DZV11

DIAG PRT2
CNDZBA0

AH-T442A-MC
FICHE 1 OF 1

```
        .REM    &
```

IDENTIFICATION

PRODUCT CODE: AC-T441A-MC

PRODUCT NAME: CNDZBA0 DZV11 DIAG PRT2

PRODUCT DATE: DEC, 1982

MAINTAINER: DIAGNOSTIC SERVICES/ISS

AUTHOR: L.FLORYAN

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 (CNDZA,CNDZB, AND CNDZC) ONE SYSTEM MODULE FOR DEC X/11 (DZBA), AND AN OVERLAY FOR ITEP (DVDZD).

CNDZA TOGETHER WITH CNDZB WILL TEST ALL LOGICAL FUNCTIONS OF THE DZV11 INTERFACE MODULE.
CNDZC 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. *
                                        SBC 11/21
                                        ---------
CSR RANGE:                          174000 TO 177770
VECTOR RANGE:                          300 TO    370
AUTO-SIZING FOR
CSR AND VECTOR:                        DISABLED
**************************************************************************
```

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, 1982.

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

                51      MISCELLANEOUS DEFINITIONS

                63      GENERAL PURPOSE REGISTER DEFINITIONS

                75      PRIORITY LEVEL DEFINITIONS

                85      "SWITCH REGISTER" SWITCH DEFINITIONS

                113     DATA BIT DEFINITIONS (BIT00 TO BIT15)

                141     BASIC "CPU" TRAP VECTOR ADDRESSES

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

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

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

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

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

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

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

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.

&
```

```
 8402                           ;;GPA   PRGFRT  ^?MAINDEC-11-DVDZBA/<200>/FOUR LINE ASYNC MUX TESTS, PART 2 OF 2?,DVDZBA
 8403                           ;;GPA   .HEADER <MD-11-DVDZB-A>,1982
  (2)                           .TITLE  CNDZB-A
  (2)                           ;*COPYRIGHT (C) 1982
  (2)                           ;*DIGITAL EQUIPMENT CORP.
  (2)                           ;*MAYNARD, MASS. 01754
  (2)                           ;*
  (2)                           ;*
  (2)                           ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
  (2)                           ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
  (2)                           ;*
  (2)          000001           $TN=1
  (2)                                   ;STARTING PROCEDURE
  (2)                                   ;LOAD PROGRAM
  (2)                                   ;LOAD ADDRESS 000200
  (2)                                   ;PRESS START
  (2)                                   ;PROGRAM WILL TYPE
  (2)                                   ;''CNDZBA/<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)                                 ;*****  THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
(2)          170000                 ODTST=  170000
(2)                                 ;*GENERAL PURPOSE REGISTER DEFINITIONS
(2)          000000                 R0=     %0              ;;GENERAL REGISTER
(2)          000001                 R1=     %1              ;;GENERAL REGISTER
(2)          000002                 R2=     %2              ;;GENERAL REGISTER
(2)          000003                 R3=     %3              ;;GENERAL REGISTER
(2)          000004                 R4=     %4              ;;GENERAL REGISTER
(2)          000005                 R5=     %5              ;;GENERAL REGISTER
(2)          000006                 R6=     %6              ;;GENERAL REGISTER
(2)          000007                 R7=     %7              ;;GENERAL REGISTER
(2)          000006                 SP=     %6              ;;STACK POINTER
(2)          000007                 PC=     %7              ;;PROGRAM COUNTER
(2)
(2)                                 ;*PRIORITY LEVEL DEFINITIONS
(2)          000000                 PR0=    0               ;;PRIORITY LEVEL 0
(2)          000040                 PR1=    40              ;;PRIORITY LEVEL 1
(2)          000100                 PR2=    100             ;;PRIORITY LEVEL 2
(2)          000140                 PR3=    140             ;;PRIORITY LEVEL 3
(2)          000200                 PR4=    200             ;;PRIORITY LEVEL 4
(2)          000240                 PR5=    240             ;;PRIORITY LEVEL 5
(2)          000300                 PR6=    300             ;;PRIORITY LEVEL 6
(2)          000340                 PR7=    340             ;;PRIORITY LEVEL 7
(2)
(2)                                 ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
(2)          100000                 SW15=   100000
(2)          040000                 SW14=   40000
(2)          020000                 SW13=   20000
(2)          010000                 SW12=   10000
(2)          004000                 SW11=   4000
(2)          002000                 SW10=   2000
(2)          001000                 SW09=   1000
(2)          000400                 SW08=   400
(2)          000200                 SW07=   200
(2)          000100                 SW06=   100
(2)          000040                 SW05=   40
(2)          000020                 SW04=   20
(2)          000010                 SW03=   10
(2)          000004                 SW02=   4
(2)          000002                 SW01=   2
(2)          000001                 SW00=   1
(2)                                 .EQUIV  SW09,SW9
(2)                                 .EQUIV  SW08,SW8
(2)                                 .EQUIV  SW07,SW7
(2)                                 .EQUIV  SW06,SW6
(2)                                 .EQUIV  SW05,SW5
(2)                                 .EQUIV  SW04,SW4
(2)                                 .EQUIV  SW03,SW3
(2)                                 .EQUIV  SW02,SW2
```

```
(2)                                 .EQUIV  SW01,SW1
(2)                                 .EQUIV  SW00,SW0
(2)
(2)                                 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(2)             100000              BIT15=  100000
(2)             040000              BIT14=  40000
(2)             020000              BIT13=  20000
(2)             010000              BIT12=  10000
(2)             004000              BIT11=  4000
(2)             002000              BIT10=  2000
(2)             001000              BIT09=  1000
(2)             000400              BIT08=  400
(2)             000200              BIT07=  200
(2)             000100              BIT06=  100
(2)             000040              BIT05=  40
(2)             000020              BIT04=  20
(2)             000010              BIT03=  10
(2)             000004              BIT02=  4
(2)             000002              BIT01=  2
(2)             000001              BIT00=  1
(2)                                 .EQUIV  BIT09,BIT9
(2)                                 .EQUIV  BIT08,BIT8
(2)                                 .EQUIV  BIT07,BIT7
(2)                                 .EQUIV  BIT06,BIT6
(2)                                 .EQUIV  BIT05,BIT5
(2)                                 .EQUIV  BIT04,BIT4
(2)                                 .EQUIV  BIT03,BIT3
(2)                                 .EQUIV  BIT02,BIT2
(2)                                 .EQUIV  BIT01,BIT1
(2)                                 .EQUIV  BIT00,BIT0
(2)
(2)                                 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(2)             000004              ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(2)             000010              RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(2)             000014              TBITVEC=14              ;;"T" BIT
(2)             000014              TRTVEC= 14              ;;TRACE TRAP
(2)             000014              BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(2)             000020              IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(2)             000024              PWRVEC= 24              ;;POWER FAIL
(2)             000030              EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(2)             000034              TRAPVEC=34              ;;"TRAP" TRAP
(2)             000060              TKVEC=  60              ;;TTY KEYBOARD VECTOR
(2)             000064              TPVEC=  64              ;;TTY PRINTER VECTOR
(2)                                 ;*****  THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(2)             000100              LKVEC=  100             ;;LINE CLOCK VECTOR
(2)             000140              BRKVEC= 140             ;;BREAK 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    004354                  .SCOPE                  ;SCOPE LOOP HANDLER
(1)   000022    000200                  MASK                    ;HANDLE AT PRIORITY 7
(1)   000024    007312                  $PWRDN                  ;POWER FAIL HANDLER
(1)   000026    000300                  300                     ;SERVICE AT PRIORITY LEVEL 6 VER:0
(1)   000030    006420                  $ERROR                  ;ERROR HANDLER
(1)   000032    000300                  300                     ;SERVICE AT PRIORITY LEVEL 6 VER:0
(1)   000034    006212                  .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
(1)   000036    000300                  300                     ;SERVICE AT PRIORITY LEVEL 6 VER:0
(2)                             .SBTTL  ACT11 HOOKS
(2)
(3)                             ;;*****************************************************************************
(2)                             ;HOOKS REQUIRED BY ACT11
(2)             000040                  $SVPC=.                 ;SAVE PC
(2)             000046                  .=46
(2)   000046    004310                  $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  047103  055104  MTITLE: .ASCIZ  <200><12>/CNDZBA/<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
(4)                                           ;;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
(4)                                           ;;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)
```

COMMON TAGS


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

```
(3)                                 .SBTTL  ERROR POINTER TABLE
(3)
(3)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(3)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(3)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(3)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(3)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(3)
(3)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(3)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
(3)                                 ;*      DT              ;;POINTS TO THE DATA
(3)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
(3)
(3)
(3)  001362                         $ERRTB:
(2)
(2)                                         ;PROGRAM CONTROL PARAMETERS
(2)                                         ;--------------------------
(2)
(2)  001362  000000                 NEXT:   0                       ;ADDRESS OF NEXT TEST TO BE EXECUTED
(2)  001364  000000                 LOCK:   0                       ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP
(2)
(2)                                         ;PROGRAM VARIABLES
(2)                                         ;-----------------
(2)
(2)  001366  000017                 LINE:   17                      ;DEFAULT ALL FOUR LINES RUNNING
(2)  001370  017470                 PAR:    17470                   ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NO PARIT
(2)  001372  000000                 MODE:   0                       ;DEFAULT MAINTENANCE MODE
(2)  001374  000000                 SAVLIN: 0                       ;LINE NUMBER
(2)  001376  000000                 XMTLIN: 0                       ;TRANSMISSION LINE NUMBER
(2)  001400  000000                 XMTCNT: 0                       ;COUNT OF WORDS IN A TRANSMISSION PATTERN
(2)  001402  000000                 REGIST: 0                       ;DEVICE ADDRESS STORAGE LOCATION
(2)  001404  000000                 SAVPC:  0                       ;PROGRAM COUNTER STORAGE
(2)  001406  000001                 DZVACTV:.BLKW   1               ;*DZV11'S SELECTED ACTIVE.
(2)  001410  000001                 SAVACTV:.BLKW   1               ;*A BIT MAP OF DZV11'S IN THE SYSTEM
(2)  001412  000001                 RUN:    1                       ;*POINTER ONE PAST RUNNING DEVICE.
(2)  001414  000001                 DZVNUM: .BLKB 1                 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM.
(2)  001415     001                 SAVNUM: .BYTE 1                 ;*WORKABLE NUMBER.
(2)  001416  000001                 SAVNO:  .BLKB   1               ;*OCTAL 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)                                     ;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  006306                             .ADVANCE
(3)          104401                     SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
(2)  001744  004620                             .SCOP1
(3)          104402                     TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
(2)  001746  004644                             .TYPE
(3)          104403                     INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
(2)  001750  005412                             .INSTR
(3)          104404                     INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
(2)  001752  005516                             .INSTER
(3)          104405                     PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
(2)  001754  005536                             .PARAM
(3)          104406                     SETFLG=TRAP+6           ;CALL TO  SET FLAG ROUTINE
(2)  001756  010150                             .SETFLG
(3)          104407                     SAVU5=TRAP+7            ;CALL TO REGISTER SAVE ROUTINE
(2)  001760  005736                             .SAV05
(3)          104410                     RES05=TRAP+10           ;CALL TO REGISTER RESTORE ROUTINE
(2)  001762  005776                             .RES05
(3)          104411                     CONVRT=TRAP+11          ;CALL TO DATA OUTPUT ROUTINE
(2)  001764  006030                             .CONVRT
(3)          104412                     CNVRT=TRAP+12           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
(2)  001766  006034                             .CNVRT
(3)          104413                     DEVICE.CLR=TRAP+13              ;CALL TO ISSUE A DEVICE CLEAR
(2)  001770  006234                             .DEVICE.CLR
(3)          104414                     DELAY=TRAP+14           ;CALL TO DELAY FOR FAST CPU'S
(2)  001772  006266                             .DELAY
(3)          104415                     PARMD=TRAP+15           ;CONVERT DECIMAL STRING TO OCTAL
(2)  001774  011154                             .PARMD
(3)          104416                     PAWCH=TRAP+16           ;SET FLAG    ECHO OR  CABLE
(2)  001776  010270                             .PAWCH
(3)          104417                     DCLASM=TRAP+17          ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2)  002000  006254                             .DCLASM
(3)          104420                     SHIFT=TRAP+20           ;CALL TO ROTATE LINE POINTER
(2)  002002  006320                             .SHIFT
(3)          104421                     LPRSET=TRAP+21          ;CALL TO SET UP LPR DEVICE REGISTER
(2)  002004  006336                             .LPRSET
(3)          104422                     BUFSET=TRAP+22          ;CALL TO ZERO BUFFER AREA
(2)  002006  006376                             .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  007312  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)                                 ;;THE FOLLOWING TWO LINES DELETED TO BE FALCON SPECIFIC.
(1)                                 ;       CALL    FALCON          ; CHECK FOR FALCON (KXT11)      ;;GPA
(1)                                 ;       BEQ     1000$           ; BR IF NOT                     ;;GPA
(1)  002220  004737  017046                 CALL    FALCINI         ; YES, INIT FOR FALCON.         ;;GPA
(1)  002224                         1000$:                                                          ;;GPA
(1)  002224  105737  001422                 TSTB    INIFLG          ;HAVE WE ALREADY BEEN HERE TODAY?
(1)  002230  001010                         BNE     10$             ;IF SO, SKIP PRINTING THE TITLE
(1)  002232  023727  000042  004310         CMP     @#42,#$ENDAD    ;IF RUNNING UNDER ACT
(1)  002240  001402                         BEQ     1$              ;DON'T PRINT TITLE
(1)  002242  104402  001000                 TYPE    ,MTITLE         ;PRINT THE DIAGNOSTIC'S TITLE
(1)  002246  105337  001422         1$:     DECB    INIFLG          ;SET THE ONCE ONLY FLAG
(1)  002252  105737  001141         10$:    TSTB    $ENVM           ;DETERMINE WHETHER APT SIZING SHOULD BE DONE
(1)  002256  100004                         BPL     15$             ;IF NOT, GO CHECK FOR AUTO-SIZING
(1)  002260  004737  011156                 JSR     PC,SETAPT       ;OTHERWISE, GO DO APT SIZING FROM ETABLE
(1)  002264  000137  003600                 JMP     105$            ;GO PRINT DZV STATUS TABLE
(1)  002270  005737  000042         15$:    TST     @#42            ; CHAINED UNDER XXDP ??         ;;GPA
(1)  002274  001404                         BEQ     16$             ; BR IF NOT                     ;;GPA
(1)  002276  004737  011156                 CALL    SETAPT          ; YES, SET-UP FROM ETABLE       ;;GPA
(1)  002302  000137  003600                 JMP     105$            ; AND PROCEED                   ;;GPA
(1)  002306  004737  007074         16$:    CALL    GETSWR          ; GET AN INITIAL SWR            ;;GPA
(1)  002312  032777  000001  176764         BIT     #SW00,@SWR      ;RESELECT ?
(1)  002320  001002                         BNE     20$             ;IF YES, GO SET UP THE INFORMATION
(1)  002322  000137  002624                 JMP     55$             ;IF NO, SKIP THE INTERROGATION
(1)  002326  012700  001500         20$:    MOV     #DZV.MAP,R0     ;POINT TO THE BEGINNING OF THE MAP TABLE
(1)  002332  105037  001423                 CLRB    HDRFLG          ;MAKE SURE A MAP GETS PRINTED
(1)  002336  005020                 25$:    CLR     (R0)+           ;CLEAR A TABLE LOCATION
(1)  002340  020027  001740                 CMP     R0,#DZV.END     ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  002344  001374                         BNE     25$             ;IF NOT ,CLEAR THE NEXT LOCATION IN THE TABLE
(1)  002346  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)             002352                          GETCSR= .               ; POINTER FOR FALCON TWEAKER.   ;;GPA
(2)  002352     104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002354     003044                          91$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002356     104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002360     160000                          160000                  ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002362     163770                          163770                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002364     001500                          DZCR0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002366        007                          .BYTE   7               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002367        001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002370     013737  001500  001174          MOV     DZCR0,$BASE     ;COPY BASE ADDRESS TO ETABLE
(1)
(1)                                             ;GET THE BASE VECTOR ADDRESS
(1)             002376                          GETVEC= .               ; POINTER FOR FALCON TWEAKER.   ;;GPA
(2)  002376     104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002400     003110                          92$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002402     104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002404     000300                          300                     ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002406     000776                          776                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002410     001502                          DZVC0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002412        003                          .BYTE   3               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002413        001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002414     013737  001502  001170          MOV     DZVC0,$VECT1    ;COPY VECTOR TO ETABLE
(1)
(1)                                             ;GET THE MODE OF OPERATION (E,I,S)
(2)  002422     104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002424     003337                          96$                     ;POINTER TO THE MESSAGE TO BE PRINTED
(2)  002426     104406                          SETFLG                  ;CALL THE MAINTENANCE FLAG SETUP ROUTINE
(2)  002430     001510                          MANT0                   ;THIS IS THE FLAG BEING SETUP
(1)
(1)                                             ;GET THE NUMBER OF DZV11'S RUNNING
(2)  002432     104403                          INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002434     003274                          95$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002436     104405                          PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002440     000001                          1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002442     000020                          16.                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002444     001344                          $TMP1                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002446        000                          .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002447        001                          .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)
(1)  002450     012737  000017  001504          MOV     #17,LINE0       ;SET UP DEFAULT LINES
(1)  002456     012737  017470  001506          MOV     #17470,PAR0     ;SET UP DEFAULT LPR PARAMETER
(1)                                             ;RECEIVER ON; 19.2 KBAUD; 2STOP BITS; 8 BIT/CHAR
(1)  002464     032777  000010  176612          BIT     #SW03,@SWR      ;DO YOU WANT PARAMETERS?
(1)  002472     001402                          BEQ     30$             ;IF NO, SKIP THE PARAMETER CALL
(1)  002474     004737  002654                  JSR     PC,65$          ;GET PARAMETERS
(1)  002500     012737  000001  001410  30$:    MOV     #1,SAVACTV      ;INITIALIZE ACTIVE DEVICE SELECTION PARAMETER
(1)  002506     113737  001344  001414          MOVB    $TMP1,DZVNUM    ;COPY THE NUMBER OF DEVICES
(1)  002514     005337  001344          35$:    DEC     $TMP1           ;$TMP1 CONTAINS THE COUNT OF UNINITIALIZED
(1)  002520     001404                          BEQ     40$             ; SELECTED DEVICES
(1)  002522     000261                          SEC                     ;SET A BIT FLAG TO INDICATE AN ACTIVE DEVICE
(1)  002524     006137  001410                  ROL     SAVACTV         ;POINT TO THE NEXT DEVICE
(1)  002530     000771                          BR      35$             ;GO DO THIS PROCEDURE AGAIN
(1)  002532     013737  001410  001346  40$:    MOV     SAVACTV,$TMP2   ;# OF TIMES
```

```
(1)  002540  012700  001500                MOV     #DZCR0,R0       ;SET A POINTER TO THE SPECIFIED INFORMATION
(1)  002544  012701  001512                MOV     #DZCR1,R1       ;POINT R1 TO THE REST OF THE MAP TABLE
(1)  002550  012702  001204                MOV     #$DDW0,R2       ;POINT TO ETABLE'S DEVICE DESCRIPTOR WORDS
(1)  002554  000241                        CLC                     ;INITIALIZE THE "C" BIT FOR A ROTATION
(1)  002556  006037  001346                ROR     $TMP2           ;SKIP MAPPING SETUP FOR DEVICE 0- IT'S DONE
(1)  002562  006237  001346          45$:  ASR     $TMP2           ;ISOLATE A SELECTION FLAG IN THE "C" BIT
(1)  002566  103404                        BCS     50$             ;IS THIS DEVICE SELECTED? IF YES, GO LOAD TABLE
(1)  002570  012711  177777                MOV     #-1,(R1)        ;TERMINATE THE LIST
(1)  002574  000137  003542                JMP     100$            ;GO TO THE NEXT BLOCK
(1)  002600  012011                  50$:  MOV     (R0)+,(R1)      ;ADDRESS
(1)  002602  062721  000010                ADD     #10,(R1)+       ;POINT TO THE NEXT DZV11 ADDRESS VALUE
(1)  002606  012011                        MOV     (R0)+,(R1)      ;VECTOR
(1)  002610  062721  000010                ADD     #10,(R1)+       ;POINT TO THE NEXT VECTOR VALUE
(1)  002614  012021                        MOV     (R0)+,(R1)+     ;LINES
(1)  002616  012021                        MOV     (R0)+,(R1)+     ;PARAMETERS
(1)  002620  012021                        MOV     (R0)+,(R1)+     ;MAINTENANCE MODE
(1)  002622  000757                        BR      45$
(1)  002624  032777  000010  176452  55$:  BIT     #SW03,@SWR      ;ASK PARAMETERS ?
(1)  002632  001002                        BNE     60$             ;IF NO, GO DO AUTO SIZING
(1)  002634  000137  003542                JMP     100$            ;GO SET UP FOR AUTO SIZING
(1)  002640  004737  002654          60$:  JSR     PC,65$          ;GO ASK PARAMETERS
(1)  002644  105337  001422                DECB    INIFLG          ;INSURE NO AUTO SIZE IF QUESTIONS ANSWERED
(1)  002650  000137  003600                JMP     105$            ;GO TO THE NEXT BLOCK
(1)
(1)                                        ;GET THE ACTIVE LINES PARAMETER
(1)
(1)  002654                          65$:
(2)  002654  104403                        INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002656  003151                        93$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002660  104405                        PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002662  000001                        1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002664  000017                        17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002666  001504                        LINE0                   ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002670     360                        .BYTE   360             ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002671     001                        .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002672  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)  002676  005737  001510                TST     MANTO           ;IS STAGGERED THE MODE OF OPERATION?
(1)  002702  100021                        BPL     85$             ;IF NOT, SKIP THIS SEGMENT
(1)  002704  013703  001504                MOV     LINE0,R3        ;GET A SCRATCH COPY OF THE ACTIVE LINES
(1)  002710  006003                  70$:  ROR     R3              ;GET A LINE SELECTION BIT(EVEN NUMBER LINE)
(1)  002712  103410                        BCS     80$             ;IF IT IS SELECTED, CHECK TO SEE IF THE NEXT IS TOO
(1)  002714  001414                        BEQ     85$             ;IF ALL HAVE BEEN CHECKED, CONTINUE PROCESSING
(1)  002716  006203                        ASR     R3              ;IF IT IS 0,CHECK TO SEE IF THE NEXT IS TOO
(1)  002720  103373                        BCC     70$             ;IF THIS ONE'S 0 TOO, GO CHECK THE NEXT PAIR
(1)  002722  104402  001356          75$:  TYPE    ,$QUES          ;THIS IS AN INCORRECT PARAMETER
(1)  002726  104402  010074                TYPE    ,MBADLN         ;LET THE USER KNOW ABOUT IT
(1)  002732  000750                        BR      65$             ;GO GET THE CORRECT PARAMETER
(1)  002734  001772                  80$:  BEQ     75$             ;IF ANOTHER FLAG ISN'T SET, THERE'S AN ERROR
(1)  002736  006203                        ASR     R3              ;GET THE NEXT FLAG
(1)  002740  103370                        BCC     75$             ;IF IT ISN'T SET, THERE'S AN ERROR
(1)  002742  000241                        CLC                     ;INITIALIZE THE "C" BIT FOR TESTING OF THE NEXT PAIR
(1)  002744  000761                        BR      70$             ;GO TEST THE NEXT PAIR OF FLAGS
```

```
(1)
(1)                                             ;GET THE LINE PARAMETER REGISTER ARGUMENT
(1)
(1)  002746                             85$:
(2)  002746  104403                             INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  002750  003224                             94$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  002752  104405                             PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  002754  000000                             0                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002756  000017                             17                      ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  002760  001506                             PAR0                    ;POINTER TO MAP LOCATION TO BE FILLED
(2)  002762     000                             .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  002763     001                             .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  002764  012702  001504                     MOV     #LINE0,R2       ;POINT TO THE LINE SELECTION PARAMETER
(1)  002770  012703  001506                     MOV     #PAR0,R3        ;POINT TO THE CHOSEN PARAMETERS
(1)  002774  011304                             MOV     (R3),R4         ;USE BAUD RATE AS AN INDEX IN DELAY TABLE
(1)  002776  006304                             ASL     R4              ;ALIGN INDEX ON WORD BOUNDARY
(1)  003000  016437  017006  006304             MOV     DLYTBL(R4),DLYCNT ;SET THE DELAY COUNT FOR THIS BAUD RATE
(1)  003006  000313                             SWAB    (R3)            ;PLACE IN HIGH BYTE
(1)  003010  052713  010070                     BIS     #10070,(R3)     ;PLACE EXTRA PARAMETERS INTO LOC
(1)  003014  011262  000012             90$:    MOV     (R2),12(R2)     ;LOAD THE LINES
(1)  003020  011363  000012                     MOV     (R3),12(R3)     ;LOAD THE PARAMETERS
(1)  003024  062702  000012                     ADD     #12,R2          ;POINT TO THE NEXT SET
(1)  003030  062703  000012                     ADD     #12,R3          ; .. OF BOTH PARAMETERS
(1)  003034  020327  001734                     CMP     R3,#PAR17       ;HAVE THE TABLE BOUNDARIES BEEN EXCEEDED?
(1)  003040  001365                             BNE     90$             ;IF NOT, GO LOAD SOME MORE PARAMETERS
(1)  003042  000207                             RTS     PC              ;RETURN TO CALLING BLOCK
(1)  003044  030600  052123  041440  91$:       .ASCIZ  <200>/1ST CSR ADDRESS (160000:163770):  /
(1)  003110  030600  052123  053040  92$:       .ASCIZ  <200>/1ST VECTOR ADDRESS (300:770):  /
(1)  003151     200  044514  042516  93$:       .ASCIZ  <200>/LINES ACTIVE BY BIT <IN OCTAL>(001:17):  /
(1)  003224  042200  043105  052501  94$:       .ASCIZ  <200>/DEFAULT BAUD RATE <IN OCTAL>(00:17):  /
(1)  003274  021600  047440  020106  95$:       .ASCIZ  <200>/# OF DZV11'S <IN OCTAL> (1:20):  /
(1)  003337     200  040515  047111  96$:       .ASCII  <200>/MAINTENANCE MODE/
(1)  003360  020200  042533  052130              .ASCII  <200>/ [EXTERNAL  <H325>     (E)]/
(1)  003414  020200  044533  052116              .ASCII  <200>/ [INTERNAL  <DZVCSR03=1>(I)]/
(1)  003451     200  055440  052123              .ASCIZ  <200>/ [STAGGERED <H329>     (S)]: /
(1)  003510  042600  052116  051105  97$:       .ASCIZ  <200>/ENTER DELAY PARAMETER: /
(1)          003542                             .EVEN
(1)  003542                             100$:
(1)  003542  122737  000377  001422              CMPB    #377,INIFLG     ;ONLY DO AUTO SIZE ON 1ST START
(1)  003550  001013                             BNE     105$            ;
(1)  003552  032777  000200  175524              BIT     #BIT7,@SWR      ;BIT7=1??
(1)  003560  001007                             BNE     105$            ;BR IF NO AUTO SIZE
(1)  003562  005737  017324                     TST     KXTFLAG         ; KXT11 ??                              ::GPA
(1)  003566  001402                             BEQ     1001$           ; SKIP NEXT IF NOT                      ::GPA
(1)  003570  000137  002326                     JMP     20$             ; YES, DON'T AUTO-SIZE                  ::GPA
(1)  003574                             1001$:                                                                  ::GPA
(1)  003574  004737  011304                     JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
(1)  003600  105737  001423             105$:   TSTB    HDRFLG          ;HAS THE TABLE BEEN TYPED YET?
(1)  003604  001021                             BNE     120$            ;IF SO, DON'T TYPE IT AGAIN
(1)  [illegible]                                [illegible]             ;INDICATE THAT THE TABLE WILL BE TYPED
(1)  [illegible]                                [illegible]             ;TYPE MAP HEADER
(1)  003616  012700  001500                     MOV     #DZV.MAP,R0     ;SET POINTER
(1)  003622  010027  001344             110$:   MOV     R0,$TMP1        ;POINT TO THE MAP LOCATION
(1)  003626  [illegible]  001346                MOV     (R0)+,$TMP2     ;SET DATA
(1)  003632  022737  177777  001346             CMP     #-1,$TMP2       ;END OF LIST?
(1)  003640  001403                             BEQ     120$            ;BR IF YES
```

```
(1)  003642  104411                  115$:  CONVRT                  ;CALL THE OCTAL TO ASCII CONVERSION ROUTINE
(1)  003644  010136                         XSTATQ                  ;CONVERT THE DATA AT THIS ADDRESS
(1)  003646  000765                         BR      110$            ;GO PRINT THE NEXT PARAMETER
(1)  003650  013737  001410  001406  120$:  MOV     SAVACTV,DZVACTV ;COPY BIT MAP OF ACTIVE DEVICES
(1)  003656  113737  001414  001416        MOVB    DZVNUM,SAVNO    ;COPY NO. OF DEVICES IN THE SYSTEM
(1)  003664  032777  000100  175412        BIT     #SW06,@SWR      ;DESELECT SPECIFIC DEVICES??
(1)  003672  001431                         BEQ     135$            ;BR IF NO.
(1)  003674                          121$:
(2)  003674  104403                         INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003676  007764                         MNEW                    ;POINTER TO MESSAGE TO BE PRINTED
(2)  003700  104405                         PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  003702  000001                         1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003704  177777                         177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003706  001406                         DZVACTV                 ;POINTER TO MAP LOCATION TO BE FILLED
(2)  003710     000                         .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  003711     001                         .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  003712  023737  001406  001410        CMP     DZVACTV,SAVACTV ;IS VALUE VALID?
(1)  003720  101403                         BLOS    122$            ;IF YES BRANCH
(1)  003722  104402  007636                 TYPE    ,MERR3          ;IF NOT TYPE ERROR
(1)  003726  000762                         BR      121$            ;THEN REASK QUESTION
(1)  003730  105037  001416          122$:  CLRB    SAVNO           ;INITIALIZE NO. OF ACTIVE DEVICES
(1)  003734  013737  001406  001344        MOV     DZVACTV,$TMP1   ;COPY BIT MAP OF ACTIVE DEVICES
(1)  003742  006237  001344          126$:  ASR     $TMP1           ;ROTATE OUT AN ACTIVE BIT
(1)  003746  103002                         BCC     127$            ;IF NOT ACTIVE SKIP RECORDING IT
(1)  003750  105237  001416                 INCB    SAVNO           ;INCREMENT NO. OF ACTIVE DEVICES
(1)  003754  001372                  127$:  BNE     126$            ;IF NOT DONE GO CONTINUE
(1)  003756  032777  000020  175320  135$:  BIT     #SW04,@SWR      ;CHECK TO SEE IF DELAY COUNT CHANGES
(1)  003764  001407                         BEQ     140$            ;IF NOT, GO CLEAR VECTOR AREA
(2)  003766  104403                         INSTR                   ;CALL THE STRING INPUT ROUTINE
(2)  003770  003510                         97$                     ;POINTER TO MESSAGE TO BE PRINTED
(2)  003772  104405                         PARAM                   ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(2)  003774  000001                         1                       ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  003776  177777                         177777                  ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(2)  004000  006304                         DLYCNT                  ;POINTER TO MAP LOCATION TO BE FILLED
(2)  004002     000                         .BYTE   0               ;MASK OF INVALID BITS FOR THIS PARAMETER
(2)  004003     001                         .BYTE   1               ;NUMBER OF PARAMETERS TO STORE
(1)  004004  012700  000300          140$:  MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
(1)  004010  012701  000302                 MOV     #302,R1         ;VECTOR AREA. 300-776
(1)  004014  010120                  145$:  MOV     R1,(R0)+        ;START PUTTING 'PC+2 - HALT"
(1)  004016  005021                         CLR     (R1)+           ;IN VECTOR AREA.
(1)  004020  022021                         CMP     (R0)+,(R1)+     ;POP POINTERS
(1)  004022  005737  017324                 TST     KXTFLAG         ; IF FALCON...                  ;;GPA
(1)  004026  001403                         BEQ     1002$           ;                               ;;GPA
(1)  004030  020027  000400                 CMP     R0,#400         ;...QUIT AT 400.                ;;GPA
(1)  004034  000402                         402                     ; SKIP NEXT                     ;;GPA
(1)  004036                          1002$:                         ;                               ;;GPA
(1)  004036  022700  001000                 CMP     #1000,R0        ;ALL DONE??
(1)  004042  001364                         BNE     145$            ;BR IF NO.
(1)
(1)                                  ;TEST START AND RESTART
(1)                                  ;----------------------
(1)
(1)  004044  012706  001120          .BEGIN: MOV    #STACK,SP       ;SET UP STACK
(1)  004050  106427  000200                 MTPS    #MASK           ;LOCK OUT INTERRUPTS
(1)  004054  005737  000042                 TST     @#42            ;IS PROGRAM UNDER MONITOR CONTROL
(1)  004060  001015                         BNE     2$              ;BR IF YES
```

```
   (1)  004062  032777  000004  175214          BIT     #BIT2,@SWR      ;CHECK FOR LOCK ON TEST
   (1)  004070  001406                          BEQ     1$              ;BR IF NO LOCK DESIRED.
   (1)  004072  104402  007662                  TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
   (1)  004076  012737  000240  004366          MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
   (1)  004104  000403                          BR      2$              ;CONTINUE ALONG.
   (1)  004106  013737  004614  004366  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
   (1)  004114  012737  010450  001252  2$:     MOV     #CYCLE,$LPADR   ;START AT ''CYCLE'' FIND WHICH DEVICE TO TEST
   (1)  004122  113737  001416  001415          MOVB    SAVNO,SAVNUM    ;COPY NO. OF ACTIVE DEVICES
   (1)  004130  104402  007553                  TYPE    ,MR             ;TYPE ''RUNNING''
   (1)  004134  000177  175112                  JMP     @$LPADR         ;START TESTING
  8404                              ;;GPA      PRGEND  DZV11,<END PASS DVDZB-A  >,10.
```

