
3311  015426  017020                          DT3
3312
3313  015430  016442                          EM27
3314  015432  016647                          DH3
3315  015434  017020                          DT3
3316
3317  015436  016510                          EM30
3318  015440  016647                          DH3
3319  015442  017020                          DT3
```

```
                                                        ;ERROR MESSAGES
015444  047200  020117  052502  EM1:    .ASCIZ  <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
015517    200   042522  044507  EM2:    .ASCIZ  <200>?REGISTER R/W FAILURE?
015545    200   051124  047101  EM3:    .ASCIZ  <200>/TRANSMIT READY (TRDY) NOT SET/
015604  051200  041505  044505  EM4:    .ASCIZ  <200>/RECEIVER DONE NOT SET/
015633    200   040504  040524  EM5:    .ASCIZ  <200>/DATA COMPARISON ERROR/
015662  042200  053132  030461  EM6:    .ASCIZ  <200>/DZV11 *RECEIVER BUFFER* ERROR/
015721    200   042522  042503  EM11:   .ASCIZ  <200>/RECEIVER FAILED TO INTERRUPT/
015757    200   044523  047514  EM13:   .ASCIZ  <200>/SILO ALARM SET TOO SOON/
016010  051600  046111  020117  EM14:   .ASCIZ  <200>/SILO ALARM FAILED TO SET/
016042  040600  052103  047511  EM15:   .ASCIZ  <200>/ACTION DETECTED ON INVALID LINE./
016104  051200  040505  044504  EM16:   .ASCIZ  <200>/READING DZVRBUF DID NOT CLEAR SILO ALARM/
016156  042200  052101  020101  EM17:   .ASCIZ  <200>/DATA VALID SHOULD NOT BE SET/
016214  051200  041505  044505  EM20:   .ASCIZ  <200>/RECEIVER DONE SHOULD NOT BE SET/
016255    200   042522  040514  EM21:   .ASCIZ  <200>/RELATIVE TIMING ERROR./
016305    200   040504  040524  EM23:   .ASCIZ  <200>/DATA VALID IS NOT SET!/
016335    200   040504  040524  EM24:   .ASCIZ  <200>/DATA OVERRUN IS SET!/
016363    200   051106  046501  EM25:   .ASCIZ  <200>/FRAMING ERROR OCCURRED/
016413    200   040520  044522  EM26:   .ASCIZ  <200>/PARITY ERROR OCCURRED/
016442  051600  046111  020117  EM27:   .ASCIZ  <200>/SILO ALARM FAILED TO CAUSE INTERRUPT/
016510  046200  047111  020105  EM30:   .ASCIZ  <200>/LINE DID NOT RECEIVE FULL BINARY COUNT PATTERN/

016570  052200  040522  020120  DH1:    .ASCIZ  <200>/TRAP PC  DZV11 REG/
016614  042600  050130  041505  DH2:    .ASCIZ  <200>/EXPECTED  FOUND  REGISTER/
016647    200   044514  042516  DH3:    .ASCIZ  <200>/LINE NO./
016661    200   054105  042520  DH4:    .ASCIZ  <200>/EXPECTED  FOUND  LINE/
016710  052200  020130  044514  DH5:    .ASCIZ  <200>/TX LINE PREVIOUS TIME  ACTUAL TIME  PARAMETER/

        016770                  .EVEN
                                ;DATA TABLES FOR ERROR MESSAGES
016770  000002                  DT1:    2
016772    006     003                   .BYTE   6,3
016774  001330                        $REG1
016776    006     001                   .BYTE   6,1
017000  001326                        $REG0

017002  000003                  DT2:    3
017004    006     004                   .BYTE   6,4
017006  001340                        $REG5
017010    006     001                   .BYTE   6,1
017012  001336                        $REG4
017014    006     001                   .BYTE   6,1
017016  001326                        $REG0

017020  000001                  DT3:    1
017022    003     001                   .BYTE   3,1
017024  001374                        SAVLIN

