

DZV11

CABLE/ECHO TST
CNDZCAO

AH-T440A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 82-83
MADE IN USA



CNDZCA
CNDZCA.P11 10-DEC-82 15:31
.REM 8

::GPA MACY11 30(1046) ^{B 1} 14-DEC-82 10:09 PAGE 57

SEQ 0001

IDENTIFICATION

PRODUCT CODE: AC-T439A-AC
PRODUCT NAME: CNDZCA0 DZV11 CABLE/ECHO TST
PRODUCT DATE: DEC, 1982
MAINTAINED: DIAGNOSTIC SERVICES/ISS
AUTHOR: L.FLORYAN

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

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

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

COPYRIGHT (C) 1982,1983 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

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

CURRENTLY THERE ARE THREE STANDALONE DIAGNOSTICS (DVDZA, DVDZB, AND DVDZC) 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.

```

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

```

2. REQUIREMENTS

2.1 EQUIPMENT

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

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 THE OPERATOR IF THE PARAMETERS HAVE BEEN ALREADY BUILT BY RUNNING EITHER THE CNDZA OR CNDZB DIAGNOSTICS. LOADING THIS DIAGNOSTIC WILL PRESERVE THESE LOCATIONS.

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 THE SWR TO ALLOW THE DESIRED PROGRAM OPTIONS TO FUNCTION.
NOTE: LOC. 000176 IS USED AS A SOFTWARE SWITCH REGISTER IN ALL OF THE DZV11 DIAGNOSTICS. (SEE SEC. 4.1)
- 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).
- C. THE PROGRAM WILL THEN ASK FOR THE DEVICE ADDRESS, THE VECTOR AND THE LINE NO. OF THE DZV11 TO BE TESTED. TYPE THESE VALUES ON THE CONSOLE TERMINAL FOLLOWED BY A <CR>. THE PROGRAM WILL THEN ASK FOR WHICH TEST IS DESIRED, ECHO OR CABLE. TYPE EITHER E OR C AND A <CR>. THE DIAGNOSTIC WILL TYPE OUT THE NAME OF THE TEST THAT IS NOW RUNNING (SEE SEC. 5.1).

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: RESERVED
SW 13	SET: INHIBIT ERROR PRINT OUT
SW 12	SET: INHIBIT **ALL** TYPE OUT/BELL ON ERROR.
SW 11	SET: RESERVED
SW 10	SET: GO TO END OF PASS AFTER AN ERROR
SW 09	SET: LOOP WITH CURRENT DATA (SEE SEC. 4.1.1)
SW 08	SET: RESTART TEST AFTER AN ERROR
SW 07	SET: RESERVED
SW 06	SET: RESERVED
SW 05	SET: RESERVED
SW 04	SET: RESERVED
SW 03	SET: RESERVED
SW 02	SET: RESERVED
SW 01	SET: RESERVED
SW 00	SET: RESERVED

4.1.1 SWITCH REGISTER RESTRICTIONS

SW 09 LOOP ON CURRENT DATA: THIS SWITCH IS ONLY USED IN THE CABLE TEST TO LOCK ON TESTING IF SETTING THE DTR BIT FOR THE DESIRED LINE IN THE TRANSMIT CONTROL REGISTER OF THE DZV11 WILL CAUSE THE CO AND RING BITS TO SET FOR THAT LINE IN THE MODEM STATUS REGISTER. 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.

4.1.2 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 RESTART THE TEST AFTER AN ERROR
5. SW 10 GO TO THE END OF PASS AFTER AN ERROR

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOPI'). IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. ON AN ERROR REPORT THEN SW09 IS INCORPORATED IN THAT TEST. THIS SWITCH PROVIDES THE OPERATOR WITH THE ABILITY TO LOCK ON A SPECIFIC TEST OPERATION. IF THE PROGRAM USER IS TECHNICALLY TRAINED TO ELECTRONICALLY ISOLATE SIGNAL PROBLEMS ON THE DZV11 MODULE, THIS SWITCH MIGHT PROVE TO BE A USEFUL AID. PRESENTLY THIS SWITCH IS ONLY USED IN THIS DIAGNOSTIC FOR THE CABLE TEST TO LOCK ON CHECKING THAT IF DTR IS SET FOR AN ACTIVE LINE THE CO AND RING WILL BECOME SET FOR THAT LINE.