END OF PASS ROUTINE

SEQ 0042

```
 8405
  (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)  004140                           $EOP:
  (5)  004140  000004                           SCOPE
  (5)  004142  005037  001262                   CLR     $ERRPC          ;CLEAR LAST ERROR PC
  (5)  004146  105037  001247                   CLRB    $ERFLG          ;CLEAR ERROR FLAG
  (5)  004152  104402  007527                   TYPE    ,MEPASS         ;TYPE END PASS
  (5)  004156  104402  007711                   TYPE    ,MCSRX          ;TYPE CSR
  (5)  004162  104412  004324                   CNVRT   ,XCSR           ;SHOW IT
  (5)  004166  104402  007717                   TYPE    ,MVECX          ;TYPE VECTOR
  (5)  004172  104412  004332                   CNVRT   ,XVEC           ;SHOW IT
  (5)  004176  005237  001126                   INC     $PASS           ;RAISE PASS COUNT
  (5)  004202  104402  007725                   TYPE    ,MPASSX         ;TYPE PASSES
  (5)  004206  104412  004340                   CNVRT   ,XPASS          ;SHOW IT
  (5)  004212  005337  001126                   DEC     $PASS           ;RESTORE PASS COUNT
  (5)  004216  104402  007736                   TYPE    ,MERRX          ;TYPE ERRORS
  (5)  004222  104412  004346                   CNVRT   ,XERR           ;SHOW IT
  (5)  004226  005237  001130                   INC     $DEVCT          ;INC DEVCNT FOR APT
  (5)  004232  105337  001415                   DECB    SAVNUM          ;ARE ALL DEVICES TESTED?
  (5)  004236  001030                           BNE     $DOAGN          ;BR IF NO.
  (5)  004240  113737  001416  001415          MOVB    SAVNO,SAVNUM    ;RESTORE THE COUNT
  (3)  004246  005037  001354                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (3)  004252  005237  001126                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (3)  004256  042737  100000  001126          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (3)  004264  005327                           DEC     (PC)+           ;;LOOP?
  (3)  004266  000001                   $EOPCT: .WORD   1
  (3)  004270  003013                           BGT     $DOAGN          ;;YES
  (3)  004272  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (3)  004274  000001                   $ENDCT: .WORD   1
  (3)  004276  004266                           $EOPCT
  (3)  004300  013700  000042           $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (3)  004304  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (3)  004306  000005                           RESET                   ;;CLEAR THE WORLD
  (3)  004310  004710                   $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (3)  004312  000240                           NOP                     ;;SAVE ROOM
  (3)  004314  000240                           NOP                     ;;FOR
  (3)  004316  000240                           NOP                     ;;ACT11
  (3)  004320                           $DOAGN:
  (3)  004320  000137                           JMP     @(PC)+          ;;RETURN
  (3)  004322  010450                   $RTNAD: .WORD   CYCLE
  (3)
  (3)
  (2)
  (2)  004324  000001                   XCSR:   1
  (2)  004326     006       002                 .BYTE   6,2
  (2)  004330  002010                           DZVCSR
```

```
(2)  004332  000001                  XVEC:   1
(2)  004334     003      002                 .BYTE   3,2
(2)  004336  002040                          DZVRIV
(2)  004340  000001                  XPASS:  1
(2)  004342     006      002                 .BYTE   6,2
(2)  004344  001126                          $PASS
(2)  004346  000001                  XERR:   1
(2)  004350     006      002                 .BYTE   6,2
(2)  004352  001256                          $ERTTL
(2)
(2)                                          ;SCOPE LOOP AND ITERATION HANDLER
(2)                                          ;-------------------------------
(2)
(3)                                  .SBTTL  SCOPE HANDLER ROUTINESTARS
(3)                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(3)                                  ;*AND LOAD THE TEST NUMBER($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)  004354                          $SCOPE:
(5)  004354  005037  001262          .SCOPE: CLR     $ERRPC          ;CLEAR LAST ERROR PC.
(5)  004360  022716  012012                  CMP     #TST1+2,(SP)    ;IS THIS THE SCOPE AT THE BEGINNING OF TST1?
(5)  004364  001413                          BEQ     $XTSTR          ;IF SO, DON'T LOOP ON IT
(5)  004366  000406                  TTST:   BR      1$              ;GOTO 1$    (IF LOCK SW02=1; THIS LOC =240)
(5)  004370  105777  174714                  TSTB    @$TKS           ;KEYBOARD DONE?
(5)  004374  100067                          BPL     $OVER           ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
(5)  004376  017766  174710  177776          MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
(3)  004404  032777  040000  174672  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(3)  004412  001060                          BNE     $OVER           ;;YES IF SW14=1
(3)                                  ;#####START OF CODE FOR THE XOR TESTER#####
(3)  004414  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3)  004416  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3)  004422  012737  004442  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(3)  004430  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
(3)  004434  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004440  000436                          BR      $SVLAD          ;;GO TO THE NEXT TEST
(3)  004442  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(3)  004444  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  004450  000441                          BR      $OVER           ;;LOOP ON THE PRESENT TEST
(3)  004452                          6$:;#####END OF CODE FOR THE XOR TESTER#####
(3)  004452  105737  001247          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(3)  004456  001404                          BEQ     3$              ;;BR IF NO
(3)  004460  105037  001247          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(3)  004464  005037  001354                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3)  004470  032777  004000  174606  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(3)  004476  001011                          BNE     1$              ;;BR IF YES
(3)  004500  005737  001126                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(3)  004504  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
(3)  004506  005237  001250                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
(3)  004512  023737  001354  001250          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(3)  004520  002015                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
```

```
(3)  004522  012737  000001  001250  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
(3)  004530  013737  004616  001354          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
(3)  004536  105237  001246          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
(3)  004542  113737  001246  001124          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
(3)  004550  011637  001252                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
(3)  004554  013777  001246  174524  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
(3)  004562  013716  001252                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
(5)  004566  004737  007046                  JSR     PC,SERV.G         ;FIND OUT IF ^G WAS TYPED
(5)  004572  105037  001424                  CLRB    MNTFLG            ;CLEAR THE MAINTENANCE BIT SETTER AFTER EACH TEST
(5)  004576  005737  001372                  TST     MODE              ;HAS THE MODE BEEN CHANGED?
(5)  004602  001003                          BNE     4$                ;IF NOT INTERNAL , GO DO A TEST
(5)  004604  112737  000010  001424          MOVB    #MAINT,MNTFLG     ;IF INTERNAL MODE NOW, SET THE MAINTENANCE BIT
(5)  004612  000002          4$:     RTI                               ;GO DO THE TEST
(5)  004614  000406          BRW:    406
(3)  004616  000005          $MXCNT: 5                                 ;;MAX. NUMBER OF ITERATIONS
(1)
(1)                                  ;CHECK FOR FREEZE ON CURRENT DATA
(1)                                  ;-------------------------------
(1)
(1)  004620  032777  001000  174456  .SCOP1: BIT     #SW09,@SWR        ;IS SW09=1(SET)?
(1)  004626  001405                          BEQ     1$                ;BR IF NOT SET.
(1)  004630  005737  001364                  TST     LOCK              ;IS THERE A TIGHT LOOP SPECIFIED?
(1)  004634  001402                          BEQ     1$                ;IF NO, RETURN
(1)  004636  013716  001364                  MOV     LOCK,(SP)         ;IF YES, GOTO THE ADDRESS IN LOCK.
(1)  004642  000002          1$:     RTI                               ;GO BACK.
(1)
(1)  004644  032777  010000  174432  .TYPE:  BIT     #SW12,@SWR        ;INHIBIT ALL PRINTOUT??
(1)  004652  001403                          BEQ     $TYPE             ;IF NOT, GO TYPE
(1)  004654  062716  000002                  ADD     #2,(SP)           ;SKIP OVER MESSAGE POINTER
(1)  004660  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)  004662  105737  001323          $TYPE:  TSTB    $TPFLG            ;;IS THERE A TERMINAL?
(2)  004666  100002                          BPL     1$                ;;BR IF YES
(2)  004670  000000                          HALT                      ;;HALT HERE IF NO TERMINAL
(2)  004672  000430                          BR      3$                ;;LEAVE
(2)  004674  010046          1$:     MOV     R0,-(SP)          ;;SAVE R0
(2)  004676  017600  000002                  MOV     @2(SP),R0         ;;GET ADDRESS OF ASCIZ STRING
(2)  004702  122737  000001  001140          CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
(2)  004710  001011                          BNE     62$               ;;NO,GO CHECK FOR APT CONSOLE
(2)  004712  132737  000100  001141          BITB    #APTSPOOL,$ENVM   ;;SPOOL MESSAGE TO APT
```

```
(2)  004720  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(2)  004722  010037  004732                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(2)  004726  004737  005152                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(2)  004732  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
(2)  004734  132737  000040  001141  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(2)  004742  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
(2)  004744  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2)  004746  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(2)  004750  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(2)  004752  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
(2)  004754  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(2)  004760  000002                          RTI                     ;;RETURN
(2)  004762  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(2)  004766  001430                          BEQ     8$
(2)  004770  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(2)  004774  001006                          BNE     5$
(2)  004776  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
(2)  005000  104402                          TYPE                    ;;TYPE A CR AND LF
(2)  005002  001357                          $CRLF
(2)  005004  105037  005140                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(2)  005010  000755                          BR      2$              ;;GET NEXT CHARACTER
(2)  005012  004737  005074          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(2)  005016  123726  001322          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(2)  005022  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
(2)  005024  013746  001320                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(2)                                                                  ;;AND THE NULL CHAR.
(2)  005030  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(2)  005034  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2)  005036  004737  005074                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
(2)  005042  105337  005140                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(2)  005046  000770                          BR      7$              ;;LOOP
(2)
(2)                                  ;HORIZONTAL TAB PROCESSOR
(2)
(2)  005050  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(2)  005054  004737  005074          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(2)  005060  132737  000007  005140          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(2)  005066  001372                          BNE     9$              ;;TAB STOP
(2)  005070  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(2)  005072  000724                          BR      2$              ;;GET NEXT CHARACTER
(2)  005074  105777  174214          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(2)  005100  100375                          BPL     $TYPEC
(2)  005102  116677  000002  174206          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)
(2)  005110  122766  000015  000002  CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(2)  005116  001003                          BNE     1$              ;;BRANCH IF NO
(2)  005120  105037  005140                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(2)  005124  000406                          BR      $TYPEX          ;;EXIT
(2)  005126  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(2)  005134  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(2)  005136  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(2)  005140  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(2)  005142  000207                  $TYPEX: RTS     PC
(2)
(2)
(2)
```

```
(2)                                     .SBTTL  APT COMMUNICATIONS ROUTINE
(2)
(3)                                     ;;*****************************************************************
(2)  005144  112737  000001  005410  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(2)  005152  112737  000001  005406  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(2)  005160  000403                          BR      $ATYC
(2)  005162  112737  000001  005410  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(2)  005170                          $ATYC:
(4)  005170  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(4)  005172  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(2)  005174  105737  005406                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(2)  005200  001450                          BEQ     5$              ;;IF NOT:  BR
(2)  005202  122737  000001  001140          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(2)  005210  001031                          BNE     3$              ;;IF NOT:  BR
(2)  005212  132737  000100  001141          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(2)  005220  001425                          BEQ     3$              ;;IF NOT:  BR
(2)  005222  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(2)  005226  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(2)  005234  005737  001120          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(2)  005240  001375                          BNE     1$              ;;IF NOT:  WAIT
(2)  005242  010037  001134                  MOV     R0,$MSGAD
(2)                                          ;;PUT ADDR IN MAILBOX
(2)  005246  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(2)  005250  001376                          BNE     2$
(2)  005252  163700  001134                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(2)  005256  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(2)  005260  010037  001136                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(2)  005264  012737  000004  001120          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(2)  005272  000413                          BR      5$
(2)  005274  017637  000004  005320  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(2)  005302  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(4)  005310  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(2)  005314  004737  004662                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
(2)  005320  0C0000                  4$:     .WORD   0
(2)  005322                          5$:
(2)  005322  105737  005410          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(2)  005326  001416                          BEQ     12$             ;;IF NOT:  BR
(2)  005330  005737  001140                  TST     $ENV            ;;RUNNING UNDER APT?
(2)  005334  001413                          BEQ     12$             ;;IF NOT:  BR
(2)  005336  005737  001120          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(2)  005342  001375                          BNE     11$             ;;IF NOT:  WAIT
(2)  005344  017637  000004  001122          MOV     @4(SP),$FATAL   ;;GET ERROR #
(2)  005352  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(2)  005360  005237  001120                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(2)  005364  105037  005410          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
(2)  005370  105037  005407                  CLRB    $LFLG           ;;CLEAR LOG FLAG
(2)  005374  105037  005406                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(4)  005400  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(4)  005402  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(2)  005404  000207                          RTS     PC              ;;RETURN
(2)  005406     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
(2)  005407     000                  $LFLG:  .BYTE   0
(2)                                          ;;LOG FLAG
(2)  005410     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
(2)          005412                          .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)  005412  010346                  .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
(1)  005414  010446                          MOV     R4,-(SP)        ;SAVE R4 ON STACK
(1)  005416  017637  000004  005434          MOV     @4(SP),.MSG     ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1)  005424  062766  000002  000004          ADD     #2,4(SP)        ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1)  005432  104402                  .INST1: TYPE                    ;PRINT THE MESSAGE
(1)  005434  000000                  .MSG:   0                       ;MESSAGE IS POINTED TO FROM HERE
(1)  005436  012704  010344                  MOV     #INBUF,R4       ;POINT R4 TO THE INPUT BUFFER
(1)  005442  012703  000007                  MOV     #7,R3           ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1)  005446  105777  173636          1$:     TSTB    @$TKS           ;HAS A CHARACTER BEEN RECEIVED?
(1)  005452  100375                          BPL     1$              ;IF NO, KEEP WAITING FOR IT
(1)  005454  117714  173632                  MOVB    @$TKB,(R4)      ;IF YES, SAVE IT IN THE INPUT BUFFER
(1)  005460  142714  000200                  BICB    #200,(R4)       ;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1)  005464  122427  000015                  CMPB    (R4)+,#15       ;IS THIS CHARACTER A LINE FEED?
(1)  005470  001417                          BEQ     INSTR2          ;IF SO, TERMINATE THE INPUT SEQUENCE
(1)  005472  105777  173616          2$:     TSTB    @$TPS           ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1)  005476  100375                          BPL     2$              ;IF WE CAN'T, WAIT UNTIL WE CAN
(1)  005500  017777  173606  173610          MOV     @$TKB,@$TPB     ;ECHO THE CHARACTER BACK
(1)  005506  005303                          DEC     R3              ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
(1)  005510  001356                          BNE     1$              ;IF WE DON'T HAVE 7, GO GET SOME MORE
(1)  005512  012604                          MOV     (SP)+,R4        ;IF WE HAVE 7, RESTORE R4
(1)  005514  012603                          MOV     (SP)+,R3        ;RESTORE R3
(1)  005516  010346                  .INSTE: MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
(1)  005520  010446                          MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
(1)  005522  104402  001356                  TYPE    ,$QUES          ;PRINT A QUESTION MARK... WHAT'S GOING ON?
(1)  005526  000741                          BR      .INST1          ;GO PRINT THE MESSAGE AGAIN
(1)  005530  012604                  INSTR2: MOV     (SP)+,R4        ;RESTORE R4
(1)  005532  012603                          MOV     (SP)+,R3        ;RESTORE R3
(1)  005534  000002                          RTI                     ;RETURN TO THE MAIN PROCEDURE
(1)
(1)                                          ;CONVERT ASCII STRING TO OCTAL
(1)                                          ;----------------------------
(1)
(1)  005536  010546                  .PARAM: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
(1)  005540  010446                          MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
(1)  005542  016605  000004                  MOV     4(SP),R5        ;GET THE SETUP INFORMATION POINTER
(1)  005546  012537  005726                  MOV     (R5)+,LOLIM     ;SET THE LOW LIMIT FOR THE INPUT
(1)  005552  012537  005730                  MOV     (R5)+,HILIM     ;SET THE HIGH LIMIT FOR THE INPUT
(1)  005556  012537  005732                  MOV     (R5)+,DEVADR    ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
(1)  005562  112537  005734                  MOVB    (R5)+,LOBITS    ;GET THE MASK OF THE INCORRECT BITS
(1)  005566  112537  005735                  MOVB    (R5)+,ADRCNT    ;GET THE COUNT OF ITEMS TO BE STORED
(1)  005572  010566  000004                  MOV     R5,4(SP)        ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
(1)  005576  005005                  PARAM1: CLR     R5              ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
(1)  005600  012704  010344                  MOV     #INBUF,R4       ;POINT TO THE INPUT BUFFER
(1)  005604  122714  000015                  CMPB    #15,(R4)        ;IS THIS CHARACTER A CARRIAGE RETURN?
(1)  005610  001420                          BEQ     PARERR          ;IF SO, PRINT THE MESSAGE AGAIN
(1)  005612  121427  000060          1$:     CMPB    (R4),#60        ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