017026  000003                  DT4:    3
017030    006     004                   .BYTE   6,4
017032  001340                        $REG5
017034    006     001                   .BYTE   6,1
017036  001336                        $REG4
017040    003     001                   .BYTE   3,1
017042  001374                        SAVLIN
```

```
3383  017044  000004                DT5:    4
3384  017046     003      005               .BYTE   3,5
3385  017050  001374                        SAVLIN
3386  017052     006      011               .BYTE   6,9.
3387  017054  001340                        $REG5
3388  017056     006      007               .BYTE   6,7
3389  017060  001344                        $TMP1
3390  017062     006      001               .BYTE   6,1
3391  017064  001402                        REGIST
3398                                        ;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES
3399                                        ;---------------------------------------------
3400
3401  017066  002450                DLYTBL: 2450                    ;TIME FOR   50 BAUD
3402  017070  001560                        1560                    ;TIME FOR   75 BAUD
3403  017072  001120                        1120                    ;TIME FOR  110 BAUD
3404  017074  000750                         750                    ;TIME FOR  134 BAUD
3405  017076  000660                         660                    ;TIME FOR  150 BAUD
3406  017100  000330                         330                    ;TIME FOR  300 BAUD
3407  017102  000150                         150                    ;TIME FOR  600 BAUD
3408  017104  000060                          60                    ;TIME FOR 1200 BAUD
3409  017106  000040                          40                    ;TIME FOR 1800 BAUD
3410  017110  000030                          30                    ;TIME FOR 2000 BAUD
3411  017112  000020                          20                    ;TIME FOR 2400 BAUD
3412  017114  000010                          10                    ;TIME FOR 3600 BAUD
3413  017116  000001                           1                    ;TIME FOR 4800 BAUD
3414  017120  000001                           1                    ;TIME FOR 7200 BAUD
3415  017122  000001                           1                    ;TIME FOR 9600 BAUD
3416  017124  000001                           1                    ;TIME OF DELAY FOR 19200 BAUD
3417
3418                                        ;DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
3419                                        ;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.
                                        ;
017126  005227  177777          FALCON: INC     #-1             ; ONCE-ONLY !!!                  ;:GPA
017132  001002                          BNE     1$                                               ;:GPA
017134  004737  000400                  CALL    KXTCHK          ; EXECUTE FALCON CHECK           ;:GPA
017140  005727                  1$:     TST     (PC)+           ; TEST FALCON FLAG...            ;:GPA
017142  000000                  KXTFLAG: 0                      ;...NZ = FALCON...               ;:GPA
017144  000207                          RETURN                  ;...AND RETURN TO CALLER...      ;:GPA
                                                                                                 ;:GPA
017146                                  $SVPC= .                                                 ;:GPA
000400                                  .= 400                  ; RESTORE FROM 374:376 AT END    ;:GPA
000400  005037  017142          KXTCHK: CLR     KXTFLAG         ; ASSUME NOT FALCON.             ;:GPA
000404  013746  000004                  MOV     @#4,-(SP)       ; SAVE ERROR VECTOR.             ;:GPA
000410  012737  000504  000004          MOV     #2$,@#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  017142                  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
                                                                                                 ;: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                                           ;:GPA
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
```