4.2 STARTING ADDRESS

SA 200 - THE STARTING ADDRESS FOR ANY DZV11 DIAGNOSTIC IS LOC. 200

NOTE: THIS DIAGNOSTIC IS NOT DESIGNED TO RUN IN AN AUTOMATIC CHAIN MODE BECAUSE OF THE OPERATOR INTERVENTION REQUIRED TO RUN IT.

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 HOW TO RUN THE "CABLE/ECHO" TESTS.

NORMAL STARTING PROCEEDURE FOR THE FIRST TIME WOULD BE:
LOAD THE DIAGNOSTIC, SET THE SWR AT LOC. 176 TO WHATEVER SETTINGS ARE
DESIRED, THEN START THE PROGRAM AT LOC. 200.
THE PROGRAM WILL PRINT OUT ON THE CONSOLE TERMINAL:

"VECTOR ADDRESS-"

YOU TYPE A VECTOR FOLLOWED BY A <CR>.

"CONTROL REGISTER ADDRESS-"

YOU TYPE IN THE DZVCSR ADDRESS UNDER TEST FOLLCWED BY A <CR>.

"WHICH TEST ? ECHO OR CABLE (E OR C)"

LETS DO THE CABLE TEST FIRST. TYPE "C" AND A <CR>.

"BAUD RATE- "

TYPE EITHER 50, 110, 135, 150, 300, 600, 1200 1800, 2000, 2400,
3600, 4800, 7200, 9600 FOLLOWED BY <CR>

"LINE: "

YOU TYPE THE LINE WHICH HAS THE H325 TEST CONNECTOR. (TYPE
EITHER 0, 1, 2, 3) PROGRAM WILL THEN PRINT:

"CABLE TEST"

AND IF EVERYTHING IS WORKING, THE END OF PASS MESSAGE WILL BE
PRINTED AFTER EACH PASS.

TO CHANGE LINES, HIT ANY PRINTING KEY ON YOUR CONSOLE TERMINAL
WHILE THE PROGRAM IS RUNNING AND THE FOLLOWING WILL BE PRINTED:

"LINE: "

NOW CHANGE THE H325 TEST CONNECTOR TO ANOTHER LINE AND TYPE THE
NEW LINE. PROGRAM WILL THEN PRINT:

"CABLE TEST"

AND BEGIN RUNNING THE DIAGNOSTIC.
CONTINUE THIS OPERATION UNTIL ALL LINES ARE TESTED.

5.2 ECHO TEST

START THE PROGRAM AT LOC. 200 AND ENTER THE VALUES FOR THE CSR ADDRESS AND THE DEVICE VECTOR. THE PROGRAM WILL THEN PRINT OUT ON THE CONSOLE:

"WHICH TEST ? ECHO OR CABLE (E OR C)"

NOW TYPE AN 'E' TO DO THE ECHO TEST. PROGRAM WILL PRINT:

"BAUD RATE-"

TYPE THE BAUD RATE. BAUD RATE CHOICES ARE: 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600. THE PROGRAM WILL THEN PRINT:

LINE: "

TYPE THE LINE NUMBER WHICH THE TERMINAL IS CONNECTED TO. THEN THE PROGRAM WILL PRINT:

"TERMINAL ECHO TEST"

*** AT THIS POINT THE MESSAGE:

"THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789"

SHOULD BE PRINTED ON THE TERMINAL CONNECTED TO THE DZV11. IF THIS MESSAGE IS DESIRED TO BE PRINTED CONTINUOUSLY, TYPE A CONTROL G <^G> ON THE CONSOLE TERMINAL WHILE THE MESSAGE IS PRINTING. THE PROGRAM WILL PRINT A PROMPT ON THE CONSOLE ASKING FOR A NEW SWR SETTING. BY SETTING THE SWR TO 377 THE QUICK BROWN FOX MESSAGE WILL BE CONTINUOUSLY PRINTED ON THE DZV TERMINAL. A CONTROL G CAN THEN BE TYPED ON THE CONSOLE TERMINAL AT ANY TIME TO RESET THE SWR AND RETURN TO THE FLOW OF THE DIAGNOSTIC. THE PROGRAM WILL THEN PRINT ON THE CONSOLE TERMINAL:

"TYPE A CHAR. ON DZV11 TERMINAL"

ANY PRINTABLE CHARACTER WHICH IS TYPED ON THE DZV11 TERMINAL WILL BE ECHOED BACK ON THE TERMINAL. IF YOU TYPE CONTROL C <^C> ON THE DZV11 TERMINAL THE PROGRAM WILL PRINT THE END OF PASS MESSAGE ON THE CONSOLE TERMINAL AND THE "QUICK BROWN FOX" MESSAGE WILL BEGIN PRINTING ON THE DZV11 TERMINAL AGAIN, THE ECHO TEST WILL BE RESUMED.