(1)  005616  002415                          BLT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)  005620  121427  000067                  CMPB    (R4),#67        ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
(1)  005624  003012                          BGT     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
```

```
(1)  005626  142714  000060          BICB    #60,(R4)        ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
(1)  005632  152405                  BISB    (R4)+,R5        ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
(1)  005634  122714  000015          CMPB    #15,(R4)        ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
(1)  005640  001406                  BEQ     LIMITS          ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
(1)  005642  006305                  ASL     R5              ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
(1)  005644  006305                  ASL     R5              ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
(1)  005646  006305                  ASL     R5              ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
(1)                                                          ;NEXT THREE BITS
(1)  005650  000760                  BR      1$              ;GO GET THE NEXT CHARACTER
(1)  005652  104404          PARERR: INSTER                  ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
(1)  005654  000750                  BR      PARAM1          ;TRY GETTING THE PARAMETERS AGAIN
(1)
(1)                                  ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(1)                                  ;--------------------------------------
(1)
(1)  005656  020537  005730  LIMITS: CMP     R5,HILIM        ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
(1)  005662  101373                  BHI     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005664  020537  005726          CMP     R5,LOLIM        ;IS THE RESULT LOWER THAN ALLOWED?
(1)  005670  103770                  BLO     PARERR          ;IF YES, GO PRINT THE MESSAGE AGAIN
(1)  005672  133705  005734          BITB    LOBITS,R5       ;ARE ANY INCORRECT BITS SET IN THE RESULT?
(1)  005676  001365                  BNE     PARERR          ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)
(1)                                  ;STORE NUMBER AT SPECIFIED ADDRESS
(1)
(1)  005700  013704  005732          MOV     DEVADR,R4       ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1)  005704  010524          1$:     MOV     R5,(R4)+        ;STORE THE RESULT
(1)  005706  062705  000002          ADD     #2,R5           ;CALCULATE THE NEXT DATUM
(1)  005712  105337  005735          DECB    ADRCNT          ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1)  005716  001372                  BNE     1$              ;IF NOT, GO STORE THE NEXT DATUM
(1)  005720  012604                  MOV     (SP)+,R4        ;RESTORE R4
(1)  005722  012605                  MOV     (SP)+,R5        ;RESTORE R5
(1)  005724  000002                  RTI                     ;RETURN TO THE MAIN PROGRAM
(1)
(1)  005726  000000          LOLIM:  0                       ;LOWEST ACCEPTABLE VALUE
(1)  005730  000000          HILIM:  0                       ;HIGHEST ACCEPTABLE
(1)  005732  000000          DEVADR: 0                       ;LOCATION WHERE RESULT WILL BE STORED
(1)  005734     000          LOBITS: .BYTE   0               ;INCORRECT BITS MASK
(1)  005735     000          ADRCNT: .BYTE   0               ;COUNT OF ITEMS TO BE STORED
(1)
(1)                                  ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1)                                  ;-------------------------------------
(1)
(1)  005736  016637  000004  001404  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
(1)
(1)                                  ;SAVE R0-R5
(1)
(1)  005744  010537  001340          SV05:   MOV     R5,$REG5        ;SAVE R5
(1)  005750  010437  001336                  MOV     R4,$REG4        ;SAVE R4
(1)  005754  010337  001334                  MOV     R3,$REG3        ;SAVE R3
(1)  005760  010237  001332                  MOV     R2,$REG2        ;SAVE R2
(1)  005764  010137  001330                  MOV     R1,$REG1        ;SAVE R1
(1)  005770  010037  001326                  MOV     R0,$REG0        ;SAVE R0
(1)  005774  000002                          RTI                     ;LEAVE.
(1)
(1)                                  ;RESTORE R0-R5
(1)
```

```
(1)  005776  013700  001326          .RES05: MOV     $REG0,R0        ;RESTORE R0
(1)  006002  013701  001330                  MOV     $REG1,R1        ;RESTORE R1
(1)  006006  013702  001332                  MOV     $REG2,R2        ;RESTORE R2
(1)  006012  013703  001334                  MOV     $REG3,R3        ;RESTORE R3
(1)  006016  013704  001336                  MOV     $REG4,R4        ;RESTORE R4
(1)  006022  013705  001340                  MOV     $REG5,R5        ;RESTORE R5
(1)  006026  000002                          RTI                     ;LEAVE
(1)
(1)                                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(1)                                          ;-----------------------------------------------------
(1)
(1)  006030  104402  001357          .CONVR: TYPE    ,$CRLF          ;PRINT A CARRIAGE RETURN
(1)  006034  010046                  .CNVRT: MOV     R0,-(SP)        ;SAVE R0
(1)  006036  010146                          MOV     R1,-(SP)        ;SAVE R1
(1)  006040  010346                          MOV     R3,-(SP)        ;SAVE R3
(1)  006042  010446                          MOV     R4,-(SP)        ;SAVE R4
(1)  006044  010546                          MOV     R5,-(SP)        ;SAVE R5
(1)  006046  017601  000012                  MOV     @12(SP),R1      ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
(1)  006052  062766  000002  000012          ADD     #2,12(SP)       ;POINT TO WHERE MAIN PROGRAM WILL RESUME
(1)  006060  012137  006204                  MOV     (R1)+,WRDCNT    ;GET NUMBER OF WORDS TO BE PRINTED
(1)  006064  112105                  1$:     MOVB    (R1)+,R5        ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
(1)  006066  112100                          MOVB    (R1)+,R0        ;GET THE NUMBER OF SPACES TO PRINT
(1)  006070  013104                          MOV     @(R1)+,R4       ;COPY THE WORD TO BE CONVERTED
(1)  006072  110537  006206                  MOVB    R5,CHRCNT       ;COPY THE CHARACTER COUNT
(1)  006076  010403                  3$:     MOV     R4,R3           ;COPY THE ARGUMENT WORD AGAIN
(1)  006100  042703  177770                  BIC     #^C<7>,R3       ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
(1)  006104  062703  000060                  ADD     #060,R3         ;MAKE AN ASCII CHARACTER OUT OF THEM
(1)  006110  110346                          MOVB    R3,-(SP)        ;SAVE THAT CHARACTER
(1)  006112  006004                          ROR     R4              ;MOVE THE NEXT THREE BITS INTO PLACE
(1)  006114  006204                          ASR     R4              ;MOVE THEM AGAIN
(1)  006116  006204                          ASR     R4              ;AND FINALLY A THIRD TIME
(1)  006120  005305                          DEC     R5              ;REDUCE CHARACTER COUNT.ARE ALL CHARACTERS
(1)                                                                  ;BUILT?
(1)  006122  001365                          BNE     3$              ;IF NO, GO BUILD THE NEXT ONE.
(1)  006124  012703  010406                  MOV     #MDATA,R3       ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1)  006130  112623                  4$:     MOVB    (SP)+,(R3)+     ;STORE THE CHARACTER, STARTING WITH THE MOST
(1)  006132  105337  006206                  DECB    CHRCNT          ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1)  006136  001374                          BNE     4$              ;IF NO, GO TRANSFER ANOTHER
(1)  006140  105700                          TSTB    R0              ;ARE ANY SPACES TO BE PRINTED?
(1)  006142  001404                          BEQ     6$              ;IF NO, DON'T SET UP ANY
(1)  006144  112723  000040          5$:     MOVB    #040,(R3)+      ;ADD A SPACE TO THE OUTPUT BUFFER
(1)  006150  105300                          DECB    R0              ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1)  006152  001374                          BNE     5$              ;IF YES, GO ADD ANOTHER SPACE
(1)  006154  105013                  6$:     CLRB    (R3)            ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
(1)  006156  104402  010406                  TYPE    ,MDATA          ;PRINT THE STRING WE JUST BUILT
(1)  006162  005337  006204                  DEC     WRDCNT          ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1)  006166  001336                          BNE     1$              ;IF YES, GO CONVERT THEM
(1)  006170  012605                          MOV     (SP)+,R5        ;RESTORE R5
(1)  006172  012604                          MOV     (SP)+,R4        ;RESTORE R4
(1)  006174  012603                          MOV     (SP)+,R3        ;RESTORE R3
(1)  006176  012601                          MOV     (SP)+,R1        ;RESTORE R1
(1)  006200  012600                          MOV     (SP)+,R0        ;RESTORE R0
(1)  006202  000002                          RTI                     ;RETURN TO THE MAIN PROGRAM
(1)  006204  000000          WRDCNT: 0
(1)  006206     000          CHRCNT: .BYTE                           ;NUMBER OF CHARACTERS TO PRINT
(1)  006207     000          SPACNT: .BYTE   0                       ;NUMBER OF SPACES TO PRINT
```

```
(1)
(1)  006210  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)  006212  010046                  .TRPSR: MOV     R0,-(SP)          ;SAVE R0. USE R0 TO FIND TRAP ROUTINE
(1)  006214  016600  000002                  MOV     2(SP),R0          ;GET TRAP ADDRESS
(1)  006220  005740                          TST     -(R0)             ;GET TRAP
(1)  006222  111000                          MOVB    (R0),R0           ;GET RIGHT BYTE OF TRAP(TRAP OFFSET)
(1)  006224  006300                          ASL     R0                ;POSITION OFFSET FOR TABLE INDEXING
(1)  006226  016000  001742                  MOV     .TRPTAB(R0),R0    ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1)  006232  000200                          RTS     R0                ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                                           ;DEVICE CLEAR ROUTINE
(1)                                           ;ISSUE A DEVICE CLEAR
(1)                                           ;--------------------
(1)  006234                          .DEVICE.CLR:
(1)  006234  052777  000020  173546          BIS     #DCLR,@DZVCSR     ;SET DCLR
(1)  006242  032777  000020  173540  1$:     BIT     #DCLR,@DZVCSR     ;DID IT CLEAR?
(1)  006250  001374                          BNE     1$                ;BR IF NO
(1)  006252  000002                          RTI                       ;EXIT ROUTINE
(1)
(1)                                           ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                                           ;-----------------------------------------------------------
(1)  006254  104413                  .DCLASM:DEVICE.CLR                ;ISSUE A DEVICE CLEAR
(1)  006256  153777  001424  173524          BISB    MNTFLG,@DZVCSR    ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1)  006264  000002                          RTI                       ;RETURN TO CALLING ROUTINE
(1)
(1)  006266                          .DELAY:
(1)  006266  010046                          MOV     R0,-(SP)          ;SAVE R0
(1)  006270  013700  006304                  MOV     DLYCNT,R0         ;SET COUNT
(1)  006274  005300                  1$:     DEC     R0                ;DELAY
(1)  006276  001376                          BNE     1$                ;
(1)  006300  012600                          MOV     (SP)+,R0          ;RESTORE R0
(1)  006302  000002                          RTI                       ;LEAVE ROUTINE
(1)  006304  000001                  DLYCNT: .WORD   1                 ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                                           ;ADVANCE TO NEXT TEST HANDLER
(1)                                           ;---------------------------
(1)
(1)  006306  013716  001362          .ADVANCE:MOV    NEXT,(SP)         ;CRUNCH STACK WITH ADDRESS OF SCOPE CALL
(1)  006312  005037  001364                  CLR     LOCK              ;RESET TIGHT LOOP ADDRESS
(1)  006316  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)  006320  106302                  .SHIFT: ASLB    R2                ;POINT TO THE NEXT LINE
(1)  006322  032702  000020                  BIT     #BIT4,R2          ;HAVE WE PASSED ALL LINE POINTERS?
(1)  006326  001402                          BEQ     1$                ;IF NOT, RETURN TO THE TEST
(1)  006330  022626                          POP2SP                    ;REMOVE THE TRAP CALL FROM THE STACK
(1)  006332  104400                          ADVANCE                   ;GO TO THE NEXT TEST
```

```
    (1)  006334  000002                1$:     RTI                     ;RETURN TO THE PRESENT TEST
    (1)
```

```
(1)                                             ;LINE PARAMETER REGISTER SETUP ROUTINE
(1)
(1)  006336  010146                 .LPRSET:MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
(1)  006340  010246                         MOV     R2,-(SP)        ;SAVE CONTENTS OF R2
(1)  006342  013701  001370                 MOV     PAR,R1          ;MOVE DEFAULT PARAM. INTO R1
(1)  006346  012702  000001                 MOV     #1,R2   ;INIT. FOR LINE 1
(1)  006352  010177  173442         1$:     MOV     R1,@DZVLPR      ;LOAD PARAM. REGISTER
(1)  006356  005201                         INC     R1              ;SET R1 FOR NEXT LINE
(1)  006360  106302                         ASLB    R2              ;SET R2 FOR NEXT LINE
(1)  006362  032702  000020                 BIT     #BIT4,R2        ;ALL LINES DONE?
(1)  006366  001771                         BEQ     1$              ;IF NO LOAD NEXT LINE
(1)  006370  012602                         MOV     (SP)+,R2        ;RELOAD R2
(1)  006372  012601                         MOV     (SP)+,R1        ;RELOAD R1
(1)  006374  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ROUTINE TO ZERO DATA BUFFER
(1)
(1)  006376  010046                 .BUFSET:MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
(1)  006400  012700  001426                 MOV     #TDO,R0         ;SET R0 TO TOP OF BUFFER
(1)  006404  005020                 1$:     CLR     (R0)+           ;CLEAR BUFFER LOCATION
(1)  006406  022700  001446                 CMP     #STOP,R0        ;IS BUFFER ALL CLEARED
(1)  006412  001374                         BNE     1$              ;IF NOT CLEAR NEXT LOCATION
(1)  006414  012600                         MOV     (SP)+,R0        ;RELOAD R0
(1)  006416  000002                         RTI                     ;RETURN
(1)
(1)                                             ;ERROR HANDLER
(1)                                             ;-------------
(1)
(1)  006420  004737  007046         $ERROR: JSR     PC,SERV.G       ;FIND OUT IF <^G> WAS HIT
(1)  006424  032777  010000  172652         BIT     #SW12,@SWR      ;BELL ON ERROR?
(1)  006432  001406                         BEQ     XBX             ;BR IF NO BELL
(1)  006434  105777  172654                 TSTB    @$TPS           ;TTY READY.
(1)  006440  100003                         BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
(1)  006442  112777  000207  172646         MOVB    #207,@$TPB      ;PUSH A BELL AT THE TTY.
(1)  006450  032777  020000  172626 XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
(1)  006456  001113                         BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
(1)  006460  021637  001262                 CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
(1)  006464  001404                         BEQ     1$              ;BR IF YES
(1)  006466  011637  001262                 MOV     (SP),$ERRPC     ;RECORD BEING HERE
(1)  006472  105037  001247                 CLRB    $ERFLG          ;PREPARE HEADER
(1)  006476  104407                 1$:     SAV05                   ;SAVE ALL PROC REGISTERS
(1)  006500  011605                         MOV     (SP),R5         ;GET THE PC OF ERROR
(1)  006502  162705  000002                 SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
(1)  006506  011504                         MOV     (R5),R4         ;GET ERROR INSTRUCTION
(1)  006510  110437  001260                 MOVB    R4,$ITEMB       ;COPY TEST NUMBER FOR APT HANDLING
(1)  006514  006304                         ASL     R4              ;MULT BY TWO
(1)  006516  061504                         ADD     (R5),R4         ;DOUBLE IT
(1)  006520  006304                         ASL     R4              ;MULT AGAIN
(1)  006522  042704  177001                 BIC     #177001,R4      ;CLEAR JUNK
(1)  006526  062704  015136                 ADD     #.ERRTAB,R4     ;GET POINTER
(1)  [illegible]                            [illegible]             [illegible]
(1)  006536  012437  006670                 MOV     (R4)+,DATAHD    ;GET DATA HEADER
(1)  006542  011437  006702                 MOV     (R4),DATABP     ;GET DATA TABLE
(1)  006546  105737  001247                 TSTB    $ERFLG          ;TYPE HEADER
(1)  006552  001403                         BEQ     TYPMSG          ;BR IF YES
(1)  006554  005737  006702                 TST     DATABP          ;DOES DATA TABLE EXIST?
```

```
(1)  006560  001044                        BNE     TYPDAT          ;BR IF YES.
(1)  006562  104402  001357          TYPMSG: TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN
(1)  006566  104402  001357                  TYPE    ,$CRLF          ;AND TYPE ANOTHER
(1)  006572  005737  001364                  TST     LOCK
(1)  006576  001402                          BEQ     1$
(1)  006600  104402  007761                  TYPE    ,MASTEK
(1)  006604  104402  007747          1$:     TYPE    ,MTSTN
(1)  006610  104412  007040                  CNVRT   ,XTSTN          ;SHOW IT
(1)  006614  104402  010041                  TYPE    ,MERRPC         ;TYPE PC.
(1)  006620  104412  007032                  CNVRT   ,ERTABO         ;SHOW IT
(1)  006624  104402  007711                  TYPE    ,MCSRX
(1)  006630  104412  004324                  CNVRT   ,XCSR
(1)  006634  104402  001357                  TYPE    ,$CRLF          ;GIVE A CR/LF
(1)  006640  112737  177777  001247          MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
(1)  006646  005737  006656                  TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
(1)  006652  001402                          BEQ     WTBS.FM         ;BR IF NO.
(1)  006654  104402                          TYPE                    ;TYPE
(1)  006656  000000                  ERRMSG: 0                       ;    ERROR MESSAGE
(1)  006660                          WTBS.FM:                        ;
(1)  006660  005737  006670                  TST     DATAHD          ;DATA HEADER?
(1)  006664  001402                          BEQ     TYPDAT          ;BR IF NO
(1)  006666  104402                          TYPE                    ;TYPE
(1)  006670  000000                  DATAHD: 0                       ;    DATA HEADER
(1)  006672  005737  006702          TYPDAT: TST     DATABP          ;DATA TABLE?
(1)  006676  001402                          BEQ     RESREG          ;BR IF NO.
(1)  006700  104411                          CONVRT                  ;SHOW
(1)  006702  000000                  DATABP: 0                       ;    DATA TABLE
(1)  006704  104410                  RESREG: RES05                   ;RESTORE PROC REGISTERS
(1)  006706  122737  000001  001140  HALTS:  CMPB    #APTENV,$ENV    ;IS APT RUNNING?
(1)  006714  001007                          BNE     15$             ;SKIP APT CALL IF NOT
(1)  006716  113737  001260  006730          MOVB    $ITEMB,5$       ;COPY ERROR NUMBER
(1)  006724  004737  005162                  JSR     PC,$ATY4        ;CALL APT SERVICE
(1)  006730  000000                  5$:     .WORD   0               ;ERROR NUMBER STUCK HERE
(1)  006732  000777                  10$:    BR      10$             ;LOCK UP HERE
(1)  006734  022737  004310  000042  15$:    CMP     #$ENDAD,@#42    ;CHECK TO SEE IF IN ACT-11 MODE
(1)  006742  001403                          BEQ     20$             ;IF SO, HANDLE ACCORDINGLY
(1)  006744  005777  172334                  TST     @SWR            ;HALT ON ERROR?
(1)  006750  100004                          BPL     EXITER          ;BR IF NO HALT ON ERROR
(1)  006752  016677  000002  172326  20$:    MOV     2(SP),@DISPLAY  ;SHOW ERROR PC IN DATA DISPLAY
(1)  006760  000000                          HALT                    ;HALT
(1)  006762  005237  001256          EXITER: INC     $ERTTL          ;UPDATE ERROR COUNT
(1)  006766  004737  007046                  JSR     PC,SERV.G       ;FIND OUT IF ^G WAS TYPED
(1)  006772  032777  000400  172304          BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
(1)  007000  001007                          BNE     1$              ;BR IF YES
(1)  007002  032777  002000  172274          BIT     #SW10,@SWR      ;GOTO NEXT TEST?
(1)  007010  001407                          BEQ     2$              ;BR IF NO
(1)  007012  013737  001362  001252          MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
(1)  007020  012706  001120          1$:     MOV     #STACK,SP       ;RESET SP
(1)  007024  000177  172222                  JMP     @$LPADR         ;GOTO SPECIFIED TEST
(1)  007030  000002                  2$:     RTI                     ;RETURN
(1)  007032  000001                  ERTABO: 1
(1)  007034     006     002                  .BYTE   6,2
(1)  007036  001404                          $AVPC
(1)  007040  000001                  XTSTN:  1
(1)  007042     002     002                  .BYTE   2,2
(1)  007044  001246                          $TSTNM
```

```
(1)   007046  017746  172240          SERV.G: MOV     @$TKB,-(SP)     ;OTHERWISE, GET THE LAST CHARACTER TYPED
(1)   007052  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY(EIGHTH) BIT
(1)   007056  122726  000007                  CMPB    #7,(SP)+        ;IS IT ^G?
(1)   007062  001076                          BNE     6$              ;IF NOT, IGNORE INPUT
(1)   007064  032777  004000  172216          BIT     #4000,@$TKS     ;RX BUSY?
(1)   007072  001365                          BNE     SERV.G          ;BR IF YES
(1)           007074                  GETSWR= .                                               ;;GPA
(1)   007074  017737  172204  007302          MOV     @SWR,90$        ;SAVE (SWR).