```
3477                                    ;
3478                                    ; THIS CODE IS RELOCATED TO AND EXECUTED IN THE STACK AREA.
3479                                    ;
3480  000546  010060  177776           5$:     MOV     R0,-2(R0)       ; RESTORE .+2...                ;:GPA
3481  000552  010110                           MOV     R1,(R0)         ;...HALT (OR IOT).              ;:GPA
3482  000554  022020                           CMP     (R0)+,(R0)+                                     ;:GPA
3483  000556  020027  000776                   CMP     R0,#776                                         ;:GPA
3484  000562  101771                           BLOS    5$              ; LOOP 'TIL DONE                ;:GPA
3485  000564  010206                           MOV     R2,SP           ; THEN RESTORE SP...            ;:GPA
3486  000566  000207                           RETURN                  ;...AND RETURN TO CALLER        ;:GPA
3487  000570                                 6$:                                                      ;:GPA
3488                                    ;
3489                                    ; IF FALCON, THIS AREA IS FREE FOR ANY PROGRAM UNIQUE
3490                                    ; CHANGES OR DATA STRUCTURES.
3491                                    ; BE SURE IT DOESN'T GET SCREWED UP !!
3492                                    ;
3493                                    ; INIT $BASE AND $VECT1 AND TWEAK THE "$GETPAR" CALLING
3494                                    ; SEQUENCE TO ACCEPT THE VALID FALCON RANGE.
3495                                    ;
3496  000570  023727  001174  160010   FALCINI: CMP    $BASE,#ABASE    ; IS $BASE VIRGIN ??            ;:GPA
3497  000576  001003                           BNE     1$              ; SKIP NEXT IF NOT              ;:GPA
3498  000600  012737  174040  001174           MOV     #174040,$BASE   ; YES, SET ENGINEERING DEFAULT  ;:GPA
3499  000606  023727  001170  000300   1$:     CMP     $VECT1,#AVECT1  ; IS $VECT1 VIRGIN ??           ;:GPA
3500  000614  001003                           BNE     2$              ; SKIP NEXT IF NOT              ;:GPA
3501  000616  012737  000370  001170           MOV     #370,$VECT1     ; YES, SET ENGINEERING DEFAULT  ;:GPA
3502  000624  012737  000670  002362   2$:     MOV     #3$,GETCSR+2    ; SUBSTITUE CSR TEXT...         ;:GPA
3503  000632  012737  174000  002364           MOV     #174000,GETCSR+6                                ;:GPA
3504  000640  012737  177770  002370           MOV     #177770,GETCSR+10 ;...AND VALID RANGE.          ;:GPA
3505  000646  012737  000732  002406           MOV     #4$,GETVEC+2    ; SUBSTITUTE VECTOR TEXT...     ;:GPA
3506  000654  005037  002412                   CLR     GETVEC+6                                        ;:GPA
3507  000660  012737  000370  002414           MOV     #370,GETVEC+10  ;...AND VALID RANGE.            ;:GPA
3508  000666  000207                           RETURN                  ; RETURN TO CALLER.             ;:GPA
3509
3510  000670  030600  052123  041440   3$:     .ASCIZ  <200>'1ST CSR ADDRESS (174000:177770) '         ;:GPA
3511  000732  030600  052123  053040   4$:     .ASCIZ  <200>'1ST VECTOR ADDRESS (000:370)    '         ;:GPA
3512                                           .EVEN                                                   ;:GPA
3513
3514          000002                   $FREE=  <1000-.>/2              ; FREE WORDS LEFT.              ;:GPA
3518
3519          017146                           .=$SVPC                                                 ;:GPA
3520  017146                           CORMAX:
3524          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  006015
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  011364
AVECT1= 000300
AVECT2= 000000
BINWRD  006270
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     004622
BUFSET= 104422
BUILD   014764