TO CHANGE LINES:

TYPE ANY PRINTABLE CHARACTER ON THE CONSOLE TERMINAL (NOT THE DZV11 TERMINAL). THE PROGRAM WILL AGAIN TYPE "LINE: " AND WAIT FOR A RESPONSE.

5.3 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 THE OPERATOR TO REGAIN CONTROL OF THE CPU. IT WILL THEN BE NECESSARY TO CHECK THE PC PROCESSOR REGISTER AND REFER TO THIS LOCATION IN THE PROGRAM LISTING TO FIND OUT WHAT THE PROGRAM WAS DOING AT THE TIME OF THE ERROR.

7. OPERATING RESTRICTIONS

WHEN RUNNING THE CABLE TEST, THE LINE THAT IS DECLARED ACTIVE MUST BE TERMINATED BY AN H325 TEST CONNECTOR WHICH WILL TURN THE TRANSMITTED SIGNAL AROUND TO THE RECEIVER ON THE SAME LINE. THE DIAGNOSTIC IS NOT DESIGNED TO DETERMINE A LOGIC PROBLEM WITH THE DZV INTERFACE. IT IS DESIGNED ONLY TO VERIFY THAT THE INTERFACE CABLE IS PROVIDING A TRUE LINK TO THE TERMINALS WHICH ARE CONNECTED TO THE DZV11.

8. MISCELLANEOUS

8.1 EXECUTION TIME

THE EXECUTION TIME FOR THE CABLE TEST DEPENDS UPON THE DESIRED BAUD RATE GIVEN AT START UP TIME. AT 9600. BAUD THE END PASS MESSAGE WILL PRINT OUT BEFORE 10 SECONDS HAVE ELAPSED.
THE EXECUTION TIME FOR THE ECHO TEST IS ENTIRELY DEPENDENT UPON THE NUMBER OF CHARACTERS THE OPERATOR WISHES TO SEND.

8.2 PASS COMPLETE

WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CNDZC-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

AFTER THE BASE DEVICE ADDRESS AND THE BASE VECTOR HAVE BEEN TYPED IN, LOCATIONS 201 THROUGH 2046 WILL CONTAIN THE VARIOUS DEVICE REGISTER ADDRESSES AND THE DEVICE VECTORS. LOCATION 1374 (SAVLIN) WILL CONTAIN THE LINE NUMBLR THAT WAS DECLARED ACTIVE.

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 (CNDZA, AND CNDZB) CAN BE RUN AS STANDALONE DIAGNOSTICS OR IN EITHER OF THE APT MODES. CNDZC, HOWEVER IS DESIGNED AS A STANDALONE DIAGNOSTIC ONLY AND REQUIRES DIRECT OPERATOR PARTICIPATION.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

ONLY ONE VARIABLE IN THE REGION SUBTITLED "APT MAILBOX-ETABLE" NEEDS TO BE SET UP BEFORE RUNNING UNDER APT. THIS VARIABLE IS:

\$\$WREG -(1142) USED AS THE SOFTWARE SWITCH REGISTER WHILE RUNNING UNDER APT.

9.1.3 RUNNING UNDER APT

\$\$WREG (LOC. 1142) SHOULD BE SET UP PRIOR TO RUNNING THE DIAGNOSTIC.

10.0 PROGRAM DESCRIPTION.

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1982.