(1)   007102  104402  007262          1$:     TYPE    ,89$            ;TYPE HEADER FOR OLD SWITCH REGISTER
(1)   007106  104412  007274                  CNVRT   ,88$            ;TYPE THE NUMBER ITSELF
(1)   007112  104402  007304                  TYPE    ,91$            ;AFTER HAVING CONVERTED IT TO ASCII
(1)   007116  105037  007310                  CLRB    92$             ;CLEAR SWR CHANGE FLAG
(1)   007122  005077  172156                  CLR     @SWR            ;CLEAR THE SOFTWARE SWITCH REGISTER
(1)   007126  105777  172156          3$:     TSTB    @$TKS           ;WAIT FOR DONE.
(1)   007132  100375                          BPL     3$              ;CONTINUE WAITING FOR IT
(1)   007134  017746  172152                  MOV     @$TKB,-(SP)     ;PUT THE CHARACTER ON THE STACK
(1)   007140  042716  000200                  BIC     #BIT7,(SP)      ;STRIP PARITY BIT
(1)   007144  122726  000015                  CMPB    #15,(SP)+       ;IS IT THE CARRIAGE RETURN CHAR?
(1)   007150  001433                          BEQ     4$              ;IF SO, GO PRINT CRLF
(1)   007152  105777  172136          2$:     TSTB    @$TPS           ;IS THE OUTPUT BUFFER AVAILABLE
(1)   007156  100375                          BPL     2$              ;IF NOT, WAIT FOR IT TO BE READY
(1)   007160  105237  007310                  INCB    92$             ;INDICATE THAT THE SWR WAS CHANGED
(1)   007164  014677  172126                  MOV     -(SP),@$TPB     ;PLACE THE CHARACTER THERE(ECHO BACK)
(1)   007170  000241                          CLC                     ;GET READY TO ROTATE
(1)   007172  006177  172106                  ROL     @SWR            ;MOVE THE EXISTING BITS OVER
(1)   007176  006177  172102                  ROL     @SWR            ;TO MAKE ROOM FOR THE INCOMING
(1)   007202  006177  172076                  ROL     @SWR            ;THREE BITS FROM THIS CHARACTER
(1)   007206  103735                          BCS     1$              ;ERROR
(1)   007210  022627  000060                  CMP     (SP)+,#60       ;IS IT LOWER THAN 0?
(1)   007214  002732                          BLT     1$              ;IF SO, GO ASK AGAIN
(1)   007216  026627  177776  000067          CMP     -2(SP),#67      ;IS IT HIGHER THAN 7?
(1)   007224  003326                          BGT     1$              ;IF SO, GO ASK AGAIN
(1)   007226  042746  177770                  BIC     #^C<7>,-(SP)    ;ISOLATE INFORMATION BITS
(1)   007232  052677  172046                  BIS     (SP)+,@SWR      ;ADD THEM TO THE SWITCH REGISTER
(1)   007236  000733                          BR      3$              ;GO CHECK FOR THE NEXT CHARACTER
(1)   007240  105737  007310          4$:     TSTB    92$             ;HAS THE SWR BEEN CHANGED?
(1)   007244  001003                          BNE     5$              ;IF YES GO TYPE CRLF
(1)   007246  013777  007302  172030          MOV     90$,@SWR        ;IF NOT RESTORE SWR
(1)   007254  104402  001357          5$:     TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
(1)   007260  000207                  6$:     RTS     PC              ;RETURN TO CALLING PROCEDURE
(1)
(1)   007262  020200  051450  051127  89$:    .ASCIZ  <200>? (SWR)=/?
(1)   007270  036451  000057
(1)                                   .EVEN
(1)   007274  000001                  88$:    1
(1)   007276     006     000                  .BYTE   6,0
(1)   007300  007302                          90$
(1)   007302  000000                  90$:    .WORD   0
(1)   007304  036457  000057          91$:    .ASCIZ  ?/=/?
(1)   007310     000                  92$:    .BYTE   0
(1)           007312                  .EVEN
(2)                                   .SBTTL  POWER DOWN AND UP ROUTINES
(2)
(3)                                   ;;**********************************************************************
(2)                                   ;POWER DOWN ROUTINE
(2)   007312  012737  007456  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
```

```
(2)  007320  012737  000300  000026          MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
(4)  007326  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(4)  007330  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(4)  007332  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(4)  007334  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(4)  007336  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(4)  007340  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(4)  007342  017746  171736                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(2)  007346  010637  007462                  MOV     SP,$SAVR6       ;;SAVE SP
(2)  007352  012737  007364  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(2)  007360  000000                          HALT
(2)  007362  000776                          BR      .-2             ;;HANG UP
(2)
(3)                                ;;**********************************************************************
(2)                                ;POWER UP ROUTINE
(2)  007364  012737  007456  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(2)  007372  013706  007462                  MOV     $SAVR6,SP       ;;GET SP
(2)  007376  005037  007462                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(2)  007402  005237  007462          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(2)  007406  001375                          BNE     1$              ;;OF  WORD
(4)  007410  012677  171670                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(4)  007414  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(4)  007416  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
(4)  007420  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(4)  007422  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(4)  007424  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(4)  007426  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(2)  007430  012737  007312  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(2)  007436  012737  000300  000026          MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
(2)  007444  104402                          TYPE                    ;;REPORT THE POWER FAILURE
(2)  007446  007464                $PWRMG:   .WORD   MPFAIL          ;;POWER FAIL MESSAGE POINTER
(2)  007450  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTART
(2)  007452  011010                $PWRAD:   .WORD   RESTART         ;;RESTART ADDRESS
(2)  007454  000002                          RTI
(2)  007456  000000                $ILLUP:   HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(2)  007460  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(2)  007462  000000                $SAVR6:   0                       ;;PUT THE SP HERE
(2)  007464  050200  051127  043040 MPFAIL:  .ASCIZ  <200>/PWR FAILED. RESTART AT LAST TEST /
(2)  007527     200  047105  020104 MEPASS:  .ASCIZ  <200>/END PASS CNDZB-A  /
(2)  007553     200  052522  047116 MR:      .ASCIZ  <200>/RUNNING   /
(2)  007567     200  051120  043517 MERR2:   .ASCIZ  <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2)  007636  044600  051516  043125 MERR3:   .ASCIZ  <200>/INSUFFICIENT DATA!/
(2)  007662  046200  041517  020113 MLOCK:   .ASCIZ  <200>/LOCK ON SELECTED TEST/
(2)  007711     103  051123  020072 MCSRX:   .ASCIZ   /CSR: /
(2)  007717     126  041505  020072 MVECX:   .ASCIZ  /VEC: /
(2)  007725     120  051501  042523 MPASSX:  .ASCIZ  /PASSES: /
(2)  007736  051105  047522  051522 MERRX:   .ASCIZ  /ERRORS: /
(2)  007747     124  051505  020124 MTSTN:   .ASCIZ  /TEST NO: /
(2)  007761     052  000040        MASTEK:   .ASCIZ  /* /
(2)  007764  052200  050131  020105 MNEW:    .ASCIZ  <200>/TYPE A BIT MAP OF DZV11'S DESIRED ACTIVE:  /
(2)  010041     120  035103  000040 MERRPC:  .ASCIZ  /PC: /
(2)  010046  046600  050101  047440 XHEAD:   .ASCIZ  <200>/MAP OF DZV11 STATUS/<200>
(2)  010074  044600  046114  043505 MBADLN:  .ASCIZ  <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)                                .EVEN
(2)  010136  000002                XSTATQ:   2
(2)  010140     006     003                   .BYTE   6,3
```

```
(2)  010142  001344                          $TMP1
(2)  010144     006     002                  .BYTE   6,2
(2)  010146  001346                          $TMP2
(1)                                .EVEN
(2)                                ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
(2)                                ;-------------------------------------------------------------
(2)                                ;E=EXTERNAL LOOP BACK
(2)                                ;I=INTERNAL LOOP BACK
(2)                                ;S=STAGGERED LOOP BACK
(2)  010150  017605  000000        .SETFLG:MOV     @(SP),R5        ;PICK UP ADDRESS OF TAG
(2)  010154  042737  000040  010344         BIC     #40,INBUF       ;STRIP LOWER CASE
(2)  010162  122737  000105  010344         CMPB    #'E,INBUF       ;IS IT EXTERNAL LOOP BACK ?
(2)  010170  001005                          BNE     4$              ;NO
(2)  010172  013715  010262                  MOV     1$,(R5)         ;YES STORE INFO
(2)  010176  105037  001424                  CLRB    MNTFLG          ;SET MAINT BIT =0
(2)  010202  000422                          BR      7$              ;GET OUT
(2)  010204  122737  000111  010344  4$:     CMPB    #'I,INBUF       ;IS IT INTERNAL LOOP BACK ?
(2)  010212  001006                          BNE     5$              ;NO
(2)  010214  013715  010264                  MOV     2$,(R5)         ;YES STORE INFO
(2)  010220  112737  000010  001424         MOVB    #MAINT,MNTFLG   ;SET UP THE MAINTENANCE FLAG LOADER
(2)  010226  000410                          BR      7$              ;GET OUT
(2)  010230  122737  000123  010344  5$:     CMPB    #'S,INBUF       ;IS IT STAGGERED LOOP BACK ?
(2)  010236  001007                          BNE     6$              ;WHAT ?
(2)  010240  013715  010266                  MOV     3$,(R5)         ;YES STORE INFO
(2)  010244  105037  001424                  CLRB    MNTFLG          ;ZERO BITS
(2)  010250  062716  000002        7$:       ADD     #2,(SP)         ;POP AROUND
(2)  010254  000002                          RTI
(2)  010256  104404                6$:       INSTER                  ;RETRY
(2)  010260  000733                          BR      .SETFLG         ;DITTO
(2)  010262  000200                1$:       .WORD   200             ;EXTERNAL =  E
(2)  010264  000000                2$:       .WORD   0               ;INTERNAL = I
(2)  010266  100000                3$:       .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)  010270  017605  000000          .PAWCH:MOV     @(SP),R5
(2)  010274  142737  000040  010344          BICB    #40,INBUF       ;SET FOR LOWER CASE INPUT
(2)  010302  122737  000105  010344          CMPB    #'E,INBUF       ;IS IT "E" ?
(2)  010310  001002                          BNE     1$
(2)  010312  105015                          CLRB    (R5)            ;000
(2)  010314  000406                          BR      2$
(2)  010316  122737  000103  010344  1$:     CMPB    #'C,INBUF       ;IS IT "C" ?
(2)  010324  001005                          BNE     3$
(2)  010326  112715  177777                  MOVB    #-1,(R5)        ;3177
(2)  010332  062716  000002          2$:     ADD     #2,(SP)
(2)  010336  000002                          RTI
(2)  010340  104404                  3$:     INSTER                  ;RETRY
(2)  010342  000752                          BR      .PAWCH
(2)
(2)                                             ;BUFFERS FOR INPUT-OUTPUT
(2)
(2)  010344  000000                  INBUF:  0
(2)          010406                  .=.+40
(2)                                  ; TEMP: 0               ; TEMP AREA UNUSED.                     ;;GPA
(2)                                  ; .=.+40                ; DELETED TO CONSERVE SPACE.            ;;GPA
(2)  010406  000000                  MDATA:  0
(2)          010450                  .=.+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)  010450  005737  001406        CYCLE: TST     DZVACTV          ;ARE ANY DZV11'S TO BE TESTED?
(2)  010454  001004                       BNE     1$               ;BR IF OK.
(2)  010456  104402  007567               TYPE    ,MERR2           ;NO DZV11'S SELECTED!!
(2)  010462  000000                       HALT                     ;STOP THE SHOW.
(2)  010464  000776                       BR      .-2              ;DISQUALIFY CONT. SW.
(2)  010466  013737  004616  001354  1$:  MOV     $MXCNT,$TIMES    ;RESTORE THE NUMBER OF ITERATIONS TO MAKE
(2)  010474  033737  001412  001406       BIT     RUN,DZVACTV      ;IS THIS ONE "ACTIVE"
(2)  010502  001017                       BNE     2$               ;BR IF GOOD ONE FOUND.
(2)  010504  006137  001412               ROL     RUN              ;UPDATE POINTER
(2)  010510  005537  001412               ADC     RUN              ;CATCH CARRY FROM RUN
(2)  010514  062737  000012  001420       ADD     #12,ACTIVE       ;UPDATE ADDRESS POINTER.
(2)  010522  022737  001740  001420       CMP     #DZV.END,ACTIVE  ;HAVE WE PASSED THE END OF THE MAP?
(2)  010530  001356                       BNE     1$               ;IF NO, KEEP GOING; NOT ALL TESTED FOR.
(2)  010532  012737  001500  001420       MOV     #DZV.MAP,ACTIVE  ;RESET ADDRESS POINTER.
(2)  010540  000752                       BR      1$               ;KEEP LOOKING FOR ACTIVE DZV11
(2)  010542  006137  001412        2$:    ROL     RUN              ;UPDATE POINTER.
(2)  010546  005537  001412               ADC     RUN              ;CATCH CARRY.
(2)  010552  013700  001420               MOV     ACTIVE,R0        ;GET ADDRESS POINTER.
(2)  010556  062737  000012  001420       ADD     #12,ACTIVE       ;UPDATE.
(2)  010564  022737  001740  001420       CMP     #DZV.END,ACTIVE
(2)                                                               ;ALL DONE?
(2)  010572  001003                       BNE     3$               ;BR IF NO.
(2)  010574  012737  001500  001420       MOV     #DZV.MAP,ACTIVE  ;RESTORE POINTER.
(2)  010602  012037  001174        3$:    MOV     (R0)+,$BASE      ;LOAD SYSTEM CTRL. REG
(2)  010606  012037  002040               MOV     (R0)+,DZVRIV     ;LOAD VECTOR
(2)  010612  012037  001366               MOV     (R0)+,LINE       ;SET UP DZV LINES ACTIVE
(2)  010616  012037  001370               MOV     (R0)+,PAR        ;SET UP PARAMETERIZATION
(2)  010622  012037  001372               MOV     (R0)+,MODE       ;SET UP MAINTENANCE MODE
(2)  010626  105037  001424               CLRB    MNTFLG  ;RESET MAINT. FLAG IF
(2)  010632  005737  001372               TST     MODE             ;RUNNING TESTS
(2)  010636  001003                       BNE     9$               ;IN
(2)  010640  112737  000010  001424       MOVB    #MAINT,MNTFLG    ;INTERNAL MAINT. MODE
(2)  010646  004737  011014        9$:    JSR     PC,DZVLEV        ;SET UP
(2)  010652  005737  000042               TST     @#42             ;ARE WE UNDER MONITOR CONTROL?
(2)  010656  001051                       BNE     7$               ;IF YES, SKIP THIS SETUP
(2)  010660  032777  000002  170416       BIT     #SW01,@SWR       ;IF SW01=1, GET STARTING TEST #
(2)  010666  001445                       BEQ     7$               ;BR IF NO TEST IS TO BE INPUTTED
(2)  010670  104402  001357        4$:    TYPE    ,$CRLF
(3)  010674  104403                       INSTR                    ;CALL THE STRING INPUT ROUTINE
(3)  010676  007747                       MTSTN                    ;POINTER TO MESSAGE TO BE PRINTED
(3)  010700  104405                       PARAM                    ;CALL THE OCTAL TO ASCII CONVERT ROUTINE
(3)  010702  000001                       1                        ;LOWEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  010704  001000                       1000                     ;HIGHEST LEGITIMATE VALUE OF EXPECTED RESPONSE
(3)  010706  001246                       $TSTNM                   ;POINTER TO MAP LOCATION TO BE FILLED
(3)  010710     000                       .BYTE   0                ;MASK OF INVALID BITS FOR THIS PARAMETER
(3)  010711     001                       .BYTE   1                ;NUMBER OF PARAMETERS TO STORE
(2)  010712  012700  012010               MOV     #TST1,R0
```

```
(2)  010716  022710  000004          5$:     CMP     #4,(R0)
(2)  010722  001020                          BNE     6$
(2)  010724  022760  012737  000002          CMP     #12737,2(R0)
(2)  010732  001014                          BNE     6$
(2)  010734  023760  001246  000004          CMP     $TSTNM,4(R0)    ;IS THIS THE TEST ?
(2)  010742  001010                          BNE     6$              ;IF NOT, DON'T PROCESS NUMBER
(2)  010744  010037  001252                  MOV     R0,$LPADR       ;SAVE PC
(2)  010750  062737  000002  001252          ADD     #2,$LPADR       ;POP OVER PREVIOUS SCOPE
(2)  010756  104402  001357                  TYPE    ,$CRLF
(2)  010762  000412                          BR      8$
(2)  010764  005720                  6$:     TST     (R0)+
(2)  010766  020027  014142                  CMP     R0,#TLAST+10
(2)  010772  001351                          BNE     5$
(2)  010774  104402  001356                  TYPE    ,$QUES
(2)  011000  000733                          BR      4$
(2)  011002  012737  012010  001252  7$:     MOV     #TST1,$LPADR    ;PREPARE TEST ADDRESS
(2)  011010                          8$:
(2)  011010  000177  170236          RESTART:JMP     @$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)  011014  013700  002040          DZVLEV: MOV     DZVRIV,R0       ;PLACE THE BASE VECTOR ADDRESS IN R0
(2)  011020  062700  000002                  ADD     #2,R0           ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
(2)  011024  010037  002042                  MOV     R0,DZVRIS       ;STORE IT HERE
(2)  011030  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
(2)  011034  010037  002044                  MOV     R0,DZVTIV       ;STORE IT HERE
(2)  011040  062700  000002                  ADD     #2,R0           ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
(2)  011044  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)  011050  013700  001174                  MOV     $BASE,R0        ;COPY THE ADDRESS BEING LOADED
(2)  011054  010037  002010                  MOV     R0,DZVCSR       ;XXX0
(2)  011060  005200                          INC     R0
(2)  011062  010037  002012                  MOV     R0,HDZVCSR      ;XXX1
(2)  011066  005200                          INC     R0
(2)  011070  010037  002014                  MOV     R0,DZVRBUF      ;XXX2
(2)  011074  010037  002020                  MOV     R0,DZVLPR       ;XXX2
(2)  011100  005200                          INC     R0
(2)  011102  010037  002016                  MOV     R0,HDZVRBUF     ;XXX3
(2)  011106  010037  002022                  MOV     R0,HDZVLPR      ;XXX3
(2)  011112  005200                          INC     R0
(2)  011114  010037  002024                  MOV     R0,DZVTCR       ;XXX4
(2)  011120  005200                          INC     R0
(2)  011122  010037  002026                  MOV     R0,HDZVTCR      ;XXX5
(2)  011126  005200                          INC     R0
(2)  011130  010037  002030                  MOV     R0,DZVMSR       ;XXX6
(2)  011134  010037  002034                  MOV     R0,DZVTDR       ;XXX6
(2)  011140  005200                          INC     R0
(2)  011142  010037  002032                  MOV     R0,HDZVMSR      ;XXX7
(2)  011146  010037  002036                  MOV     R0,HDZVTDR      ;XXX7
(2)  011152  000207                          RTS     PC
```

```
(2)                                     ;CONVERT DECIMAL ASCII STRING TO OCTAL
(2)  011154  000002                .PARMD: RTI             ; DECIMAL PARAMETERS UNUSED.          ;;GPA
(2)                                .REM &                  ; DELETED TO CONSERVE SPACE...        ;;GPA
(2)                                                        ;...AND REMAIN UNDER 4KW SIZE.        ;;GPA
(2)                                .PARMD: MOV     (SP),R5
(2)                                        MOV     (R5)+,6$
(2)                                        MOV     (R5)+,7$
(2)                                        MOV     (R5)+,8$
(2)                                        MOVB    (R5)+,9$
(2)                                        MOVB    (R5)+,10$
(2)                                        MOV     R5,(SP)
(2)                                2$:     CLR     R5
(2)                                        MOV     #INBUF,R4
(2)                                        CMPB    #15,(R4)
(2)                                        BEQ     3$
(2)                                1$:     CMPB    (R4),#'0
(2)                                        BLT     3$
(2)                                        CMPB    (R4),#'9
(2)                                        BGT     3$
(2)                                        BICB    #'0,(R4)
(2)                                        CLR     R2
(2)                                        BISB    (R4)+,R2
(2)                                        ADD     R2,R5
(2)                                        CMPB    #15,(R4)
(2)                                        BEQ     4$
(2)                                        ASL     R5      ;X2
(2)                                        MOV     R5,R2   ;SAVE X2
(2)                                        ASL     R5      ;X4
(2)                                        ASL     R5      ;X8
(2)                                        ADD     R2,R5   ;TIMES 10
(2)                                        BR      1$
(2)                                3$:     INSTER
(2)                                        BR      2$
(2)
(2)                                        ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(2)
(2)                                4$:     CMP     R5,7$
(2)                                        BHI     3$
(2)                                        CMP     R5,6$
(2)                                        BLO     3$
(2)                                        BITB    9$,R5
(2)                                        BNE     3$
(2)
(2)                                        ;STORE NUMBER AT SPECIFIED ADDRESS
(2)
(2)                                        MOV     8$,R4
(2)                                5$:     MOV     R5,(R4)+
(2)                                        ADD     #2,R5
(2)                                        DECB    10$
(2)                                        BNE     5$
(2)                                        RTI
(2)                                6$:     0
(2)                                7$:     0
(2)                                8$:     0
(2)                                9$:     .BYTE 0
(2)                                10$:    .BYTE 0
```