CHRCNT  006266
CLEAR = 000000
CNVRT = 104412
CONVRT= 104411
CORMAX  017146
C00   = 000400
C01   = 001000
C02   = 002000
C03   = 004000
CR    = 000015
CRLF  = 000200
CSRMAP  011372
CYCLE   010530
DATABP  006762
DATAHD  006750
DCLASM= 104417
DCLR  = 000020
DDISP = 177570
DELAY = 104414
DEVADR  006012
DEVICE= 104413
DH1     016570
DH2     016614
DH3     016647
DH4     016661
DH5     016710
DISPLA  001306
DISPRE  000174
DLYCNT  006364
DLYTBL  017066
DONFLG  001425
DSWR  = 177570
DTR0  = 000400
DTR1  = 001000
DTR2  = 002000
DTR3  = 004000
DT1     016770
DT2     017002
DT3     017020
DT4     017026
DT5     017044
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  011074
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     015444
EM11    015721
EM13    015757
EM14    016010
EM15    016042
EM16    016104
EM17    016156
EM2     015517
EM20    016214
EM21    016255
EM23    016305
EM24    016335
EM25    016363
EM26    016413
EM27    016442
EM3     015545
EM30    016510
EM4     015604
EM5     016633
EM6     015662
ERRMSG  006736
ERRVEC= 000004
ERTAB0  007112
EVEPAR= 000000
EXITER  007042
FALCIN  000570
FALCON  017126
FIVE  = 000000
FIVES = 000040
FRMERR= 020000
GETCSR= 002360
GETSWR= 007154
GETVEC= 002404
HALTS   006766
HDRFLG  001423
HDZVCS  002012
HDZVLP  002022
HDZVMS  002032
HDZVRB  002016
HDZVTC  002026
HDZVTD  002036
HILIM   006010
HT    = 000011
INBUF   010424
INIFLG  001422
INSTER= 104404
INSTR = 104403
INSTR2  005610
IOTVEC= 000020
KXTCHK  000400
KXTFLA  017142
LF    = 000012
LIMITS  005736
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  006014
LOCK    001364
LOLIM   006006
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  010041
MBADLN  010154
MCSRX   007771
MDATA   010466
MEPASS  007607
MERRPC  010121
MERRX   010016
MERR2   007647
MERR3   007716
MLOCK   007742
MNEW    010044
MNTFLG  001424
MODE    001372
MPASSX  010005
MPFAIL  007544
MR      007633
MSENAB= 000040
MTITLE  001000
MTSTN   010027
MVECX   007777
NEXT    001362
ODDPAR= 000200
OFFSET  015214
ONESTO= 000000
OUT     013520
OVRRUN= 040000
PAR     001370
PARAM = 104405
PARAM1  005656
PARER = 010000
PARERR  005732
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  006764
RESTAR  011070
RESVEC= 000010
RES05 = 104410
RIE   = 000100
RING0 = 000001
RING1 = 000002
RING2 = 000004
RING3 = 000010
RL0   = 000000
RL1   = 000400
RL2   = 001000
RL3   = 001400
RUN     001412
RXISR1  015012
RXSVC   013314
RXTCR   013572
R6    =%000006
R7    =%000007
SAVACT  001410
SAVLIN  001374
SAVNO   001416
SAVNUM  001415
SAVPC   001404
SAV05 = 104407
SCOP1 = 104401
SERV.G  007126
SETAPT  011236
SETFLG= 104406
SEVEN = 000020
SEVENS= 000060
SHIFT = 104420
SILOAL= 020000
SILOEN= 010000
SIX   = 000010
SIXS  = 000050
SNAP    013142
SPACNT  006267
STACK = 001120
STKLMT= 177774
SVOF    001446
SVO5    006024
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
TCR0  = 000001
TCR1  = 000002
TCR2  = 000004
TCR3  = 000010
TD0     001426
TD1     001430
TD2     001432
TD3     001434
TEIGHT  002106
TFIVE   002114
TIE   = 000100
TKVEC = 000060
TLAST = 014212
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    012070
TST2    012532
TST3    013032
TST4    013574
TST5    014212
TTST    004374
TWOSTO= 000040
TXSVC   013200
TXTCR   013312
TYPDAT  006752
TYPE  = 104402
TYPMSG  006642
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   002104
VECMAP  011700
WRDCNT  006264
WTBS.F  006740
XBX     006530
XCSR    004332
XERR    004354
XHEAD   010126
XMTCNT  001400
XMTLIN  001376
XMTSRV  014642
XPASS   004346
XSTATQ  010216
XTSTN   007120
XVEC    004340
XX    = 160210
YY    = 000500
ZZ    = 000020
$APTHD  001446
$ATYC   005250
$ATY1   005224
$ATY3   005232
$ATY4   005242
$AUTOB  001300
$BASE   001174
$BDADR  001266
$BDDAT  001272
$CDW1   001200
$CDW2   001202
$CHARC  005220
$CMTAG  001244
$CM1  = 000006
$CM2  = 000014
$CM3  = 000006
$CM4  = 000005
$CPUOP  001146
$CRLF   001357
$DDW0   001204
```