11 STARTING PROCEDURE
LOAD PROGRAM
START THE PROGRAM AT LOC. 000200
PROGRAM WILL TYPE DZV11 ECHO/CABLE TEST
PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
TYPE IN E OR C RESPECTIVELY
PROGRAM WILL TYPE "VECTOR ADDRESS-"
TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"
TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "LINE NUMBER-"
TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
, FOLLOWED BY <CARRIAGE RETURN>
PROGRAM WILL TYPE "BAUD RATE-"
TYPE IN THE BAUD RATE OF THE DZV11 TERMINAL
, FOLLOWED BY <CARRIAGE RETURN>
THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL
50
75
110
135 (ROUNDED OFF 134.5)
150
300
600
1200
1800
2000
2400
3600
4800
7200
9600
ALL OTHERS ARE REJECTED

47 PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTED

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

79 MISCELLANEOUS DEFINITIONS
91 GENERAL PURPOSE REGISTER DEFINITIONS
103 PRIORITY LEVEL DEFINITIONS
113 "SWITCH REGISTER" SWITCH DEFINITIONS
141 DATA BIT DEFINITIONS (BIT00 TO BIT15)
169 BASIC "CPU" TRAP VECTOR ADDRESSES
384 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
392 MEM.TYPE BYTE -- (HIGH BYTE)
900 NSEC (ORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
397 MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
436 THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
USED IN THE PROGRAM.
488 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 CONIAINS 4 POINTERS EXPLAINED AS FOLLOWS:
494 EM ;:POINTS TO THE ERROR MESSAGE
DH ;:POINTS TO THE DATA HEADER
DT ;:POINTS TO THE DATA
DF ;:POINTS TO THE DATA FORMAT
873 INCREMENT THE PASS NUMBER (\$PASS)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO XBEGIN
995 ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
NOTE1: \$N'LL 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 ASCII STRING
OR

TYPE
MESADR

- 1728 ***** ECHO TEST *****
THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
(IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
- 1799 ***** CABLE TEST *****
THIS TEST TRANSMITS A BINARY COUNT PATTERN
VIA INTERRUPT MODE TO THE RECEIVER
...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
- 1808 TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
JUMPERS W1,W2,W3 AND W4 MUST BE INSTALLED ON THE
INTERFACE MODULE OTHERWISE AN ERROR REPORT WILL RESULT.

8


```

(1) SW14=40000 :=1, LOOP ON CURRENT TEST
(1) SW13=20000 :=1, INHIBIT ERROR TIMEOUT
(1) SW12=10000 :=1, DELETE TIMEOUT/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 INITIAL 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) 17C000 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

```

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09 PAGE 77-3
 CNDZCA.P11 10-DEC-82 15:31 BASIC DEFINITIONS

SEQ 0017

```

(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  PUSHRO=10046     ;SAVE R0 ON STACK
(1)      012600  POPRO=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  TLO=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)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)

:TABLE OF LOOP AROUND FUNCTIONS (H325)

I	^
V	^
REC	TRANS
DATA	DATA

I	^
V	^
CO	RTS

I	^
V	^
RING	DTR

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09 PAGE 77-7
CNDZCA.P11 10-DEC-82 15:31

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

SEQ 0021

```

(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) . =24
(1) 000024 005622 $PWRDN ;POWER FAIL HANDLER
(1) 000026 000300 300 ;SERVICE AT PRIORITY LEVEL 6 VER:0
(1) 000030 004730 $ERROR ;ERROR HANDLER
(1) 000032 000300 300 ;SEFVICE AT PRIORITY LEVEL 6 VER:0
(1) 000034 004522 .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 002670 $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) . =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 000200 . =200
(1) 000200 000137 002116 JMP .START ;GO TO START OF PROGRAM
(2)
(2) . =1000
(2) 001000 001000 047103 055104 MTITLE: .ASCIZ <200><12>/CNDZCA/<200>/DZV11 ECHO AND CABLE TESTS /<200>
(2)

```