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

```
(2)                                             ;*ROUTINE USED TO SET UP THE DIAGNOSTIC VIA APT.
(2)                                             ;*IF BIT7 IN THE ENVIRONMENT MODE ($ENVM) BYTE IS SET,
(2)                                             ;*THE PROGRAM WILL LOAD ITS PARAMETERS FROM THE ETABLE.
(2)
(2)  011156  012700  001500          SETAPT:    MOV     #DZV.MAP,R0     ;POINT TO THE DEVICE MAP TABLE
(2)  011162  013701  001174                     MOV     $BASE,R1        ;BUILD DEVICE ADDRESSES IN R1
(2)  011166  013702  001170                     MOV     $VECT1,R2       ;BUILD DEVICE VECTORS IN R2
(2)  011172  042702  177007                     BIC     #^C<770>,R2     ;STRIP AWAY OTHER INFORMATION
(2)  011176  012704  001204                     MOV     #$DDW0,R4       ;POINT TO THE BEGINNING OF DEVICE PARAMETERS
(2)  011202  013705  001176                     MOV     $DEVM,R5        ;GET THE MAP OF ACTIVE DEVICES
(2)  011206  105037  001414                     CLRB    DZVNUM          ;INITIALIZE THE NO. OF ACTIVE DEVICES
(2)  011212  005037  001410                     CLR     SAVACTV         ;CLEAR THE ACTIVE BIT MAP
(2)  011216  006005                    1$:      ROR     R5              ;GET A DEVICE SELECTION BIT
(2)  011220  103407                             BCS     3$              ;IF IT IS SELECTED, GO SET UP A MAP
(2)  011222  001422                             BEQ     5$              ;IF NO MORE ARE SELECTED, GET OUT OF SETUP
(2)  011224  005724                             TST     (R4)+           ;POINT TO NEXT DEVICE DESCRIPTOR
(2)  011226  062701  000010            2$:      ADD     #10,R1          ;SET UP THE NEXT ADDRESS
(2)  011232  062702  000010                     ADD     #10,R2          ;SET UP THE NEXT VECTOR GROUP
(2)  011236  000767                             BR      1$              ;GO SEE IF MORE DEVICES REMAIN
(2)  011240  006137  001410            3$:      ROL     SAVACTV         ;SET BIT IN ACTIVE DEVICE MAP
(2)  011244  105237  001414                     INCB    DZVNUM          ;INCREMENT NO. OF ACTIVE DEVICES
(2)  011250  010120                             MOV     R1,(R0)+        ;LOAD DEVICE ADDRESS
(2)  011252  010220                             MOV     R2,(R0)+        ;LOAD THE VECTOR ADDRESS
(2)  011254  013720  001200                     MOV     $CDW1,(R0)+     ;GET THE NUMBER OF LINES IN OPERATION
(2)  011260  012420                             MOV     (R4)+,(R0)+     ;LOAD DEVICE PARAMETERS
(2)  011262  013720  001202                     MOV     $CDW2,(R0)+     ;LOAD DEFAULT TESTING MODE
(2)  011266  000757                             BR      2$              ;GO BUILD THE NEXT ADDRESS
(2)  011270  012710  177777            5$:      MOV     #-1,(R0)        ;TERMINATE THE DEVICE MAP
(2)  011274  012737  001142  001304             MOV     #$SWREG,SWR     ;SET TO SOFTWARE APT SWITCH REGISTER
(2)  011302  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)  011304                            AUTO.SIZE:
(2)  011304  000005                             RESET                   ;INSURE A BUS INIT.
(2)  011306  105337  001422                     DECB    INIFLG          ;SHOW THAT I WAS HERE
(2)  011312  012702  001500            CSRMAP:  MOV     #DZV.MAP,R2     ;LOAD MAP POINTER.
(2)  011316  012703  001204                     MOV     #$DDW0,R3       ;POINT TO ETABLE DEVICE DESCRIPTOR WORDS
(2)  011322  005022                    1$:      CLR     (R2)+           ;ZERO ENTIRE MAP
(2)  011324  022702  001740                     CMP     #DZV.END,R2     ;ALL DONE?
(2)  011330  001374                             BNE     1$              ;BR IF NO
(2)  011332  105037  001414                     CLRB    DZVNUM          ;SET OCTAL NUMBER OF DZV11'S TO 0
(2)  011336  012702  001500                     MOV     #DZV.MAP,R2
(2)  011342  012701  160000                     MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
(2)  011346  012737  011612  000004             MOV     #6$,@#4         ;SET FOR NON-EXISTENT DEVICE TIME OUT
(2)  011354  052711  000040            2$:      BIS     #BIT5,(R1)      ;TRY TO SET MASTER SCAN ENABLE
(2)  011360  052761  000017  000004             BIS     #17,4(R1)       ;TRY TO TRANSMIT ON ANY LINE
(2)  011366  005000                             CLR     R0              ;USE R0 AS A COUNTER
(2)  011370  005711                    7$:      TST     (R1)            ;HAS TRANSMITTER READY COME UP?
```

```
(2)  011372  100403                          BMI     8$              ;IF SO, GO GET A FINAL CHECK
(2)  011374  005300                          DEC     R0              ;REDUCE COUNT. TIME UP?
(2)  011376  001374                          BNE     7$              ;IF NOT, KEEP WAITING
(2)  011400  000437                          BR      3$              ;ASSUME IT'S NOT A DZV11
(2)  011402  032761  000017  000004  8$:     BIT     -#17,4(R1)      ;ARE ANY TCR BITS  STILL SET? THEY SHOULD BE
(2)  011410  001433                          BEQ     3$              ;IF IT'S NOT, ASSUME IT'S NOT A DZV11
(2)  011412  032711  000040                  BIT     #BIT5,(R1)      ;IS MASTER SCAN ENABLE STILL SET?
(2)  011416  001430                          BEQ     3$              ;IF NOT, ASSUME IT'S NOT A DZV11
(2)  011420  052711  000020                  BIS     #20,(R1)        ;SET DEVICE CLEAR
(2)  011424  000240                          NOP
(2)  011426  032711  000040                  BIT     #40,(R1)        ;DID SCANNER CLEAR
(2)  011432  001022                          BNE     3$              ;IF NOT ASSUME IT IS NOT DZV
(2)  011434  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)  011440  010122                          MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
(2)  011442  005722                          TST     (R2)+           ;POP OVER VECTOR STORE AREA
(2)  011444  012722  000017                  MOV     #17,(R2)+       ;SET THE DEFAULT LINE SELECTION PARAMETER
(2)  011450  012712  017470                  MOV     #17470,(R2)     ;SET THE DEFAULT PARAMETERS
(2)  011454  012223                          MOV     (R2)+,(R3)+     ;COPY PARAMETERS INTO ETABLE DESCRIPTOR
(2)  011456  005022                          CLR     (R2)+           ;SET THE DEFAULT MODE OF OPERATION
(2)  011460  012712  177777                  MOV     #-1,(R2)        ;TERMINATE LIST
(2)  011464  105237  001414                  INCB    DZVNUM          ;UPDATE DEVICE COUNTER
(2)  011470  122737  000020  001414          CMPB    #20,DZVNUM      ;ARE MAX. NO. OF DEV FOUND?
(2)  011476  001405                          BEQ     100$            ;YES DON'T LOOK FOR ANY MORE.
(2)  011500  062701  000010          3$:     ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
(2)  011504  022701  164000                  CMP     #164000,R1
(2)  011510  001321                          BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
(2)  011512                          100$:
(2)  011512  105737  001414                  TSTB    DZVNUM          ;WERE ANY DZV11'S FOUND AT ALL?
(2)  011516  001430                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO DZV11'S IN THIS SYS.
(2)  011520  113701  001414                  MOVB    DZVNUM,R1
(2)  011524  012737  000001  001410          MOV     #1,SAVACTV      ;CREATE A BIT MAP OF
(2)  011532  005301                  4$:     DEC     R1              ;THE DEVICES IN THE SYSTEM
(2)  011534  001404                          BEQ     98$
(2)  011536  000261                          SEC
(2)  011540  006137  001410                  ROL     SAVACTV
(2)  011544  000772                          BR      4$
(2)  011546  013737  001500  001174  98$:     MOV     DZCR0,$BASE     ;POINT TO THE ADDRESS OF FIRST DEVICE
(2)  011554  013737  001510  001202          MOV     MANT0,$CDW2     ;INDICATE TO ETABLE WHAT MODE IS BEING USED
(2)  011562  012737  000006  000004  99$:     MOV     #6,@#4          ;RESTORE TRAP VECTOR
(2)  011570  013737  001410  001176          MOV     SAVACTV,$DEVM   ;SAVE ACTIVE REGISTER
(2)  011576  000410                          BR      VECMAP          ;GO FIND THE VECTOR NOW.
(2)  011600  104402  007567          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO DZV11'S FOUND.
(2)  011604  005000                          CLR     R0              ;MAKE DATA DISPLAY ZERO
(2)  011606  000000                          HALT                    ;STOP THE SHOW
(2)  011610  000776                          BR      .-2             ;DISABLE CONT. SW.
(2)  011612  012716  011500          6$:     MOV     #3$,(SP)        ;ENTERED BY NON-EXISTENT TIME-OUT
(2)  011616  000002                          RTI                     ;RETURN TO MAINSTREAM
(2)
(2)  011620  012737  000200  000022  VECMAP: MOV     #MASK,@#22      ;SET IOT TRAP PRIORITY
(2)  011626  012737  011742  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
(2)  011634  012702  001500                  MOV     #DZV.MAP,R2     ;SET SOFTWARE POINTER
(2)  011640  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
(2)  011644  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
(2)  011650  010120                  1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
(2)  011652  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
```

```
(2)  011656  022021                      CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
(2)  011660  020127  001000              CMP     R1,#1000        ;HAS THE VECTOR AREA BEEN EXCEEDED?
(2)  011664  101771                      BLOS    1$              ;BR IF MORE TO FILL
(2)  011666  013704  001410              MOV     SAVACTV,R4      ;STORE TEMPORARILY
(2)  011672  006004              2$:     ROR     R4              ;BRING OUT A BIT
(2)  011674  103036                      BCC     5$              ;BR IF ALL DONE
(2)  011676  106427  000000              MTPS    #0              ;ZERO CPU PRIO
(2)  011702  012772  040040  000000      MOV     #BIT14+BIT5,@(R2)       ;SET TIE AND MAS SCAN
(2)  011710  011201                      MOV     (R2),R1         ;GET CSR
(2)  011712  112761  000017  000004      MOVB    #17,4(R1)       ;SET THE TCR BITS FOR ALL LINES
(2)                                                              ;ATTEMPT TO FORCE AN INTERRUPT
(2)  011720  005200                      INC     R0              ;STALL
(2)  011722  001376                      BNE     .-2             ;       FOR TIME TO INTERRUPT
(2)  011724  012762  000300  000002      MOV     #300,2(R2)      ;NO INTERRUPT ASSUME 300 AND FIX DZV11 LATER
(2)  011732  000005                      RESET                   ;INIT
(2)  011734  062702  000012      3$:     ADD     #12,R2          ;POP SOFTWARE POINTER
(2)  011740  000754                      BR      2$              ;KEEP GOING
(2)  011742  011662  000002      4$:     MOV     (SP),2(R2)      ;GET VECTOR ADDRESS
(2)  011746  162762  000010  000002      SUB     #10,2(R2)       ;POINT BACK TO THE CORRECT VECTOR
(2)  011754  042762  000007  000002      BIC     #7,2(R2)        ;CLEAR JUNK
(2)  011762  022626                      POP2SP                  ;POP IOT JUNK OFF STACK
(2)  011764  012716  011734              MOV     #3$,(SP)        ;SET FOR RETURN
(2)  011770  000002                      RTI
(2)  011772  013737  001502  001170  5$:  MOV     DZVC0,$VECT1    ;COPY VECTOR OF FIRST DEVICE INTO ETABLE
(2)  012000  012737  004354  000020      MOV     #.SCOPE,IOTVEC  ;RESTORE THE SCOPE TRAP
(2)  012006  000207                      RTS     PC              ;ALL DONE WITH "AUTO SIZING"
(2)
```

```
 8409
 8410                                   ;************************** TEST 1 *********************************
 8411                                   ;*THIS TEST VERIFIES OVERRUN AND SILO ALARM
 8412                                   ;*ONE LINE AT A TIME  - BASED UPON VALID LINES
 8413                                   ;*AS EACH OF THE FIRST 16 CHARS ARE SENT; SILO ALARM IS
 8414                                   ;*TESTED TO BE CLEARED. ON THE 16TH CHAR THE PROGRAM THEN
 8415                                   ;*EXPECTS SILO ALARM TO SET. THEN THE ENTIRE
 8416                                   ;*SILO IS FILLED AND AN OVERRUN IS EXPECTED ON THE 65TH
 8417                                   ;*CHAR PULLED OUT OF THE SILO.
 8418                                   ;*ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
 8419                                   ;*USING SWITCH NINE FOR THIS TEST SENDS 20. CHARACTERS
 8420                                   ;*ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
 8421                                   ;*USED TO SCOPE SILO ALARM PULSES, ETC.
 8423                                ;:*  TEST 1
  (5)                                ;;*****************************************************************
  (4)  012010  000004                TST1:   SCOPE
  (4)
  (2)  012012  012737  000001  001246        MOV     #1,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
  (2)  012020  012737  012452  001362        MOV     #TST2,NEXT      ;POINT TO THE START OF THE NEXT TEST
 8424  012026  012737  012366  001364        MOV     #18$,LOCK       ;SET FOR LOOP
 8425  012034  104417                        DCLASM                  ;SET DCLR IN CSR AND SET MNTFLG
 8426  012036  104421                        LPRSET                  ;LOAD LINE PARAMETERS
 8427  012040  005037  001374                CLR     SAVLIN          ;INIT LINE INDICATOR
 8428  012044  104422                        BUFSET                  ;ZERO DATA BUFFER
 8429  012046  012702  000001                MOV     #1,R2           ;LINE POINTER
 8430  012052  052777  010040  167730        BIS     #MSENAB!SILOEN,@DZVCSR  ;START SCANNER & SET SILO ENABLE
 8431  012060  030237  001366        3$:     BIT     R2,LINE         ;VALID LINE?
 8432  012064  001533                        BEQ     21$             ;IF NOT GO TO NEXT LINE
 8433  012066  013700  001374                MOV     SAVLIN,R0       ;MAKE OFFSET
 8434  012072  006300                        ASL     R0              ;MAKE POWER OF TWO
 8435  012074  010277  167724                MOV     R2,@DZVTCR      ;SET TCR BIT
 8436  012100  105777  167704        4$:     TSTB    @DZVCSR         ;REC DONE = 1 ?
 8437  012104  100001                        BPL     .+4
 8438  012106  104020                        ERROR   20              ;REC DONE SHOULD NOT = 1
 8439  012110  005003                        CLR     R3              ;SET CHARACTER COUNT
 8440  012112  005004                5$:     CLR     R4
 8441  012114  005777  167670        6$:     TST     @DZVCSR         ;IS TRDY SET?
 8442  012120  100404                        BMI     7$              ;IF YES, LOAD CHAR.
 8443  012122  104414                        DELAY                   ;WAIT FOR TRDY TO SET
 8444  012124  005204                        INC     R4              ;INC DELAY COUNTER
 8445  012126  001372                        BNE     6$
 8446  012130  104003                        ERROR   3               ;*TRDY FAILED TO SET
 8447  012132  116077  001426  167674  7$:   MOVB    TDO(R0),@DZVTDR ;LOAD A CHARACTER
 8448  012140  005260  001426                INC     TDO(R0)         ;SET UP NEXT CHARACTER
 8449  012144  020327  000017                CMP     R3,#15.         ;16 CHARACTERS ?
 8450  012150  103006                        BHIS    8$
 8451  012152  032777  020000  167630        BIT     #SILOAL,@DZVCSR ;SILO ALARM = 0 ?
 8452  012160  001413                        BEQ     10$             ;YES
 8453  012162  104013                        ERROR   13              ;*SILO ALARM SHOULD NOT = 1
 8454                                                                ;UNTIL 16. DATA CHARACTERS
 8455  012164  000411                        BR      10$
 8456  012166  005004                8$:     CLR     R4
 8457  012170  032777  020000  167612  9$:   BIT     #SILOAL,@DZVCSR
 8458  012176  001004                        BNE     10$
 8459  012200  104414                        DELAY
 8460  012202  005204                        INC     R4
```

```
 8461  012204  001371                          BNE     9$
 8462  012206  104014                          ERROR   14              ;*SILO ALARM FAILED TO SET!
 8463                                                                  ;SILO ALARM SHOULD =1 AFTER 16.
 8464                                                                  ;DATA CHARACTERS
 8465  012210  005203                  10$:    INC     R3              ;INC CHAR COUNT
 8466  012212  022703  000102                  CMP     #66.,R3         ;FINISHED SENDING CHARACTERS ?
 8467  012216  001335                          BNE     5$              ;NO
 8468  012220  005004                          CLR     R4
 8469  012222  104414                          DELAY
 8470  012224  105204                          INCB    R4
 8471  012226  001375                          BNE     .-4
 8472                                          ;NOW LETS READ THE SILO
 8473  012230  013705  001374                  MOV     SAVLIN,R5       ;MAKE EXPECTED LINE #
 8474  012234  005737  001372                  TST     MODE            ;IS THIS TEST IN STAGGERED MODE?
   (1) 012240  100006                          BPL     13$             ;IF NOT, SKIP STAGGERED SETUP
   (1)
   (1)                                        ;WE MUST NOW INVERT THE LAST BIT OF THE LINE NUMBER
   (1)
   (1)  012242  006205                        ASR     R5              ;GET THE LAST BIT INTO THE CARRY BIT
   (1)  012244  103402                        BCS     11$             ;IF IT IS SET, GO CLEAR IT
   (1)  012246  000261                        SEC                     ;IF IT IS CLEAR SET IT HERE
   (1)  012250  000401                        BR      12$             ;SKIP THE CLEARING
   (1)  012252  000241                11$:    CLC                     ;CLEAR THE CARRY BIT (INVERSION OF LINE PARITY)
   (1)  012254  006105                12$:    ROL     R5              ;GET THE NEW BIT BACK INTO R5
 8475  012256  000305                13$:    SWAB    R5              ;PUT IN UPPER BYTE
 8476  012260  052705  100000                BIS     #DVALID,R5      ;ADD DATA VALID
 8477  012264  017704  167524          14$:  MOV     @DZVRBUF,R4     ;ACTUAL
 8478  012270  020405                        CMP     R4,R5           ;ACTUAL VS. EXPECTED
 8479  012272  001401                        BEQ     15$             ;YES
 8480  012274  104006                        ERROR   6               ;*DATA/CONTENTS DID NOT COMPARE
 8481  012276  032777  020000  167504  15$:  BIT     #SILOAL,@DZVCSR ;SILO ALARM= 0 ?
 8482  012304  001401                        BEQ     16$             ;YES
 8483  012306  104016                        ERROR   16              ;READING DZVRBUF DID NOT CLEAR SILO ALARM
 8484  012310  005205                16$:    INC     R5              ;UP CHARACTER
 8485  012312  120527  000077                CMPB    R5,#63.         ;LAST SILO CHAR ?....64TH CHAR
 8486  012316  101762                        BLOS    14$
 8487  012320  005205                        INC     R5              ;ADD 1 MORE FOR THE CLOBBERED CHAR
 8488  012322  052705  040000                BIS     #OVRRUN,R5      ;ADD OVERRUN TO EXPECTED
 8489  012326  120527  000101                CMPB    R5,#65.         ;LAST CHARACTER ?
 8490  012332  001754                        BEQ     14$
 8491  012334  017704  167454                MOV     @DZVRBUF,R4     ;FOR GOOD MEASURE
 8492  012340  005704                        TST     R4              ;DATA VALID SHOULD = 0
 8493  012342  100001                        BPL     17$             ;YES
 8494  012344  104017                        ERROR   17              ;DATA VALID SHOULD = 0
 8495  012346  040277  167452          17$:  BIC     R2,@DZVTCR      ;CLR TCR BIT
 8496  012352  104401                        SCOP1                   ;LOOP?
 8497  012354  005237  001374          21$:  INC     SAVLIN          ;INC EXPECTED LINE
 8498  012360  104420                        SHIFT                   ;NEXT LINE
 8499  012362  000137  012060                JMP     3$              ;YES
 8500
 8501                                        ;TIGHT SCOPE LOOP FOR THIS TEST. SENDS 20. CHARACTERS
 8502                                        ;ON DZV LINE PREVIOUSLY SELECTED CONTINUOUSLY WHILE SW09=1.
 8503                                        ;USED TO SCOPE SILO ALARM PULSES, ETC.
 8504
 8505  012366  052777  010040  167414  18$:  BIS     #MSENAB!SILOEN,@DZVCSR  ;SETUP DEVICE
 8506  012374  012777  012442  167442        MOV     #20$,@DZVTIV    ;SETUP TRANSMITTER VECTOR
```

```
 8507  012402  012701  000024                    MOV     #20.,R1          ;TEMPORARY COUNT OF CHARACTER BURST
 8508  012406  050277  167412                    BIS     R2,@DZVTCR       ;ENABLE LINE
 8509  012412  052777  040000  167370            BIS     #TIE,@DZVCSR     ;ENABLE INTERRUPTS
 8510  012420  106427  000000                    MTPS    #0               ;LOWER PRIORITY
 8511  012424  000001                19$:        WAIT                     ;ALLOW INTERRUPTS
 8512  012426  077102                            SOB     R1,19$           ;REDUCE COUNT. ALL CHARACTERS SENT?
 8513  012430  042777  050040  167352            BIC     #SILOEN!MSENAB!TIE,@DZVCSR ;RESET SILO COUNTER, CLEAR STROBE
 8514  012436  104401                            SCOP1                    ;LOOP AGAIN?
 8515  012440  000742                            BR      17$              ;IF NOT, RETURN TO WHERE YOU LEFT OFF
 8516  012442  112777  000252  167364  20$:     MOVB    #252,@DZVTDR     ;SEND A CHARACTER
 8517  012450  000002                            RTI                      ;ALLOW MORE CHARACTERS TO COME
 8518                                            ;************************ TEST 2 *******************************
 8519                                            ;*THIS TEST THAT "SILO ENABLE" WILL INHIBIT
 8520                                            ;*RECEIVER INTERRUPTS AND THAT ON THE
 8521                                            ;*16TH CHAR THAT "SILO ALARM" WILL CAUSE AN
 8522                                            ;*INTERRUPT WITH "RIE" SET.
 8523                                            ;*THIS WILL DO ALL SELECTED LINES ONE AT A TIME.
 8524                                            ;*ERROR PRINTOUTS WILL REPORT TRANSMITTING LINE NO.
 8526                                    ;:*  TEST 2
  (5)                                    ;;*****************************************************************
  (4)   012452  000004                  TST2:   SCOPE
  (4)
  (2)   012454  012737  000002  001246            MOV     #2,$TSTNM        ;LOAD THE NUMBER OF THIS TEST
  (2)   012462  012737  012752  001362            MOV     #TST3,NEXT       ;POINT TO THE START OF THE NEXT TEST
 8527   012470  012737  012514  001364            MOV     #3$,LOCK         ;SET FOR LOOP
 8528   012476  104417                            DCLASM                   ;SET DCLR IN CSR AND SET MNTFLG
 8529   012500  104421                            LPRSET                   ;LOAD LINE PARAMETERS
 8530   012502  005037  001374                    CLR     SAVLIN           ;INIT LINE INDICATOR
 8531   012506  104422                            BUFSET                   ;ZERO DATA BUFFER
 8532   012510  012702  000001                    MOV     #1,R2            ;LINE POINTER
 8533   012514  012777  012724  167316  3$:      MOV     #11$,@DZVRIV     ;SET FOR UNEXPECTED INTER.
 8534   012522  012777  000200  167312            MOV     #MASK,@DZVRIS    ;SET PRIO.
 8535   012530  052777  010140  167252            BIS     #MSENAB!SILOEN!RIE,@DZVCSR
 8536                                                                     ;START SCANNER & SET SILO ENABLE
 8537   012536  030237  001366                    BIT     R2,LINE          ;VALID LINE?
 8538   012542  001477                            BEQ     18$              ;IF NOT GO TO NEXT LINE
 8539   012544  005777  167244                    TST     @DZVRBUF         ;EMPTY THE SILO
 8540   012550  100775                            BMI     .-4              ;BR IF DATA VALID IS SET!
 8541   012552  106427  000000                    MTPS    #0               ;SET PROCESSOR PRIORITY TO 0
 8542   012556  013700  001374                    MOV     SAVLIN,R0        ;MAKE OFFSET
 8543   012562  006300                            ASL     R0               ;MAKE POWER OF TWO
 8544   012564  010277  167234                    MOV     R2,@DZVTCR       ;SET TCR BIT
 8545   012570  005004                  5$:       CLR     R4
 8546   012572  005777  167212          6$:       TST     @DZVCSR
 8547   012576  100404                            BMI     7$
 8548   012600  104414                            DELAY
 8549   012602  005204                            INC     R4
 8550   012604  001372                            BNE     6$
 8551   012606  104003                            ERROR   3                ;*TRDY FAILED TO SET
 8552   012610  116077  001426  167216  7$:      MOVB    TDO(R0),@DZVTDR  ;LOAD A CHARACTER
 8553   012616  005260  001426                    INC     TDO(R0)          ;SET UP NEXT CHARACTER
 8554   012622  022760  000017  001426            CMP     #15.,TDO(R0)     ;15 CHARS YET?
 8555   012630  001406                            BEQ     8$
 8556   012632  032777  020000  167150            BIT     #SILOAL,@DZVCSR  ;SILO ALARM = 0 ?
 8557   012640  001401                            BEQ     .+4              ;YES
 8558   012642  104013                            ERROR   13               ;*SILO ALARM SHOULD NOT = 1
```

```
                                                                    ;UNTIL 16. DATA CHARACTERS
012644  000752                          BR      6$
012646  012777  012732  167164  8$:     MOV     #12$,@DZVRIV        ;SET NEW VECTOR
012654  005777  167130                  TST     @DZVCSR             ;READY FOR 16TH CHAR
012660  100375                          BPL     .-4
012662  016077  001426  167144          MOV     TDO(R0),@DZVTDR     ;LOAD THE 16TH CHAR.
012670  005004                          CLR     R4
012672  032777  020000  167110  9$:     BIT     #SILOAL,@DZVCSR
012700  001005                          BNE     10$