```
CVDZB-B MACY11 30G(1063)  10-AUG-81  11:11  PAGE 30-2
CVDZBB.P11    10-AUG-81 10:56          SYMBOL TABLE                                    SEQ 0084

$DDW1   001206      $ETABL  001140      $MFLG   005466      $SVLAD  004544      $Y    = 000023
$DDW10  001230      $ETEND  001244      $MSGAD  001134      $SVPC = 017146      $$GET4= 000000
$DDW11  001232      $FATAL  001122      $MSGLG  001136      $SWR  = 164000      .     = 017146
$DDW12  001234      $FFLG   005470      $MSGTY  001120      $SWREG  001142      .ADVAN  006366
$DDW13  001236      $FILLC  001322      $MTYP1  001151      $SWRMK= 000000      .BEGIN  004052
$DDW14  001240      $FILLS  001321      $MTYP2  001155      $TESTN  001124      .BUFSE  006456
$DDW15  001242      $FLIP = 177777      $MTYP3  001161      $TIMES  001354      .CNVRT  006114
$DDW2   001210      $FREE = 000002      $MTYP4  001165      $TKB    001312      .CONVR  006110
$DDW3   001212      $GDADR  001264      $MXCNT  004624      $TKS    001310      .DCLAS  006334
$DDW4   001214      $GDDAT  001270      $N    = 000005      $TMP0   001342      .DELAY  006346
$DDW5   001216      $GET42  004306      $NULL   001320      $TMP1   001344      .DEVIC  006314
$DDW6   001220      $HD   = 000001      $NWTST= 000000      $TMP2   001346      .ERRTA  015216
$DDW7   001222      $HIBTS  001446      $OVER   004562      $TMP3   001350      .INSTE  005576
$DDW8   001224      $ICNT   001250      $PASS   001126      $TMP4   001352      .INSTR  005472
$DDW9   001226      $ILLUP  007536      $PASTM  001454      $TN   = 000006      .INST1  005512
$DEVCT  001130      $INTAG  001301      $PWRAD  007532      $TPB    001316      .LPRSE  006416
$DEVM   001176      $ITEMB  001260      $PWRDN  007372      $TPFLG  001323      .MSG    005514
$DOAGN  004326      $LF     001360      $PWRMG  007526      $TPS    001314      .PARAM  005616
$E    = 000007      $LFLG   005467      $PWRUP  007444      $TSTM   001452      .PARRD  011234
$ENDAD  004316      $LPADR  001252      $QUES   001356      $TSTNM  001246      .PAWCH  010350
$ENDCT  004302      $LPERR  001254      $REGAD  001324      $TYPE   004670      .RESOS  006056
$ENV    001140      $MADR1  001152      $REG0   001326      $TYPEC  005102      .SAVOS  006016
$ENVM   001141      $MADR2  001156      $REG1   001330      $TYPEX  005222      .SCOPE  004362
$EOP    004146      $MADR3  001162      $REG2   001332      $UNIT   001132      .SCOP1  004626
$EOPCT  004274      $MADR4  001166      $REG3   001334      $UNITM  001456      .SETFL  010230
$ERFLG  001247      $MAIL   001120      $REG4   001336      $USWR   001144      .SHIFT  006400
$ERMAX  001261      $MAMS1  001150      $REG5   001340      $VECT1  001170      .START  002116
$ERROR  006500      $MAMS2  001154      $RTNAD  004330      $VECT2  001172      .TRPSR  006272
$ERRPC  001262      $MAMS3  001160      $SAVR6  007542      $XOFF = 000023      .TRPTA  001742
$ERRTB  001362      $MAMS4  001164      $SCOPE  004362      $XON  = 000021      .TYPE   004652
$ERTTL  001256      $MBADR  001450      $SETUP= 000000      $XTSTR  004422      .$X   = 001446

. ABS.  017146      000         OVR   RW    REL   GBL   I

 ERRORS DETECTED:  0

 CVDZBB,CVDZBB=CVDZBB
 RUN-TIME: 21 13 .7 SECONDS
 RUN-TIME RATIO: 195/35=5.5
 CORE USED:  37K  (73 PAGES)
```