```

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

```



```
(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 .STSTNM: .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 ERRGR 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 $REGAD: .WORD 0 ::CONTAINS THE ADDRESS FROM
(3) WHICH ($REGO) 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: .ASCII <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) $ERRTB:
(3) ;PROGRAM CONTROL PARAMETERS
(3) ;-----
(3)
(3) 001362 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
(3) 001364 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP
(3)
(3) ;PROGRAM VARIABLES
(3) ;-----
(3)
(3) 001366 000017 LINP: 17 ;DEFAULT ALL FOUR LINES RUNNING
(3) 001370 017470 PAR: 17470 ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NJ PARIT
(3) 001372 000000 MODE: 0 ;DEFAULT MAINTENANCE MODE
(3) 001374 000000 SAVLIN: 0 ;LINE NUMBER
(3) 001376 000000 XMTLIN: 0 ;TRANSMISSION LINE NUMBER
(3) 001400 000000 XMTCNT: 0 ;COUNT OF WORDS IN A TRANSMISSION PATTERN
(3) 001402 000000 REGIST: 0 ;DEVICE ADDRESS STORAGE LOCATION
(3) 001404 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
(3) 001406 000001 DZVACTV: .BLKW 1 ;*DZV11'S SELECTED ACTIVE.
(3) 001410 000001 SAVACTV: .BLKW 1 ;*A BIT MAP OF DZV11'S IN THE SYSTEM
(3) 001412 000001 RUN: 1 ;*POINTER ONE PAST RUNNING DEVICE.
(3) 001414 000001 DZVNUM: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM
(3) 001415 001 SAVNUM: .BYTE 1 ;*WORKABLE NUMBER.
(3) 001416 000001 SAVNO: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S BEING TESTED
(3) 001420 001420 .EVEN
(3) 001420 001500 ACTIVE: DZV.MAP ;TABLE POINTER.
```