012702  104414                          DELAY
012704  005204                          INC     R4
012706  001371                          BNE     9$
012710  104014                          ERROR   14                  ;*SILO ALARM FAILED TO SET!
012712  000410                          BR      17$                 ;SILO ALARM SHOULD =1 AFTER 16.
                                                                    ;DATA CHARACTERS
012714  000240                  10$:    NOP                         ;STALL
012716  000240                          NOP
012720  104027                          ERROR   27                  ;SILO ALARM NOT INTERRUPTING.
012722  000404                          BR      17$                 ;CONTINUE TEST.
012724  022626                  11$:    POP2SP                      ;FAKE RTI
012726  104012                          ERROR   12                  ;RX SHOULD NOT INTERRUPT
012730  000401                          BR      17$                 ;CONTINUE
012732  022626                  12$:    POP2SP                      ;GOOD INTERRUPT TO HERE.
012734  040277  167064          17$:    BIC     R2,@DZVTCR          ;CLR TCR BIT
012740  104401                          SCOP1                       ;LOOP?
012742  005237  001374          18$:    INC     SAVLIN              ;INC EXPECTED LINE
012746  104420                          SHIFT                       ;NEXT LINE
012750  000661                          BR      3$                  ;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
                                ;:*************************************************************************
012752  000004                 TST3:   SCOPE
012754  012737  000003  001246          MOV     #3,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
012762  012737  013514  001362          MOV     #TST4,NEXT      ;POINT TO THE START OF THE NEXT TEST
012770  104417                          DCLASM                  ;SET DCLR IN CSR AND SET MNTFLG
012772  013737  001366  013512          MOV     LINE,RXTCR      ;SET IMAGE OF TCR BITS
013000  013737  001366  013232          MOV     LINE,TXTCR      ;SET IMAGE OF TCR BITS
013006  104421                          LPRSET                  ;LOAD LINE PARAMETERS
013010  104422                          BUFSET                  ;ZERO DATA BUFFER
013012  012777  013234  167020          MOV     #RXSVC,@DZVRIV  ;SET UP REC INTR VECTOR
013020  012777  000200  167014          MOV     #MASK,@DZVRIS   ;STATUS
013026  012777  013120  167010          MOV     #TXSVC,@DZVTIV  ;SET UP TRANS INTR VECTOR
013034  012777  000200  167004          MOV     #MASK,@DZVTIS   ;STATUS
013042  052777  040140  166740          BIS     #MSENAB!RIE!TIE,@DZVCSR ;SET MASTER SCAN ENABLE
013050  113777  001366  166746          MOVB    LINE,@DZVTCR    ;SET TCR BITS
013056  106427  000000                  MTPS    #CLEAR          ;ALLOW INTERRUPTS

013062  005037  013116          SNAP:   CLR     4$              ;CLEAR DELAY COUNTER
013066  104414                  2$:     DELAY                   ;WAIT FOR RECEIVERS TO FINISH
013070  105737  013512                  TSTB    RXTCR           ;WAIT FOR ALL RECIEVERS TO FINISH
013074  001002                          BNE     3$              ;
013076  000137  013440                  JMP     OUT
013102  005237  013116          3$:     INC     4$              ;INCREMENT DELAY COUNTER
013106  001367                          BNE     2$              ;DELAY FINISHED?
013110  104007                          ERROR   7               ;*TRANSMITTER FAILED TO INTERRUPT
013112  104011                          ERROR   11              ;*RECEIVER FAILED TO INTERRUPT
013114  104400                          ADVANCE                 ;LEAVE THIS TEST
013116  000000                  4$:     0

                                ;TRANS INTR SVC ROUTINE
013120  117703  166666          TXSVC:  MOVB    @HDZVCSR,R3     ;FIND LINE NO.
013124  042703  177774                  BIC     #^C<3>,R3       ;ISOLATE LINE NO.
013130  010304                          MOV     R3,R4           ;SAVE LINE NO.
013132  010337  001374                  MOV     R3,SAVLIN       ;SAVE LINE NO.
013136  005777  166646                  TST     @DZVCSR         ;TRANS READY SET ?
013142  100401                          BMI     .+4
013144  104003                          ERROR   3               ;*TRANSMITTER FAILED
013146  012702  000001                  MOV     #1,R2           ;SET UP POSITION POINTER
013152  105303                  3$:     DECB    R3              ;IS IT THIS LINE ?
013154  100402                          BMI     4$              ;YES
013156  006302                          ASL     R2              ;UP THE LINE #
013160  000774                          BR      3$              ;GO 'ROUND AGAIN
013162  030237  001366          4$:     BIT     R2,LINE         ;VALID LINE?
013166  001001                          BNE     .+4             ;YES
013170  104015                          ERROR   15              ;NO,INVALID LINE!!!!
013172  030237  013232                  BIT     R2,TXTCR        ;DATA FINISHED?
013176  001003                          BNE     6$              ;IF NOT SEND CHAR.
013200  040277  166620                  BIC     R2,@DZVTCR      ;CLEAR TCR BIT
```

```
8640  013204  000411                        BR      5$              ;RETURN
8641  013206  006304                6$:     ASL     R4              ;MAKE POWER OF 2
8642  013210  116477  001426  166616        MOVB    TDO(R4),@DZVTDR ;LOAD CHARACTER
8643  013216  105264  001426                INCB    TDO(R4)         ;SET UP NEXT CHARACTER
8644  013222  001002                        BNE     5$              ;LAST CHARACTER ?
8645  013224  040237  013232                BIC     R2,TXTCR        ;INDICAT LINE FINISHED
8646  013230  000002                5$:     RTI
8647
8648  013232  000000                TXTCR:  0
8649
8650                                ;REC INTR SVC ROUTINE
8651  013234  105777  166550        RXSVC:  TSTB    @DZVCSR         ;REC DONE ?
8652  013240  100401                        BMI     .+4             ;YES
8653  013242  104004                        ERROR   4               ;FALSE INTERRUPT
8654  013244  032777  020000  166536        BIT     #SILOAL,@DZVCSR ;SILO ALARM?
8655  013252  001401                        BEQ     .+4             ;NO
8656  013254  104013                        ERROR   13              ;SILO ALARM SHOULD NOT =1
8657  013256  017704  166532                MOV     @DZVRBUF,R4     ;SAVE IT
8658  013262  010403                        MOV     R4,R3
8659  013264  000303                        SWAB    R3
8660  013266  042703  177774                BIC     #^C<3>,R3       ;STRIP JUNK
8661  013272  010337  001374                MOV     R3,SAVLIN       ;SAVE LINE NUMBER
8662  013276  005704                        TST     R4              ;DATA VALID?
8663  013300  100401                        BMI     4$              ;IF YES SKIP ERROR PRINTOUT
8664  013302  104023                        ERROR   23              ;YOU LOSE  ...DATA VALID WAS'NT SET
8665  013304  032704  040000        4$:     BIT     #OVRRUN,R4      ;TEST FOR OVERRUN
8666  013310  001401                        BEQ     1$              ;IF NO OVERRUN SKIP ERROR
8667  013312  104024                        ERROR   24              ;DATA OVERRUN
8668  013314  032704  020000        1$:     BIT     #FRMERR,R4      ;DATA FRAMING ERROR
8669  013320  001401                        BEQ     2$              ;IF NO FRAMING ERROR CONTINUE
8670  013322  104025                        ERROR   25              ;FRAMING ERROR
8671  013324  032704  010000        2$:     BIT     #PARER,R4       ;TEST FOR PARITY ERROR
8672  013330  001401                        BEQ     3$              ;BRANCH IF NO ERROR
8673  013332  104026                        ERROR   26              ;TYPE OUT PARITY ERROR
8674  013334  012702  000001        3$:     MOV     #1,R2           ;SET UP POSITION POINTER
8675  013340  105303                5$:     DECB    R3
8676  013342  100402                        BMI     6$
8677  013344  006302                        ASL     R2              ;RE POSITION POINTER
8678  013346  000774                        BR      5$              ;GO 'ROUND AGAIN
8679  013350  030237  001366        6$:     BIT     R2,LINE         ;LINE VALID ?
8680  013354  001001                        BNE     .+4             ;YES
8681  013356  104015                        ERROR   15              ;INVALID LINE #
8682  013360  013703  001374                MOV     SAVLIN,R3       ;GET THE LINE NUMBER AGAIN
8683  013364  006303                        ASL     R3              ;USE R3 AS A POINTER IN THE DATA TABLE
8684  013366  126304  001436                CMPB    TRO(R3),R4      ;DOES THE DATA CHARACTER COMPARE ?
8685  013372  001410                        BEQ     7$              ;YES
8686  013374  013705  001374                MOV     SAVLIN,R5       ;MOVE LINE NO INTO EXPECTED
8687  013400  000305                        SWAB    R5              ;ADJUST TO HIGH BYTE
8688  013402  052705  100000                BIS     #DVALID,R5      ;SET DVALID IN EXPECTED
8689  013406  056305  001436                BIS     TRO(R3),R5      ;SET DATA IN EXPECTED
8690  013412  104005                        ERROR   5               ;*NO, DATA DOES NOT COMPARE
8691  013414  005263  001436        7$:     INC     TRO(R3)         ;SET UP FOR NEXT CHARACTER
8692  013420  105763  001436                TSTB    TRO(R3)         ;ALL CHARS DONE?
8693  013424  001002                        BNE     .+6
8694  013426  040237  013512                BIC     R2,RXTCR        ;ZERO LINE DONE INDICATOR.
8695  013432  012716  013062                MOV     #SNAP,(SP)      ;RESET THE BACKGROUND TIMING LOOP
```

```
 8696  013436  000002                          RTI
 8697
 8698
 8699                                          ;FINISH UP ROUTINE
 8700  013440  106427  000200          OUT:    MTPS    #MASK           ;STOP ALL INTERRUPTS
 8701  013444  104413                          DEVICE.CLR              ;CLEAR ALL INTERRUPTS AWAY
 8702  013446  005003                          CLR     R3
 8703  013450  005037  001374                  CLR     SAVLIN
 8704  013454  012702  000001                  MOV     #1,R2
 8705  013460  030237  001366          1$:     BIT     R2,LINE         ;VALID LINE ?
 8706  013464  001405                          BEQ     2$              ;NO
 8707  013466  022763  000400  001436          CMP     #400,TRO(R3)    ;RECEIVED A BINARY COUNT PATTERN ?
 8708  013474  001401                          BEQ     .+4             ;YES
 8709  013476  104030                          ERROR   30              ;THE LINE FAILED TO RECEIVE A FULL
 8710                                                                  ;BINARY COUNT PATTERN
 8711  013500  005237  001374          2$:     INC     SAVLIN          ;SET UP FOR NEXT LINE
 8712  013504  005723                          TST     (R3)+           ;ADD 2
 8713  013506  104420                          SHIFT                   ;SET UP NEXT LINE POINTER
 8714  013510  000763                          BR      1$              ;FINISHED ?
 8715  013512  000000                  RXTCR:  0                       ;RX IMAGE OF TCR BITS
 8716
 8717
 8718                                          ;************************* TEST 4 *********************************
 8719                                          ;*DZV11 RELATIVE TIMING TEST.
 8720                                          ;*EACH SELECTED LINE WILL IN TURN RUN 16. CHARS
 8721                                          ;*AT ALL BAUD RATES AND THEN THE HIGHEST BAUD
 8722                                          ;*WITH ALL CHAR LENGTHS. EACH NEW PARAMETER SHOULD
 8723                                          ;*DECREASE IN TIME FROM THE PREVIOUS PARAMETERS SELECTED.
 8724                                          ;*THE TIME IS CHECKED AGAINST THE LAST PARAMETER USED
 8725                                          ;* AND A LOWER TIME IS EXPECTED ON THE CURRENT PARAMETER.
 8726                                          ;*PARAMETERS ARE:
 8727                                          ;*  EIGHT BITS/PER/CHAR - TWO STOP BITS AT
 8728                                          ;*      50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000
 8729                                          ;*      2400, 3600, 4800, 7200, 9600 BAUD.
 8730                                          ;*  19.2 K BAUD - TWO STOP BITS AT
 8731                                          ;*      SEVEN, SIX, FIVE BITS/PER/CHAR.
 8732                                          ;*AFTER EACH LINE HAS FINISHED ALL THE ABOVE PARAMETERS
 8733                                          ;*THE NEXT SELECTED LINE IS THEN TESTED.
 8734                                          ;*WHEN RUNNING UNDER THE APT MANUFACTURING SYSTEM
 8735                                          ;*THIS TEST IS ONLY RUN THE FIRST PASS
 8737                                  ;:*  TEST 4
  (5)                                  ;;*****************************************************************
  (4)  013514  000004                  TST4:   SCOPE
  (4)
  (2)  013516  012737  000004  001246          MOV     #4,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
  (2)  013524  012737  014132  001362          MOV     #TST5,NEXT      ;POINT TO THE START OF THE NEXT TEST
  (1)  013532  012737  013652  001364          MOV     #3$,LOCK        ;USE THIS ADDRESS IF A TIGHT SCOPE LOOP IS SELECTED
 8738  013540  132737  000001  001140          BITB    #1,$ENV         ;RUNNING UNDER APT?
 8739  013546  001405                          BEQ     10$             ;IF NOT CONTINUE WITH TEST
 8740  013550  005737  001126                  TST     $PASS           ;IF YES IS THIS FIRST PASS
 8741  013554  001402                          BEQ     10$             ;IF NOT 1ST PASS SKIP TEST
 8742  013556  000177  165600                  JMP     @NEXT
 8743  013562  012737  000002  001354  10$:    MOV     #2,$TIMES       ;SET UP FOR 2 ITERATIONS
 8744  013570  005037  015134                  CLR     OFFSET          ;RESET THIS VARIABLE
 8745  013574  005037  001374                  CLR     SAVLIN          ;RESET LINE NUMBER INDICATOR
 8746  013600  005037  001376                  CLR     XMTLIN          ;USE THIS WORD TO TELL WHAT LINE TRANSMITTED
```

```
 8747  013604  012702  000001                MOV     #1,R2                   ;USE R2 AS A BIT POINTER
 8748  013610  012703  010070                MOV     #RCVON!S50!EIGHT!TWOSTOP,R3 ;BUILD TEMPORARY PARAMETERS
 8749  013614  030237  001366          1$:   BIT     R2,LINE                 ;IS THIS LINE ACTIVE?
 8750  013620  001014                        BNE     3$                      ;IF SO, GO GET STARTED
 8751  013622  012703  010070          2$:   MOV     #RCVON!S50!EIGHT!TWOSTOP,R3 ;LOAD PARAMETERS TEMPORARILY
 8752  013626  005237  001376                INC     XMTLIN                  ;POINT TO THE NEXT LINE TO TRANSMIT
 8753  013632  042703  000007                BIC     #7,R3                   ;MAKE SURE TEMPORARY PARAMETERS POINT TO 0
 8754  013636  053703  001376                BIS     XMTLIN,R3               ;ADD DESIRED LINE NUMBER
 8755  013642  005037  015134                CLR     OFFSET
 8756  013646  104420                        SHIFT                           ;POINT TO THE NEXT LINE
 8757  013650  000761                        BR      1$                      ;PROCESS THE NEXT LINE
 8758  013652                          3$:
   (1) 013652  104417                        DCLASM                          ;CLEAR DEVICE AND SET MAINT BIT IF I MODE
 8759  013654  042703  010000                BIC     #RCVON,R3               ;ZERO PARAMTERS FOR TX LINE
 8760  013660  010377  166134                MOV     R3,@DZVLPR              ;LOAD PARAMTERS FOR TX
 8761  013664  005737  001372                TST     MODE                    ;STAGGERED?
 8762  013670  100007                        BPL     100$                    ;BR IF NO
 8763  013672  000241                        CLC                             ;SET UP LINE
 8764  013674  006003                        ROR     R3                      ;
 8765  013676  103002                        BCC     98$                     ;BR IF LINE WAS EVEN
 8766  013700  000241                        CLC                             ;PREPARE TO MKE LINE EVEN
 8767  013702  000401                        BR      99$                     ;CONTINUE
 8768  013704  000261                  98$:  SEC                             ;PREPARE TO MAKE LINE ODD
 8769  013706  006103                  99$:  ROL     R3                      ;SET ALTERED LINE
 8770  013710  052703  010000          100$: BIS     #RCVON,R3               ;SET RX ON
 8771  013714  010377  166100                MOV     R3,@DZVLPR              ;LOAD RX PARAMETERS
 8772  013720  010337  001374                MOV     R3,SAVLIN               ;SET FOR RECEIV. LINE
 8773  013724  042737  177774  001374        BIC     #^C<3>,SAVLIN           ;ISOLATE LINE NO.
 8774  013732  042703  000003                BIC     #3,R3                   ;CLEAR OLD LINE #
 8775  013736  053703  001376                BIS     XMTLIN,R3               ;SET LINE UP AGAIN
 8776  013742  010337  001402                MOV     R3,REGIST               ;SAVE PARAMETERS FOR PRINTOUT
 8777  013746  104422                        BUFSET                          ;ZERO DATA BUFFER
 8778  013750  005037  001342                CLR     $TMP0                   ;USE $TMP0 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
 8779  013754  005037  001344                CLR     $TMP1                   ;INITIALIZE THE TIMER
 8780  013760  005037  001350                CLR     $TMP3                   ;INITIALIZE THESE BITS ALSO
 8781  013764  012737  000020  001400        MOV     #20,XMTCNT              ;SET HOW MANY CHARACTERS TO TRANSMIT
 8782  013772  012777  014562  166044        MOV     #XMTSRV,@DZVTIV
 8783  014000  012777  014732  166032        MOV     #RXISR1,@DZVRIV
 8784  014006  012777  000200  166026        MOV     #MASK,@DZVRIS
 8785  014014  012777  000200  166024        MOV     #MASK,@DZVTIS
 8786  014022  110277  165776                MOVB    R2,@DZVTCR              ;START THE VALID LINE
 8787  014026  052777  040140  165754        BIS     #TIE!RIE!MSENAB,@DZVCSR
 8788  014034  106427  000000                MTPS    #0                      ;LOWER THE PRIORITY TO ALLOW INTERRUPTS
 8789  014040  032777  000100  165742  4$:   BIT     #RIE,@DZVCSR            ;IS ROUTINE DONE?
 8790  014046  001407                        BEQ     5$                      ;WHEN ALL IS DONE RX IE IS CLEARED IN ISR.
 8791  014050  005237  001344                INC     $TMP1                   ;INCREMENT TIMER
 8792  014054  001371                        BNE     4$                      ;WHEN IT OVERFLOWS
 8793  014056  005237  001350                INC     $TMP3                   ;CATCH CARRY
 8794  014062  001366                        BNE     4$                      ;CONTINUE TEST
 8795  014064  104011                        ERROR   11                      ;INTERRUPTS NOT FINISHED
 8796  014066  004737  007046          5$:   JSR     PC,SERV.G               ;<^G>?
 8797  014072  104401                        SCOP1                           ;LOOP?
 8798  014074  062737  000002  015134        ADD     #2,OFFSET
 8799  014102  022703  017400                CMP     #17400,R3
 8800  014106  003006                        BGT     6$
 8801  014110  032703  000030                BIT     #BIT4+BIT3,R3           ;IS  CHARACTER SIZE DONE?
```

```
 8802  014114  001642                          BEQ     2$
 8803  014116  162703  000010                  SUB     #BIT3,R3
 8804  014122  000653                          BR      3$
 8805  014124  062703  000400          6$:     ADD     #400,R3
 8806  014130  000650                          BR      3$
 8807                                          ;************************** 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)  014132  000004                  TST5:   SCOPE
  (5)
  (3)  014134  012737  000005  001246          MOV     #5,$TSTNM       ;LOAD THE NUMBER OF THIS TEST
  (2)  014142  012737  004140  001362          MOV     #$EOP,NEXT      ;POINT TO THE END-OF-PASS HANDLER
  (1)  014150  005737  001372                  TST     MODE            ;IS THIS STAGGERED MODE?
  (1)  014154  100131                          BPL     17$             ;IF NOT, DON'T DO THIS TEST
  (1)  014156  105037  001425                  CLRB    DONFLG          ;SET UP FOR FIRST TEST PASS
  (1)  014162  104413                  14$:    DEVICE.CLR              ;SET DCLR IN CSR
  (1)  014164  013701  001370                  MOV     PAR,R1          ;USE R1 TO BUILD PARAMETERS TO BE LOADED
  (1)  014170  042701  000200                  BIC     #ODDPAR,R1      ;MAKE SURE ODD PARITY ISN'T SET
  (1)  014174  052701  000100                  BIS     #PARITY,R1      ;MAKE SURE PARITY IS TURNED ON
  (1)  014200  012702  000001                  MOV     #1,R2           ;USE R2 AS A LINE POINTER
  (1)  014204  030237  001366          1$:     BIT     R2,LINE         ;IS THIS A VALID LINE?
  (1)  014210  001420                          BEQ     3$              ;IF NOT, SKIP TO THE NEXT LINE
  (1)  014212  105737  001425                  TSTB    DONFLG          ;FIRST PASS THROUGH TEST?
  (1)  014216  001004                          BNE     15$             ;IF NO BRANCH
  (1)  014220  032701  000001                  BIT     #BIT0,R1        ;IS THIS LINE AN ODD LINE?
  (1)  014224  001006                          BNE     2$              ;IF IT'S ODD, USE EVEN PARITY
  (1)  014226  000403                          BR      16$             ;IF EVEN SET FOR ODD PARITY
  (1)  014230  032701  000001          15$:    BIT     #BIT0,R1        ;IF THE LINE  IS EVEN SET FOR EVEN PAR.
  (1)  014234  001402                          BEQ     2$              ;GO LOAD PARAMETER
  (1)  014236  052701  000200          16$:    BIS     #ODDPAR,R1      ;IF IT'S ODD, USE ODD PARITY
  (1)  014242  010177  165552          2$:     MOV     R1,@DZVLPR      ;LOAD THE LINE PARAMETER REGISTER
  (1)  014246  042701  000200                  BIC     #ODDPAR,R1      ;SET UP THE NEXT PARITY TO EVEN
  (1)  014252  005201                  3$:     INC     R1              ;POINT TO THE NEXT LINE
  (1)  014254  006302                          ASL     R2              ;
  (1)  014256  032702  000020                  BIT     #BIT4,R2        ;ALL LINES DONE?
  (1)  014262  001750                          BEQ     1$              ;IF NOT, GO CHECK THE NEXT LINE
  (1)  014264  005037  001374                  CLR     SAVLIN          ;CLEAR THE LINE NUMBER INDICATOR
  (1)  014270  005037  001342                  CLR     $TMP0           ;USE $TMP0 TO COUNT TOTAL NUMBER OF TRANSMISSIONS
  (1)  014274  005003                          CLR     R3              ;USE R3 TO COUNT TOTAL NUMBER OF RECEPTIONS
  (1)  014276  012737  000040  001400          MOV     #40,XMTCNT      ;TRANSMIT A BINARY COUNT PATTERN(00-40)
  (1)  014304  104422                          BUFSET                  ;ZERO BUFFER AREA
  (2)  014306  012777  014562  165530          MOV     #XMTSRV,@DZVTIV ;SET UP THE TRANSMITTER INTERRUPT VECTOR
  (2)  014314  012777  014442  165516          MOV     #9$,@DZVRIV     ;SET UP THE RECEIVER INTERRUPT VECTOR
  (2)  014322  012777  000200  165512          MOV     #MASK,@DZVRIS   ;SET THE INTERRUPT VECTOR STATUS
  (2)  014330  012777  000200  165510          MOV     #MASK,@DZVTIS   ;SET TRANSMITTER INTERRUPT PRIORITY
  (2)  014336  052777  040140  165444          BIS     #RIE!TIE!MSENAB,@DZVCSR ;ENABLE THE DEVICE
  (1)  014344  113777  001366  165452          MOVB    LINE,@DZVTCR    ;ENABLE ALL SELECTED LINES
  (1)  014352  106427  000000                  MTPS    #0              ;ALLOW INTERRUPTS
```

```
(1)  014356  005037  014434          4$:    CLR     7$
(1)  014362  005037  014436                 CLR     8$
(1)  014366  032777  000100  165414  5$:    BIT     #RIE,@DZVCSR    ;WHEN RX DONE; RIE WILL =0
(1)  014374  001407                         BEQ     6$              ;BR IF ALL DONE
(1)  014376  005237  014434                 INC     7$
(1)  014402  001371                         BNE     5$
(1)  014404  105237  014436                 INCB    8$
(1)  014410  100366                         BPL     5$
(1)  014412  104011                         ERROR   11              ;*RX FAILED TO FINISH (INTERRUPT)
(1)  014414  106427  000200          6$:    MTPS    #MASK           ;SHUT OFF INTERRUPTS
(1)  014420  105737  001425                 TSTB    DONFLG          ;IS THIS SECOND TEST PASS
(1)  014424  001005                         BNE     17$             ;IF SO GET OUT
(1)  014426  105237  001425                 INCB    DONFLG          ;INDICATE FIRST TEST PASS DONE
(1)  014432  000653                         BR      14$             ;START OVER
(1)  014434  000000          7$:    0
(1)  014436  000000          8$:    0
(1)  014440  104400          17$:   ADVANCE
(1)
(1)
(1)                                  ;RECEIVER SERVICE ROUTINE
(1)
(1)  014442  017704  165346          9$:    MOV     @DZVRBUF,R4     ;GET THE CHARACTER
(1)  014446  010401                         MOV     R4,R1           ;COPY THE RECEIVED INFORMATION
(1)  014450  000301                         SWAB    R1              ;GET THE LINE NUMBER IN THE LOWER BYTE
(1)  014452  042701  177774                 BIC     #^C<3>,R1       ;ISOLATE THE LINE NUMBER
(1)  014456  010137  001374                 MOV     R1,SAVLIN       ;SET LINE INDIC. TO RECEIVING LINE
(1)  014462  005704                         TST     R4              ;IS DATA VALID SET?
(1)  014464  100401                         BMI     10$             ;IF YES DON'T PRINT ERROR
(1)  014466  104023                         ERROR   23              ;DATA VALID NOT SET
(1)  014470  010105          10$:   MOV     R1,R5           ;BUILD LINE NO. FOR
(1)  014472  000305                         SWAB    R5              ;EXPECTED DATA IN RECEIVER BUFFER
(1)  014474  006301                         ASL     R1              ;ADJUST R1 FOR OFFSET
(1)  014476  156105  001436                 BISB    TRO(R1),R5      ;LOAD CHARACTER IN EXPECTED
(1)  014502  052705  110000                 BIS     #DVALID!PARER,R5 ;BUILD WHAT WAS EXPECTED
(1)  014506  020405                         CMP     R4,R5           ;DOES RECEIVED=EXPECTED
(1)  014510  001401                         BEQ     12$             ;IF YES DON'T PRINT ERROR
(1)  014512  104006                         ERROR   6               ;*ERROR- DID NOT GET CORRECT INFORMATION
(1)  014514  005261  001436          12$:   INC     TRO(R1)         ;SET UP THE NEXT CHARACTER
(1)  014520  005203                         INC     R3              ;ADD TO THE TOTAL RECEIVED COUNT
(1)  014522  032777  040000  165260        BIT     #TIE,@DZVCSR    ;ARE TRANSMISSIONS DONE?
(1)  014530  001011                         BNE     13$             ;IF NO, GO RECEIVE SOME MORE
(1)  014532  023703  001342                 CMP     $TMP0,R3        ;ARE ALL CHARACTERS RECEIVED?
(1)  014536  001006                         BNE     13$             ;IF NO, GO RECEIVE SOME MORE
(1)  014540  042777  000100  165242        BIC     #RIE,@DZVCSR    ;DISABLE RECEIVER INTERRUPTS
(1)  014546  012716  014414                 MOV     #6$,(SP)        ;CRUNCH THE STACK
(1)  014552  000002                         RTI                     ;RETURN AND FINISH
(1)  014554  012716  014356          13$:   MOV     #4$,(SP)        ;CRUNCH THE STACK
(1)  014560  000002                         RTI                     ;GO BACK TO RECEIVER WAIT LOOP
```

```
8809
8810                                    ;TRANSMITTER INTERRUPT SERVICE
8811                                    ;----------------------------
8812
8813  014562  117701  165224          XMTSRV: MOVB    @HDZVCSR,R1     ;GET THE LINE NUMBER.
8814  014566  042701  177774                  BIC     #^C<3>,R1       ;CLEAR JUNK
8815  014572  013705  001374                  MOV     SAVLIN,R5       ;SAVE REC. LINE NO.
8816  014576  010137  001374                  MOV     R1,SAVLIN       ;LOAD TRANS LINE NO FOR ERROR PRINTOUT
8817  014602  006301                          ASL     R1              ;ADJUST R1 FOR OFFSET
8818  014604  023761  001400  001426          CMP     XMTCNT,TDO(R1)  ;HAVE ALL CHAR. BEEN SENT
8819  014612  003414                          BLE     6$              ;IF YES GO CLEAR TCR
8820  014614  005777  165170                  TST     @DZVCSR         ;TRDY SET?
8821  014620  100401                          BMI     2$              ;IF YES GO LOAD CHAR.
8822  014622  104003                          ERROR   3               ;*TRANSMITTER NOT READY- FALSE INTERRUPT
8823  014624  116177  001426  165202  2$:     MOVB    TDO(R1),@DZVTDR ;LOAD THE CURRENT CHARACTER FOR THIS LINE
8824  014632  005261  001426                  INC     TDO(R1)         ;SET UP NEXT CHARACTER FOR THIS LINE
8825  014636  005237  001342                  INC     $TMPO           ;UP THE NUMBER OF TRANSMISSIONS
8826  014642  000415                          BR      7$              ;GO RETURN
8827  014644  012700  000001          6$:     MOV     #1,R0           ;SET UP A DESELECTION POINTER
8828  014650  006201                          ASR     R1              ;GET LINE NO. AGAIN
8829  014652  005301                  12$:    DEC     R1              ;REDUCE THE COUNT. WAS THIS THE LINE?
8830  014654  100402                          BMI     3$              ;IF SO, GO DISABLE THE ENABLE BIT FOR IT
8831  014656  006300                          ASL     R0              ;MOVE THE POINTER TO THE NEXT LINE
8832  014660  000774                          BR      12$             ;GO CHECK THE NEXT LINE
8833  014662  140077  165136          3$:     BICB    R0,@DZVTCR      ;DISABLE THE LINE POINTED TO BY R0
8834  014666  001003                          BNE     7$              ;IF MORE LINES ARE ACTIVE , GO CONTINUE TRANSMIT
8835  014670  042777  040000  165112          BIC     #TIE,@DZVCSR    ;IF NOT, DISABLE TRANSMITTER INTERRUPTS
8836  014676  010537  001374          7$:     MOV     R5,SAVLIN       ;RESTORE RECEIV. LINE
8837  014702  000002                          RTI                     ;RETURN TO THE TIMING LOOP
8838
8839                                    ; RELATIVE TIME BUILDING ROUTINE
8840                                    ; -----------------------------
8841
8842  014704  012737  000002  001346  BUILD:  MOV     #2,$TMP2        ;ROTATE 2 BITS BACK INTO $TMP1
8843  014712  006037  001350          1$:     ROR     $TMP3           ;GET THE BITS FROM $TMP3, THE HIGH BYTE
8844  014716  006037  001344                  ROR     $TMP1           ;OF THE RELATIVE TIME COUNTER. PUT THEM BACK
8845  014722  005337  001346                  DEC     $TMP2           ;INTO $TMP1 USING THE CARRY BIT WITH
8846                                                                  ;ROTATE INSTRUCTIONS
8847  014726  001371                          BNE     1$              ;REDUCE COUNT. ALL BITS BACK? IF NOT, GET MORE
8848  014730  000207                          RTS     PC              ;RETURN TO CALLING TEST
8849
```

```
                                        ;RECEIVER SERVICE ROUTINE
                                
014732  105777  165052      RXISR1: TSTB    @DZVCSR         ;IS THE RECEIVER REALLY READY?
014736  100401                  BMI     1$              ;IF SO, GO SERVICE IT
014740  104004                  ERROR   4               ;*ERROR- RECEIVER DONE FLAG ISN'T SET
014742  017704  165046      1$:     MOV     @DZVRBUF,R4     ;SAVE THE RECEIVER INFORMATION
014746  100401                  BMI     2$              ;IF IT WAS VALID, GO PROCESS IT
014750  104023                  ERROR   23              ;ERROR- DATA VALID WASN'T SET
014752  032704  040000      2$:     BIT     #OVRRUN,R4      ;OVERRUN ERROR FLAG SET?
014756  001401                  BEQ     6$              ;IF NOT DON'T TYPE ERROR
014760  104024                  ERROR   24              ;OVERRUN ERROR
014762  032704  020000      6$:     BIT     #FRMERR,R4      ;FRAMING ERROR FLAG SET?
014766  001401                  BEQ     9$              ;IF NOT DON'T TYPE ERROR
014770  104025                  ERROR   25              ;FRAMING ERROR
014772  032704  010000      9$:     BIT     #PARER,R4       ;PARITY ERROR FLAG SET?
014776  001401                  BEQ     3$              ;IF NOT, GO CONTINUE PROCESSING
015000  104026                  ERROR   26              ;ERROR- RECEIVER ERROR FLAG SET
015002  013701  001374      3$:     MOV     SAVLIN,R1       ;CALCULATE THE DATA OFFSET
015006  006301                  ASL     R1              ;ALIGN IT ON A WORD BOUNDARY
015010  120461  001436          CMPB    R4,TRO(R1)      ;IS THE CHARACTER WHAT IT SHOULD BE?
015014  001407                  BEQ     4$              ;IF SO,GO CONTINUE PROCESSING
015016  116105  001436          MOVB    TRO(R1),R5      ;GET WHAT WAS EXPECTED FOR ERROR REPORTING
015022  042705  177400          BIC     #^C<377>,R5     ;ELIMINATE PROPAGATED SIGN
015026  042704  177400          BIC     #^C<377>,R4     ;ISOLATE THE ACTUAL CHARACTER
015032  104005                  ERROR   5               ;*DATA ERROR
015034  005261  001436      4$:     INC     TRO(R1)         ;SET UP THE NEXT EXPECTED CHARACTER
015040  122761  000020  001436      CMPB    #20,TRO(R1)     ;HAVE ALL CHARACTERS BEEN RECEIVED?
015046  001037                  BNE     8$              ;IF NOT RETURN
015050  126137  001436  001342      CMPB    TRO(R1),$TMP0   ;ALL CHARAC. RECEIVED?
015056  001025                  BNE     8$              ;IF SO,GO DETERMINE THE TIMING
015060  004737  014704          JSR     PC,BUILD        ;GET THE RELATIVE TIME (SIGNIFICANT BITS)
015064  013700  015134          MOV     OFFSET,R0       ;GET POINTER
015070  013760  001344  002050      MOV     $TMP1,TMTBL(R0) ; SAVE THIS TEST'S TIME
015076  005737  015134          TST     OFFSET          ;FIRST TEST?
015102  001410                  BEQ     7$              ;IF NOT, GO CHECK THE TIME
015104  005740                  TST     -(R0)           ;POINT TO THE PREVIOUS  TIME TAKEN
015106  026037  002050  001344      CMP     TMTBL(R0),$TMP1 ;IS THIS TIME WHAT IT SHOULD BE?
015114  101003                  BHI     7$              ;IF SO, GO TO THE NEXT TEST
015116  016005  002050          MOV     TMTBL(R0),R5    ;PLACE WHAT WAS EXPECTED IN R5
015122  104021                  ERROR   21              ;TIMING ERROR
015124  042777  000140  164656  7$:     BIC     #RIE!MSENAB,@DZVCSR ;DISABLE THE DEVICE
015132  000002              8$:     RTI                     ;RETURN TO THE PROGRAM
015134  000000          OFFSET: 0
```

```
                                        ;ERROR TABLE
015136  000000          .ERRTAB:        0       ;ERROR 0
015140  000000                          0
015142  000000                          0

015144  015364                          EM1     ;ERROR
015146  016510                          DH1
015150  016710                          DT1

015152  015437                          EM2     ;ERROR 2
015154  016534                          DH2
015156  016722                          DT2

015160  015465                          EM3     ;ERROR 3
015162  016567                          DH3
015164  016740                          DT3

015166  015524                          EM4     ;ERROR 4
015170  016567                          DH3
015172  016740                          DT3

015174  015553                          EM5     ;ERROR 5
015176  016601                          DH4
015200  016746                          DT4

015202  015602                          EM6     ;ERROR 6
015204  016601                          DH4
015206  016746                          DT4

015210  000000                          0
015212  000000                          0
015214  000000                          0

015216  000000                          0
015220  000000                          0
015222  000000                          0

015224  015641                          EM11    ;ERROR 11
015226  016567                          DH3
015230  016740                          DT3

015232  000000                          0
015234  000000                          0
015236  000000                          0

015240  015677                          EM13    ;ERROR 13
015242  016567                          DH3
015244  016740                          DT3

015246  015730                          EM14    ;ERROR 14
015250  016567                          DH3
015252  016740                          DT3

015254  015762                          EM15    ;ERROR 15
015256  000000                          0
015260  000000                          0
```

```
 8951
 8952  015262  016024                                  EM16
 8953  015264  016567                                  DH3
 8954  015266  016740                                  DT3
 8955
 8956  015270  016076                                  EM17    ;ERROR 17
 8957  015272  016567                                  DH3
 8958  015274  016740                                  DT3
 8959
 8960  015276  016134                                  EM20
 8961  015300  016567                                  DH3
 8962  015302  016740                                  DT3
 8963
 8964  015304  016175                                  EM21    ;ERROR 21
 8965  015306  016630                                  DH5
 8966  015310  016764                                  DT5
 8967
 8968  015312  000000                                  0
 8969  015314  000000                                  0
 8970  015316  000000                                  0
 8971
 8972  015320  016225                                  EM23    ;ERROR 23
 8973  015322  016567                                  DH3
 8974  015324  016740                                  DT3
 8975
 8976  015326  016255                                  EM24
 8977  015330  016567                                  DH3
 8978  015332  016740                                  DT3
 8979
 8980  015334  016303                                  EM25
 8981  015336  016567                                  DH3
 8982  015340  016740                                  DT3
 8983
 8984  015342  016333                                  EM26
 8985  015344  016567                                  DH3
 8986  015346  016740                                  DT3
 8987
 8988  015350  016362                                  EM27
 8989  015352  016567                                  DH3
 8990  015354  016740                                  DT3
 8991
 8992  015356  016430                                  EM30
 8993  015360  016567                                  DH3
 8994  015362  016740                                  DT3
```

```
 8996                                           ;ERROR MESSAGES
 9000  015364  047200  020117  052502  EM1:     .ASCIZ  <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
 9001  015437     200  042522  044507  EM2:     .ASCIZ  <200>?REGISTER R/W FAILURE?
 9002  015465     200  051124  047101  EM3:     .ASCIZ  <200>/TRANSMIT READY (TRDY) NOT SET/
 9003  015524  051200  041505  044505  EM4:     .ASCIZ  <200>/RECEIVER DONE NOT SET/
 9004  015553     200  040504  040524  EM5:     .ASCIZ  <200>/DATA COMPARISON ERROR/
 9005  015602  042200  053132  030461  EM6:     .ASCIZ  <200>/DZV11 *RECEIVER BUFFER* ERROR/
 9006  015641     200  042522  042503  EM11:    .ASCIZ  <200>/RECEIVER FAILED TO INTERRUPT/
 9007  015677     200  044523  047514  EM13:    .ASCIZ  <200>/SILO ALARM SET TOO SOON/
 9008  015730  051600  046111  020117  EM14:    .ASCIZ  <200>/SILO ALARM FAILED TO SET/
 9009  015762  040600  052103  047511  EM15:    .ASCIZ  <200>/ACTION DETECTED ON INVALID LINE./
 9010  016024  051200  040505  044504  EM16:    .ASCIZ  <200>/READING DZVRBUF DID NOT CLEAR SILO ALARM/
 9011  016076  042200  052101  020101  EM17:    .ASCIZ  <200>/DATA VALID SHOULD NOT BE SET/
 9012  016134  051200  041505  044505  EM20:    .ASCIZ  <200>/RECEIVER DONE SHOULD NOT BE SET/
 9013  016175     200  042522  040514  EM21:    .ASCIZ  <200>/RELATIVE TIMING ERROR./
 9014  016225     200  040504  040524  EM23:    .ASCIZ  <200>/DATA VALID IS NOT SET!/
 9015  016255     200  040504  040524  EM24:    .ASCIZ  <200>/DATA OVERRUN IS SET!/
 9016  016303     200  051106  046501  EM25:    .ASCIZ  <200>/FRAMING ERROR OCCURRED/
 9017  016333     200  040520  044522  EM26:    .ASCIZ  <200>/PARITY ERROR OCCURRED/
 9018  016362  051600  046111  020117  EM27:    .ASCIZ  <200>/SILO ALARM FAILED TO CAUSE INTERRUPT/
 9019  016430  046200  047111  020105  EM30:    .ASCIZ  <200>/LINE DID NOT RECEIVE FULL BINARY COUNT PATTERN/
 9020
 9021  016510  052200  040522  020120  DH1:     .ASCIZ  <200>/TRAP PC  DZV11 REG/
 9022  016534  042600  050130  041505  DH2:     .ASCIZ  <200>/EXPECTED  FOUND  REGISTER/
 9023  016567     200  044514  042516  DH3:     .ASCIZ  <200>/LINE NO./
 9024  016601     200  054105  042520  DH4:     .ASCIZ  <200>/EXPECTED  FOUND  LINE/
 9025  016630  052200  020130  044514  DH5:     .ASCIZ  <200>/TX LINE PREVIOUS TIME  ACTUAL TIME  PARAMETER/
 9026
 9027          016710                  .EVEN
 9031                                           ;DATA TABLES FOR ERROR MESSAGES
 9032  016710  000002                  DT1:     2
 9033  016712     006  003                      .BYTE   6,3
 9034  016714  001330                           $REG1
 9035  016716     006  001                      .BYTE   6,1
 9036  016720  001326                           $REG0
 9037
 9038  016722  000003                  DT2:     3
 9039  016724     006  004                      .BYTE   6,4
 9040  016726  001340                           $REG5
 9041  016730     006  001                      .BYTE   6,1
 9042  016732  001336                           $REG4
 9043  016734     006  001                      .BYTE   6,1
 9044  016736  001326                           $REG0
 9045
 9046  016740  000001                  DT3:     1
 9047  016742     003  001                      .BYTE   3,1
 9048  016744  001374                           SAVLIN
 9049
 9050  016746  000003                  DT4:     3
 9051  016750     006  004                      .BYTE   6,4
 9052  016752  001340                           $REG5
 9053  016754     006  001                      .BYTE   6,1
 9054  016756  001336                           $REG4
 9055  016760     003  001                      .BYTE   3,1
 9056  016762  001374                           SAVLIN
 9057
```

```
 9058  016764  000004                DT5:    4
 9059  016766     003      005               .BYTE   3,5
 9060  016770  001374                       SAVLIN
 9061  016772     006      011               .BYTE   6,9.
 9062  016774  001340                       $REG5
 9063  016776     006      007               .BYTE   6,7
 9064  017000  001344                       $TMP1
 9065  017002     006      001               .BYTE   6,1
 9066  017004  001402                       REGIST
 9073                                        ;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES
 9074                                        ;---------------------------------------------
 9075
 9076  017006  002450                DLYTBL: 2450                    ;TIME FOR   50 BAUD
 9077  017010  001560                        1560                    ;TIME FOR   75 BAUD
 9078  017012  001120                        1120                    ;TIME FOR  110 BAUD
 9079  017014  000750                         750                    ;TIME FOR  134 BAUD
 9080  017016  000660                         660                    ;TIME FOR  150 BAUD
 9081  017020  000330                         330                    ;TIME FOR  300 BAUD
 9082  017022  000150                         150                    ;TIME FOR  600 BAUD
 9083  017024  000060                          60                    ;TIME FOR 1200 BAUD
 9084  017026  000040                          40                    ;TIME FOR 1800 BAUD
 9085  017030  000030                          30                    ;TIME FOR 2000 BAUD
 9086  017032  000020                          20                    ;TIME FOR 2400 BAUD
 9087  017034  000010                          10                    ;TIME FOR 3600 BAUD
 9088  017036  000001                           1                    ;TIME FOR 4800 BAUD
 9089  017040  000001                           1                    ;TIME FOR 7200 BAUD
 9090  017042  000001                           1                    ;TIME FOR 9600 BAUD
 9091  017044  000001                           1                    ;TIME OF DELAY FOR 19200 BAUD
 9092
 9093                                        ;DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
 9094                                        ;FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.
```

```
                                        ;; INIT $BASE AND $VECT1 AND TWEAK THE "$GETPAR" CALLING
                                        ;; SEQUENCE TO ACCEPT THE VALID FALCON RANGE.
                                        ;;
017046  023727  001174  160010  FALCINI: CMP     $BASE,#ABASE     ; IS $BASE VIRGIN ??              ;;GPA
017054  001003                           BNE     1$               ; SKIP NEXT IF NOT                ;;GPA
017056  012737  174040  001174           MOV     #174040,$BASE    ; YES, SET ENGINEERING DEFAULT    ;;GPA
017064  023727  001170  000300  1$:      CMP     $VECT1,#AVECT1   ; IS $VECT1 VIRGIN ??            ;;GPA
017072  001003                           BNE     2$               ; SKIP NEXT IF NOT                ;;GPA
017074  012737  000370  001170           MOV     #370,$VECT1      ; YES, SET ENGINEERING DEFAULT    ;;GPA
017102  012737  017146  002354  2$:      MOV     #3$,GETCSR+2     ; SUBSTITUE CSR TEXT...           ;;GPA
017110  012737  174000  002360           MOV     #174000,GETCSR+6                                   ;;GPA
017116  012737  177770  002362           MOV     #177770,GETCSR+10 ;...AND VALID RANGE.             ;;GPA
017124  012737  017210  002400           MOV     #4$,GETVEC+2     ; SUBSTITUTE VECTOR TEXT...       ;;GPA
017132  005037  002404                   CLR     GETVEC+6                                           ;;GPA
017136  012737  000370  002406           MOV     #370,GETVEC+10   ;...AND VALID RANGE.              ;;GPA
017144  000207                           RETURN                   ; RETURN TO CALLER.               ;;GPA

017146  030600  052123  041440  3$:      .ASCIZ  <200>'1ST CSR ADDRESS (174000:177770) '            ;;GPA
017154  051123  040440  042104
017162  042522  051523  024040
017170  033461  030064  030060
017176  030472  033467  033467
017204  024460  000040
017210  030600  052123  053040  4$:      .ASCIZ  <200>'1ST VECTOR ADDRESS (000:370)    '            ;;GPA
017216  041505  047524  020122
017224  042101  051104  051505
017232  020123  030050  030060
017240  031472  030067  020051
017246  020040  000040

                                        .EVEN                                                       ;;GPA

                                        ;$FREE= <1000-.>/2               ; FREE WORDS LEFT.         ;;GPA
                                        ;       .IF LT $FREE                                        ;;GPA
                                        ;       .ERROR ; VECTOR OVERLAY SPACE EXCEEDED              ;;GPA
                                        ;       .ENDC                                               ;;GPA
                                        ;
                                        ;       .=$SVPC                                             ;;GPA
017252                                  CORMAX:
                                        ;THE FOLLOWING CALL TO CNMAC2.SML ADDED TO INIT FALCON
                                        ;SPECIFICS.
        017252                          POINT=.                  ;SAVE POINTER
(1)     000100                          .=100
(1)  000100  017252                     $CLKVEC                  ;LKVEC  HANDLER
(1)  000102  000300                     300                      ;INTERRUPT HANDLER PRI
(1)     000140                          .=140                    ;BRKVEC
(1)  000140  170000                     170000                   ;ODT START ADDRESS
(1)  000142  000300                     300                      ;PRIORITY
(1)     017252                          .=POINT                  ;RESTORE POINTER
(1)  017252  104402  017260             $CLKVEC:        TYPE,CLKMES
(1)  017256  000000                                     HALT
(1)  017260  005015  045514  042526  CLKMES:  .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
(1)  017266  020103  047111  042524
(1)  017274  051122  050125  020124
(1)  017302  020055  044504  041523
(1)  017310  047117  042516  052103
(1)  017316  046040  041524  000040
```

```
CNDZB-A MACY11 30(1046)  15-DEC-82  14:38  PAGE 85-1
CNDZBA.P11    15-DEC-82 14:35          DZV11 DEVICE DIAGNOSTICS.    COPYRIGHT 1982  DIGITAL EQUIP. CORP.                SEQ 0082

  9206  017324  000001                KXTFLAG:1
  9207          000001                .END
```

```
ABASE = 160010           8294#   8403    9175
ACDW1 = 000017           8299#   8403
ACDW2 = 000000           8403
ACPUOP= 000000           8403
ACTIVE  001420           8403#*  8405*
ADDW0 = 017470           8298#   8403
ADDW1 = 017470           8298#   8403
ADDW10= 017470           8298#   8403
ADDW11= 017470           8298#   8403
ADDW12= 017470           8298#   8403
ADDW13= 017470           8298#   8403
ADDW14= 017470           8298#   8403
ADDW15= 017470           8298#   8403
ADDW2 = 017470           8298#   8403
ADDW3 = 017470           8298#   8403
ADDW4 = 017470           8298#   8403
ADDW5 = 017470           8298#   8403
ADDW6 = 017470           8298#   8403
ADDW7 = 017470           8298#   8403
ADDW8 = 017470           8298#   8403
ADDW9 = 017470           8298#   8403
ADEVCT= 000000           8403
ADEVM = 000001           8300#   8403
ADRCNT  005735           8405#*
ADVANC= 104400           8403#   8405    8618    8807
AENV  = 000000           8403
AENVM = 000000           8403
AFATAL= 000000           8403
AMADR1= 000000           8403
AMADR2= 000000           8403
AMADR3= 000000           8403
AMADR4= 000000           8403
AMAMS1= 000000           8403
AMAMS2= 000000           8403
AMAMS3= 000000           8403
AMAMS4= 000000           8403
AMSGAD= 000000           8403
AMSGLG= 000000           8403
AMSGTY= 000000           8403
AMTYP1= 000000           8403
AMTYP2= 000000           8403
AMTYP3= 000000           8403
AMTYP4= 000000           8403
APASS = 000000           8403
APRIOR= 000000           8403
APTCSU= 000040           8405#
APTENV= 000001           8405#
APTSIZ= 000200           8405#
APTSPO= 000100           8405#
ASWREG= 000000           8403
ATESTN= 000000           8403
AUNIT = 000000           8403
AUSWR = 000000           8403
AUTO.S  011304           8403    8405#
AVECT1= 000300           8295#   8403    9178
AVECT2= 000000           8403
```

```
BINWRD  006210           8405#
BIT0   = 000001          8403#   8807
BIT00  = 000001          8403#
BIT01  = 000002          8403#
BIT02  = 000004          8403#
BIT03  = 000010          8403#
BIT04  = 000020          8403#
BIT05  = 000040          8403#
BIT06  = 000100          8403#
BIT07  = 000200          8403#
BIT08  = 000400          8403#
BIT09  = 001000          8403#
BIT1   = 000002          8403#
BIT10  = 002000          8403#
BIT11  = 004000          8403#   8405
BIT12  = 010000          8403#
BIT13  = 020000          8403#
BIT14  = 040000          8403#   8405
BIT15  = 100000          8403#
BIT2   = 000004          8403#
BIT3   = 000010          8403#   8801    8803
BIT4   = 000020          8403#   8405    8801    8807
BIT5   = 000040          8403#   8405
BIT6   = 000100          8403#
BIT7   = 000200          8403#   8405
BIT8   = 000400          8403#
BIT9   = 001000          8403#
BPTVEC= 000014           8403#
BRKVEC= 000140           8403#
BRK0   = 000400          8403#
BRK1   = 001000          8403#
BRK2   = 002000          8403#
BRK3   = 004000          8403#
BRW     004614           8403    8405#
BUFSET= 104422           8403#   8428    8531    8599    8777    8807
BUILD   014704           8842#   8881
CHRCNT  006206           8405#*
CLEAR  = 000000          8403#   8606*
CLKMES  017260           9205#
CNVRT  = 104412          8403#   8405
CONVRT= 104411           8403#   8405
CORMAX  017252           9199#   9200
CO0    = 000400          8403#
CO1    = 001000          8403#
CO2    = 002000          8403#
CO3    = 004000          8403#
CR     = 000015          8403#   8405
CRLF   = 000200          8403#   8405
CSRMAP  011312           8405#
CYCLE   010450           8403    8405#
DATABP  006702           8405#*
DATAHD  006670           8405#*
DCLASM= 104417           8403#   8425    8528    8595    8758
DCLR   = 000020          8403#   8405
DDISP  = 177570          8403#
DELAY  = 104414          8403#   8443    8459    8469    8548    8568    8610
```

```
DEVADR  005732            8405#*
DEVICE= 104413            8403#    8405     8701     8807
DH1     016510            8901     9021#
DH2     016534            8905     9022#
DH3     016567            8909     8913     8933     8941     8945     8953     8957     8961     8973     8977     8981     8985     8989
                          8993     9023#
DH4     016601            8917     8921     9024#
DH5     016630            8965     9025#
DISPLA  001306            8403#*   8405*
DISPRE  000174            8403#
DLYCNT  006304            8403*    8405#
DLYTBL  017006            8403     9076#
DONFLG  001425            8403#    8807*
DSWR   = 177570           8403#
DTR0   = 000400           8403#
DTR1   = 001000           8403#
DTR2   = 002000           8403#
DTR3   = 004000           8403#
DT1     016710            8902     9032#
DT2     016722            8906     9038#
DT3     016740            8910     8914     8934     8942     8946     8954     8958     8962     8974     8978     8982     8986     8990
                          8994     9046#
DT4     016746            8918     8922     9050#
DT5     016764            8966     9058#
DVALID= 100000            8403#    8476     8688     8807
DZCR0   001500            8403#    8405
DZCR1   001512            8403#
DZCR10  001620            8403#
DZCR11  001632            8403#
DZCR12  001644            8403#
DZCR13  001656            8403#
DZCR14  001670            8403#
DZCR15  001702            8403#
DZCR16  001714            8403#
DZCR17  001726            8403#
DZCR2   001524            8403#
DZCR3   001536            8403#
DZCR4   001550            8403#
DZCR5   001562            8403#
DZCR6   001574            8403#
DZCR7   001606            8403#
DZVACT  001406            8403#*   8405
DZVCSR  002010            8403#    8405*    8430*    8436     8441     8451     8457     8481     8505*    8509*    8513*    8535*    8546
                          8556     8562     8566     8604*    8626     8651     8654     8787*    8789     8807*    8820     8835*    8853
                          8891*
DZVC0   001502            8403#    8405
DZVC1   001514            8403#
DZVC10  001622            8403#
DZVC11  001634            8403#
DZVC12  001646            8403#
DZVC13  001660            8403#
DZVC14  001672            8403#
DZVC15  001704            8403#
DZVC16  001716            8403#
DZVC17  001730            8403#
DZVC2   001526            8403#
```

```
DZVC3   001540          8403#
DZVC4   001552          8403#
DZVC5   001564          8403#
DZVC6   001576          8403#
DZVC7   001610          8403#
DZVLEV  011014          8405#
DZVLPR  002020          8403#   8405*   8760*   8771*   8807*
DZVMSR  002030          8403#   8405*
DZVNUM  001414          8403#*  8405*
DZVRBU  002014          8403#   8405*   8477    8491    8539    8657    8807    8856
DZVRIS  002042          8403#   8405*   8534*   8601*   8784*   8807*
DZVRIV  002040          8403#   8405*   8533*   8561*   8600*   8783*   8807*
DZVTCR  002024          8403#   8405*   8435*   8495*   8508*   8544*   8582*   8605*   8639*   8786*   8807*   8833*
DZVTDR  002034          8403#   8405*   8447*   8516*   8552*   8564*   8642*   8823*
DZVTIS  002046          8403#   8405*   8603*   8785*   8807*
DZVTIV  002044          8403#   8405*   8506*   8602*   8782*   8807*
DZV.EN  001740          8403#   8405
DZV.MA  001500          8403#   8405
EIGHT =  000030         8403#   8748    8751
EIGHTS=  000070         8403#
EMTVEC=  000030         8403#
EM1     015364          8900    9000#
EM11    015641          8932    9006#
EM13    015677          8940    9007#
EM14    015730          8944    9008#
EM15    015762          8948    9009#
EM16    016024          8952    9010#
EM17    016076          8956    9011#
EM2     015437          8904    9001#
EM20    016134          8960    9012#
EM21    016175          8964    9013#
EM23    016225          8972    9014#
EM24    016255          8976    9015#
EM25    016303          8980    9016#
EM26    016333          8984    9017#
EM27    016362          8988    9018#
EM3     015465          8908    9002#
EM30    016430          8992    9019#
EM4     015524          8912    9003#
EM5     015553          8916    9004#
EM6     015602          8920    9005#
ERRMSG  006656          8405#*
ERRVEC=  000004         8403#   8405*
ERTABO  007032          8405#
EVEPAR=  000000         8403#
EXITER  006762          8405#
FALCIN  017046          8403*   9175#
FIVE  =  000000         8403#
FIVES =  000040         8403#
FRMERR=  020000         8403#   8668    8862
GETCSR=  002352         8403#   9181*   9182*   9183*
GETSWR=  007074         8403*   8405#
GETVEC=  002376         8403#   9184*   9185*   9186*
HALTS   006706          8405#
HDRFLG  001423          8403#*
HDZVCS  002012          8403#   8405*   8622    8813
```

```
HDZVLP  002022            8403#   8405*
HDZVMS  002032            8403#   8405*
HDZVRB  002016            8403#   8405*
HDZVTC  002026            8403#   8405*
HDZVTD  002036            8403#   8405*
HILIM   005730            8405#*
HT    = 000011            8403#   8405
INBUF   010344            8405#*
INIFLG  001422            8403#*  8405*
INSTER= 104404            8403#   8405
INSTR = 104403            8403#   8405
INSTR2  005530            8405#
IOTVEC= 000020            8403#   8405*
KXTFLA  017324            8403    9206#
LF    = 000012            8403#   8405
LIMITS  005656            8405#
LINE    001366            8403#   8405*   8431    8537    8596    8597    8605    8634    8679    8705    8749    8807
LINE0   001504            8403#*
LINE1   001516            8403#
LINE10  001624            8403#
LINE11  001636            8403#
LINE12  001650            8403#
LINE13  001662            8403#
LINE14  001674            8403#
LINE15  001706            8403#
LINE16  001720            8403#
LINE17  001732            8403#
LINE2   001530            8403#
LINE3   001542            8403#
LINE4   001554            8403#
LINE5   001566            8403#
LINE6   001600            8403#
LINE7   001612            8403#
LKVEC = 000100            8403#
LOBITS  005734            8405#*
LOCK    001364            8403#   8405*   8424*   8527*   8737*
LOLIM   005726            8405#*
LPRSET= 104421            8403#   8426    8529    8598
LP0   = 000000            8403#
LP1   = 000001            8403#
LP2   = 000002            8403#
LP3   = 000003            8403#
MAINT = 000010            8403#   8405
MANT0   001510            8403#   8405
MANT1   001522            8403#
MANT10  001630            8403#
MANT11  001642            8403#
MANT12  001654            8403#
MANT13  001666            8403#
MANT14  001700            8403#
MANT15  001712            8403#
MANT16  001724            8403#
MANT17  001736            8403#
MANT2   001534            8403#
MANT3   001546            8403#
MANT4   001560            8403#
```

```
MANT5   001572           8403#
MANT6   001604           8403#
MANT7   001616           8403#
MASK  = 000200           8403#*  8405    8534    8601    8603    8700*   8784    8785    8807*
MASTEK  007761           8405#
MBADLN  010074           8403    8405#
MCSRX   007711           8405#
MDATA   010406           8405#
MEPASS  007527           8405#
MERRPC  010041           8405#
MERRX   007736           8405#
MERR2   007567           8405#
MERR3   007636           8403    8405#
MLOCK   007662           8403    8405#
MNEW    007764           8403    8405#
MNTFLG  001424           8403#   8405*
MODE    001372           8403#   8405*   8474    8761    8807
MPASSX  007725           8405#
MPFAIL  007464           8405#
MR      007553           8403    8405#
MSENAB= 000040           8403#   8430    8505    8513    8535    8604    8787    8807    8891
MTITLE  001000           8403#
MTSTN   007747           8405#
MVECX   007717           8405#
NEXT    001362           8403#   8405    8423*   8526*   8594*   8737*   8742    8807*
NOLIST= ****** U         8398
ODDPAR= 000200           8403#   8807
ODTST = 170000           8403#
OFFSET  015134           8744*   8755*   8798*   8882    8884    8893#
ONESTO= 000000           8403#
OUT     013440           8613    8700#
OVRRUN= 040000           8403#   8488    8665    8859
PAR     001370           8403#   8405*   8807
PARAM = 104405           8403#   8405
PARAM1  005576           8405#
PARER = 010000           8403#   8671    8807    8865
PARERR  005652           8405#
PARITY= 000100           8403#   8807
PARMD = 104415           8403#
PAR0    001506           8403#*
PAR1    001520           8403#
PAR10   001626           8403#
PAR11   001640           8403#
PAR12   001652           8403#
PAR13   001664           8403#
PAR14   001676           8403#
PAR15   001710           8403#
PAR16   001722           8403#
PAR17   001734           8403#
PAR2    001532           8403#
PAR3    001544           8403#
PAR4    001556           8403#
PAR5    001570           8403#
PAR6    001602           8403#
PAR7    001614           8403#
PAWCH = 104416           8403#
```

```
PIRQ   = 177772          8403#
PIRQVE= 000240           8403#
POINT  = 017252          9205#
POPRO  = 012600          8403#
POP1SP= 005726           8403#
POP2SP= 022626           8403#   8405    8578    8581
PR0    = 000000          8403#
PR1    = 000040          8403#
PR2    = 000100          8403#
PR3    = 000140          8403#
PR4    = 000200          8403#
PR5    = 000240          8403#
PR6    = 000300          8403#   8405
PR7    = 000340          8403#
PS     = 177776          8403#
PSW    = 177776          8403#
PUSHR0= 010046           8403#
PUSH1S= 005746           8403#
PUSH2S= 024646           8403#
PWRVEC= 000024           8403#   8405*
RCVON  = 010000          8403#   8748    8751    8759    8770
RDONE  = 000200          8403#
REGIST  001402           8403#   8776*   9066
RESREG  006704           8405#
RESTAR  011010           8405#
RESVEC= 000010           8403#
RES05  = 104410          8403#   8405
RIE    = 000100          8403#   8535    8604    8787    8789    8807    8891
RING0  = 000001          8403#
RING1  = 000002          8403#
RING2  = 000004          8403#
RING3  = 000010          8403#
RL0    = 000000          8403#
RL1    = 000400          8403#
RL2    = 001000          8403#
RL3    = 001400          8403#
RUN     001412           8403#*  8405*
RXISR1  014732           8783    8853#
RXSVC   013234           8600    8651#
RXTCR   013512           8596*   8611    8694*   8715#
SAVACT  001410           8403#*  8405*
SAVLIN  001374           8403#   8427*   8433    8473    8497*   8530*   8542    8584*   8625*   8661*   8682    8686    8703*
                         8711*   8745*   8772*   8773*   8807*   8815    8816*   8836*   8868    9048    9056    9060
SAVNO   001416           8403#*  8405
SAVNUM  001415           8403#*  8405*
SAVPC   001404           8403#   8405*
SAV05  = 104407          8403#   8405
SCOP1  = 104401          8403#   8496    8514    8583    8797
SERV.G  007046           8405#   8796
SETAPT  011156           8403*   8405#
SETFLG= 104406           8403#
SEVEN  = 000020          8403#
SEVENS= 000060           8403#
SHIFT  = 104420          8403#   8498    8585    8713    8756
SILOAL= 020000           8403#   8451    8457    8481    8556    8566    8654
SILOEN= 010000           8403#   8430    8505    8513    8535
```

```
SIX    = 000010            8403#
SIXS   = 000050            8403#
SNAP     013062            8609#   8695
SPACNT   006207            8405#
STACK  = 001120            8403#   8405
STKLMT= 177774             8403#
STOP     001446            8403#   8405
SV05     005744            8405#
SWR      001304            8403#*  8405*
SWREG    000176            8403#
SW0    = 000001            8403#
SW00   = 000001            8403#
SW01   = 000002            8403#   8405
SW02   = 000004            8403#
SW03   = 000010            8403#
SW04   = 000020            8403#
SW05   = 000040            8403#
SW06   = 000100            8403#
SW07   = 000200            8403#
SW08   = 000400            8403#   8405
SW09   = 001000            8403#   8405
SW1    = 000002            8403#
SW10   = 002000            8403#   8405
SW11   = 004000            8403#
SW12   = 010000            8403#   8405
SW13   = 020000            8403#   8405
SW14   = 040000            8403#
SW15   = 100000            8403#
SW2    = 000004            8403#
SW3    = 000010            8403#
SW4    = 000020            8403#
SW5    = 000040            8403#
SW6    = 000100            8403#
SW7    = 000200            8403#
SW8    = 000400            8403#
SW9    = 001000            8403#
S110   = 001000            8403#
S1200  = 003400            8403#
S134   = 001400            8403#
S150   = 002000            8403#
S1800  = 004000            8403#
S19200= 007400             8403#
S2000  = 004400            8403#
S2400  = 005000            8403#
S300   = 002400            8403#
S3600  = 005400            8403#
S4800  = 006000            8403#
S50    = 000000            8403#   8748    8751
S600   = 003000            8403#
S7200  = 006400            8403#
S75    = 000400            8403#
S9600  = 007000            8403#
TBITVE= 000014             8403#
TCR0   = 000001            8403#
TCR1   = 000002            8403#
TCR2   = 000004            8403#
```

```
TCR3   = 000010            8403#
TDO      001426            8403#   8405    8447    8448*   8552    8553*   8554    8564    8642    8643*   8818    8823    8824*
TD1      001430            8403#
TD2      001432            8403#
TD3      001434            8403#
TEIGHT   002106            8403#
TFIVE    002114            8403#
TIE    = 040000            8403#   8509    8513    8604    8787    8807    8835
TKVEC  = 000060            8403#
TLAST  = 014132            8405    9071#
TL0    = 000000            8403#
TL1    = 000400            8403#
TL2    = 001000            8403#
TL3    = 001400            8403#
TMTBL    002050            8403#   8883*   8887    8889
TPVEC  = 000064            8403#
TRAPVE= 000034             8403#
TRDY   = 100000            8403#
TRTVEC= 000014             8403#
TRO      001436            8403#   8684    8689    8691*   8692    8707    8807*   8870    8872    8876*   8877    8879
TR1      001440            8403#
TR2      001442            8403#
TR3      001444            8403#
TSEVEN   002110            8403#
TSIX     002112            8403#
TST1     012010            8405    8423#
TST2     012452            8423    8526#
TST3     012752            8526    8594#
TST4     013514            8594    8737#
TST5     014132            8737    8807#   9071
TST6   = ****** U          8807
TTST     004366            8403*   8405#
TWOSTO= 000040             8403#   8748    8751
TXSVC    013120            8602    8622#
TXTCR    013232            8597*   8637    8645*   8648#
TYPDAT   006672            8405#
TYPE   = 104402            8403#   8405    9205
TYPMSG   006562            8405#
T110     002054            8403#
T1200    002066            8403#
T134     002056            8403#
T150     002060            8403#
T1800    002070            8403#
T2000    002072            8403#
T2400    002074            8403#
T300     002062            8403#
T3600    002076            8403#
T4800    002100            8403#
T50      002050            8403#
T600     002064            8403#
T7200    002102            8403#
T75      002052            8403#
T9600    002104            8403#
VECMAP   011620            8405#
WRDCNT   006204            8405#*
WTBS.F   006660            8405#
```

```
XBX      006450            8405#
XCSR     004324            8405#
XERR     004346            8405#
XHEAD    010046            8403    8405#
XMTCNT   001400            8403#   8781*   8807*   8818
XMTLIN   001376            8403#   8746*   8752*   8754    8775
XMTSRV   014562            8782    8807    8813#
XPASS    004340            8405#
XSTATQ   010136            8403    8405#
XTSTN    007040            8405#
XVEC     004332            8405#
XX     = 160210            8403#
YY     = 000500            8403#
ZZ     = 000020            8403#
$APTHD   001446            8403#
$ASTAT= ****** U           8405
$ATYC    005170            8405#
$ATY1    005144            8405#
$ATY3    005152            8405#
$ATY4    005162            8405#
$AUTOB   001300            8403#
$BASE    001174            8403#*  8405*   9175    9177*
$BDADR   001266            8403#
$BDDAT   001272            8403#
$CDW1    001200            8403#   8405
$CDW2    001202            8403#   8405*
$CHARC   005140            8405#*
$CLKVE   017252            9205#
$CMTAG   001244            8403#
$CM1   = 000006            8403#
$CM2   = 000014            8403#
$CM3   = 000006            8403#
$CM4   = 000005            8403#
$CPUOP   001146            8403#
$CRLF    001357            8403#   8405
$DDW0    001204            8403#   8405
$DDW1    001206            8403#
$DDW10   001230            8403#
$DDW11   001232            8403#
$DDW12   001234            8403#
$DDW13   001236            8403#
$DDW14   001240            8403#
$DDW15   001242            8403#
$DDW2    001210            8403#
$DDW3    001212            8403#
$DDW4    001214            8403#
$DDW5    001216            8403#
$DDW6    001220            8403#
$DDW7    001222            8403#
$DDW8    001224            8403#
$DDW9    001226            8403#
$DEVCT   001130            8403#   8405*
$DEVM    001176            8403#   8405*
$DOAGN   004320            8405#
$E     = 000007            6801#   8423#   8526#   8594#   8737#   8807#
$ENDAD   004310            8403    8405#
```

```
$ENDCT  004274          8405#
$ENV    001140          8403#   8405    8738
$ENVM   001141          8403#   8405
$EOP    004140          8405#   8807
$EOPCT  004266          8405#
$ERFLG  001247          8403#*  8405*
$ERMAX  001261          8403#   8405
$ERROR  006420          8403    8405#
$ERRPC  001262          8403#*  8405*
$ERRTB  001362          8403#
$ERTTL  001256          8403#*  8405*
$ETABL  001140          8403#
$ETEND  001244          8403#
$FATAL  001122          8403#   8405*
$FFLG   005410          8405#*
$FILLC  001322          8403#   8405
$FILLS  001321          8403#   8405
$FLIP = 177777          5676#   8410#   8422#   8518#   8525#   8588#   8593#   8718#   8736#   8807#
$GDADR  001264          8403#
$GDDAT  001270          8403#
$GET42  004300          8405#
$HD   = 000001          8403
$HIBTS  001446          8403#
$ICNT   001250          8403#   8405*
$ILLUP  007456          8405#
$INTAG  001301          8403#
$ITEMB  001260          8403#   8405*
$LF     001360          8403#   8405
$LFLG   005407          8405#*
$LPADR  001252          8403#*  8405*
$LPERR  001254          8403#
$MADR1  001152          8403#
$MADR2  001156          8403#
$MADR3  001162          8403#
$MADR4  001166          8403#
$MAIL   001120          8403#   8405
$MAMS1  001150          8403#
$MAMS2  001154          8403#
$MAMS3  001160          8403#
$MAMS4  001164          8403#
$MBADR  001450          8403#
$MFLG   005406          8405#*
$MSGAD  001134          8403#   8405*
$MSGLG  001136          8403#   8405*
$MSGTY  001120          8403#   8405*
$MTYP1  001151          8403#
$MTYP2  001155          8403#
$MTYP3  001161          8403#
$MTYP4  001165          8403#
$MXCNT  004616          8405#
$N    = 000005          6800#   8410    8423#   8518    8526#   8588    8594#   8718    8737#   8807#   9068#   9069
$NULL   001320          8403#   8405
$NWTST= 000000          8423#   8526#   8594#   8737#   8807#
$OVER   004554          8405#
$PASS   001126          8403#*  8405*   8740
$PASTM  001454          8403#
```

```
$PWRAD  007452            8405#
$PWRDN  007312            8403    8405#
$PWRMG  007446            8405#
$PWRUP  007364            8405#
$QUES   001356            8403#   8405
$REGAD  001324            8403#
$REG0   001326            8403#   8405*   9036    9044
$REG1   001330            8403#   8405*   9034
$REG2   001332            8403#   8405*
$REG3   001334            8403#   8405*
$REG4   001336            8403#   8405*   9042    9054
$REG5   001340            8403#   8405*   9040    9052    9062
$RTNAD  004322            8405#
$SAVR6  007462            8405#*
$SCOPE  004354            8405#
$SETUP= 000000            8405
$SVLAD  004536            8405#
$SVPC = 000040            8403#
$SWR  = 164000            5683#   8403    8405    8423    8526    8594    8737    8807
$SWREG  001142            8403#   8405
$SWRMK= 000000            8405
$TESTN  001124            8403#   8405*
$TIMES  001354            8403#   8405*   8743*
$TKB    001312            8403#   8405
$TKS    001310            8403#   8405
$TMP0   001342            8403#   8778*   8807*   8825*   8879
$TMP1   001344            8403#*  8405    8779*   8791*   8844*   8883    8887    9064
$TMP2   001346            8403#*  8405    8842*   8845*
$TMP3   001350            8403#   8780*   8793*   8843*
$TMP4   001352            8403#
$TN   = 000006            8403#   8423#   8526#   8594#   8737#   8807#
$TPB    001316            8403#   8405*
$TPFLG  001323            8403#   8405
$TPS    001314            8403#   8405
$TSTM   001452            8403#
$TSTNM  001246            8403#*  8405*   8423*   8526*   8594*   8737*   8807*
$TYPE   004662            8405#
$TYPEC  005074            8405#
$TYPEX  005142            8405#
$UNIT   001132            8403#
$UNITM  001456            8403#
$USWR   001144            8403#
$VECT1  001170            8403#*  8405*   9178    9180*
$VECT2  001172            8403#
$XTSTR  004414            8405#
$Y    = 000023            8403#
$$GET4= 000000            8405#
$40CAT= ****** U          8405
.     = 017326            8403#   8405#   8437    8471    8540    8557    8563    8627    8635    8652    8655    8680    8693
                          8708    9027#   9205#
.ADVAN  006306            8403    8405#
.BEGIN  004044            8403#
.BUFSE  006376            8403    8405#
.CNVRT  006034            8403    8405#
.CONVR  006030            8403    8405#
.DCLAS  006254            8403    8405#
```

```
.DELAY  006266            8403    8405#
.DEVIC  006234            8403    8405#
.ERRTA  015136            8405    8896#
.INSTE  005516            8403    8405#
.INSTR  005412            8403    8405#
.INST1  005432            8405#
.LPRSE  006336            8403    8405#
.MSG    005434            8405#*
.PARAM  005536            8403    8405#
.PARMD  011154            8403    8405#
.PAWCH  010270            8403    8405#
.RESO5  005776            8403    8405#
.SAVO5  005736            8403    8405#
.SCOPE  004354            8403    8405#
.SCOP1  004620            8403    8405#
.SETFL  010150            8403    8405#
.SHIFT  006320            8403    8405#
.START  002116            8403#
.TRPSR  006212            8403    8405#
.TRPTA  001742            8403#   8405
.TYPE   004644            8403    8405#
.$ASTA= ****** U          8405
.$X   = 001446            8403#
```

```
COMMEN   1566#  8403#
ENDCOM   1578#  8403#
ERROR    8403#  8438   8446   8453   8462   8480   8483   8494   8551   8558   8571   8576   8579   8616   8617
         8628   8636   8653   8656   8664   8667   8670   8673   8681   8690   8709   8795   8807   8822   8855
         8858   8861   8864   8867   8875   8890
ESCAPE   1698#  8403#
GETPRI   1312#  8403#
GETSWR   1771#  8403#
MULT     4484#  8403#
NEWTST   1626#  8403#  8423   8526   8594   8737   8807
PASEND   6831#  8405
POP      2153#  8403#  8405
PRGEND   7873#  8405
PRGFRT   6978#  8403
PUSH     2145#  8403#  8405
REPORT   5463#  5682#  8403#
SC       6771#  8405
SCOPE    8403#  8405   8423   8526   8594   8737   8807
SC1      6780#  8405
SETPRI   1279#  8403#
SETUP    1337#  8403#
SKIP     1733#  8403#
SLASH    1517#  8403#
SPACE    8403#
STARS    1485#  8403#  8405   8423   8526   8594   8737   8807
SWRSU    1453#  8403#
TYPBIN   2088#  8403#
TYPDEC   2058#  8403#
TYPNAM   1826#  8403#
TYPNUM   2025#  8403#
TYPOCS   1978#  8403#
TYPOCT   1941#  8403#
TYPTXT   1894#  8403#
$BUFFE   6909#  8405
$CYCLE   7554#  8405
$EOP     6851#  8405
$GETFL   6715#  8403
$GETPA   6705#  8403   8405
$HEADE   6466#  8403
$INTSE   6955#  8807
$JUNK    6941#  8403
$MRESE   6696#  8758
$MSG     6876#  8405
$PARTS   8302#  8807
$SCOPE   6789#  8405
$SETFL   6721#  8405
$STAG    6962#  8474
$STAGF   6949#
$TRPDE   6491#  8403
$TSTN    6803#  8423   8526   8594   8737   8807
$VARIA   6503#  8403
$XZ      7855#  8410   8422   8518   8525   8588   8593   8718   8736   8807
$$CMRE   8403#
$$CMTM   8403#
$$ESCA   1711#  8403#
$$NEWT   1662#  8403#  8423   8526   8594   8737   8807
```

```
$$SKIP   1746#   8403#
.EQUAT    189#   5681#   8403
.HEADE     61#   5680#   8403
.INIT    5658#   8403#   9205
.SETUP   1213#   5680#
.SWRHI    104#
.$ACT1   5064#   5682#   8403
.$APTB   5109#   5682#   8403#
.$APTH   5370#   5682#   8403
.$APTY   5547#   5682#   8405
.$ASTA   5417#
.$CATC    932#   5680#
.$CMTA   1047#   8403#
.$DB2D   4686#
.$DB2O   4812#
.$DIV    4587#
.$EOP    2214#   5680#   8405
.$ERRO   2700#   5681#
.$ERRT   2896#
.$MULT   4523#
.$POWE   4229#   5681#   8405
.$RAND   4307#
.$RDDE   3891#
.$RDOC   3797#
.$READ   3395#
.$R2AZ   4958#
.$SAVE   3969#
.$SB2D   4771#
.$SB2O   4874#
.$SCOP   2454#   5681#   8405
.$SIZE   4361#
.$SUPR   4913#
.$TRAP   4073#   5681#
.$TYPB   3287#
.$TYPD   3209#
.$TYPE   2985#   5680#   8405
.$TYPO   3112#
.$40CA    972#

. ABS.  017326     000

 ERRORS DETECTED:  0

 CNDZBA,CNDZBA/CR/NL:TOC=CNMAC2.SML,CNDZBA.P11
 RUN-TIME: 16 18 1 SECONDS
 RUN-TIME RATIO: 110/36=2.9
 CORE USED:  50K  (100 PAGES)
```