```

(2)
(2)
(2)
(2)
(2) 001422 000 INIFLG: .BYTE 0 ;PROGRAM CONTROL FLAGS
(2) 001423 000 HDRFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
(2) 001424 000 MNTFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
(2) 001425 000 DONFLG: .BYTE 0 ;MAINTENANCE BIT SET FLAG
(2) .EVEN ;TRANSMISSION COMPLETION FLAG
(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) ;*****
(2) ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(3) ;*****
(2) 001446 .SX= ;SAVE CURRENT LOCATION
(2) 000024 =24 ;SET POWER FAIL TO POINT TO START OF PROGRAM
(2) 000024 200 ;FOR APT START UP
(2) 000044 =44 ;POINT TO APT INDIRECT ADDRESS PNTR.
(2) 000044 $APTHDR ;POINT TO APT HEADER BLOCK
(2) 001446 .=.SX ;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 $MBOXADR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
(2) 001452 000000 $STMT: .WORD 0. ;RUN TIME OF LONGEST TEST
(2) 001454 000000 $PASTM: .WORD 0. ;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 0J0052 .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 DZVCO: .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

```

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09 PAGE 77-13
 CNDZCA.P11 10-DEC-82 15:31 APT PARAMETER BLOCK

SEQ 0027

(3)	001520	000001	PAR1:	.BLKW	1	:PARAMETERS
(3)	001522	000001	MANT1:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(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)	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)	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)	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)	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)	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)	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)	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)	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) 001742 .TRPTAB:
(3) 104400 ADVANCE=TRAP+0 ;CALL TO ADVANCE TO NEXT TEST
(2) 001742 004616 .ADVANCE
(3) 104401 SCOP1=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
(2) 001744 003130 .SCOP1
(3) 104402 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
(2) 001746 003154 .TYPE
(3) 104403 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
(2) 001750 003722 .INSTR
(3) 104404 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
(2) 001752 004026 .INSTER
(3) 104405 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
(2) 001754 004046 .PARAM
(3) 104406 SETFLG=TRAP+6 ;CALL TO SET FLAG ROUTINE
(2) 001756 006456 .SETFLG
(3) 104407 SAV05=TRAP+7 ;CALL TO REGISTER SAVE ROUTINE
(2) 001760 004246 .SAV05
(3) 104410 RES05=TRAP+10 ;CALL TO REGISTER RESTORE ROUTINE
(2) 001762 004306 .RES05
(3) 104411 CONVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE
(2) 001764 004340 .CONVRT
(3) 104412 CNVRT=TRAP+12 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
(2) 001766 004344 .CNVRT
(3) 104413 DEVICE.CLR=TRAP+13 ;CALL TO ISSUE A DEVICE CLEAR
(2) 001770 004544 .DEVICE.CLR
(3) 104414 DELAY=TRAP+14 ;CALL TO DELAY FOR FAST CPU'S
(2) 001772 004576 .DELAY
(3) 104415 PARMD=TRAP+15 ;CONVERT DECIMAL STRING TO OCTAL
(2) 001774 002734 .PARMD
(3) 104416 PAWCH=TRAP+16 ;SET FLAG ECHO OR CABLE
(2) 001776 006576 .PAWCH
(3) 104417 DCLASM=TRAP+17 ;CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2) 002000 004564 .DCLASM
(3) 104420 SHIFT=TRAP+20 ;CALL TO ROTATE LINE POINTER
(2) 002002 004630 .SHIFT
(3) 104421 LPRSET=TRAP+21 ;CALL TO SET UP LPR DEVICE REGISTER
(2) 002004 004646 .LPRSET
(3) 104422 BUFSET=TRAP+22 ;CALL TO ZERO BUFFER AREA
(2) 002006 004706 .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)
(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 T EIGHT: 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 000005 .START: RESET ;CLEAR THE WORLD
(1) 002120 012706 001120 MOV #STACK,SP ;SET UP PROCESSOR STACK
(1) 002124 106427 000200 MTPS #MASK ;LOCK OUT INTERRUPTS
(1) 002130 012737 005622 000024 MOV #SPWRDN,@#24 ;SET UP FOR POWER FAIL
(1) 002136 012737 002116 001252 MOV #.START,$LPADR ;SET UP IN CASE OF POWER FAIL
(1) 002144 105737 001141 TSTB $ENVM ;RUNNING UNDER APT?
(1) 002150 100004 BPL 1$ ;IF NOT SKIP SWR SETUP FOR APT
(1) 002152 012737 001142 001304 MOV #$$SWREG,SWR ;SETUP FOR APT SWR
(1) 002160 000405 BR 2$ ;SKIP SOFTWARE SWR SETUP
(1) 002162 012737 000176 001304 1$: MOV #SWREG,SWR ;SETUP SOFTWARE SWITCH REGISTER OTHERWISE
(1) 002170 004737 005404 CALL GETSWR ; GET INITIAL SWITCT SETTING ;;GPA
(1) 002174 012737 000174 001306 2$: MOV #DISPREG,DISPLAY ;SETUP DISPLAY REGISTER
(1) 002202 005037 007050 CLR STFLG ;CLEAR TEST START FLAG
(1) 002206 005037 001126 CLR $PASS ;CLEAR PASS COUNT
(1) 002212 005037 001256 CLR $ERTTL ;CLEAR ERROR COUNT
(1) 002216 105037 001247 CLR $ERFLG ;CLEAR ERROR FLAG
(1) 002222 005037 001246 CLR $STNM ;CLEAR TEST NO. INDICATOR
(1) 002226 005037 007054 CLR LAST ;CLEAR LAST ERROR PC
(1) ;FOLLOWING TWO LINES DELETED TO BE SPECIFIC TO FALCON
(1) ;
(1) ; CALL FALCON ; TEST FOR FALCON (KXT11) ;;GPA
(1) ; BEQ 1000$ ; BR IF NOT ;;GPA
(1) 002232 004737 013224 CALL FALCINI ; INIT FOR FALCON SYSTEM ;;GPA
(1) 002236 1000$: TSTB INIFLG ;HAS TITLE BEEN TYPED YET?
(1) 002242 001010 BNE VEC1 ;IF YES SKIP PRINTING AGAIN
(1) 002244 023727 000042 002670 CMP @#42,#SENDAD ;RUNNING UNDER ACT?
(1) 002252 001402 BEQ 3$ ;IF YES DON'T PRINT TITLE
(1) 002254 104402 001000 TYPE ,MTITLE ;PRINT TITLE
(1) 002260 105337 001422 3$: DECB INIFLG ;INDICATE TITLE ALREADY TYPED
(1) 002264 012701 000300 VEC1: MOV #300,R1
(1) 002270 012702 000302 MOV #302,R2
(1) 002274 010221 1$: MOV R2,(R1)+ ;RESTORE TRAPCATCHER
(1) 002276 005022 CLR (R2)+ ;IN FLOATING VECTOR AREA
(1) 002300 022122 CMP (R1)+,(R2)+ ;UPDATE THE POINTERS
(1) 002302 005737 013502 TST KXTFLAG ; IF FALCON... ;;GPA
(1) 002306 001403 BEQ 1001$ ; ;GPA
(1) 002310 020127 000400 CMP R1,#400 ;...QUIT AT 400. ;;GPA
(1) 002314 000402 402 ; SKIP NEXT ;;GPA
(1) 002316 020127 001000 1001$: CMP R1,#1000
(1) 002322 001364 BNE 1$
(1)
(1) 002324 002324 GETCSR= . ; POINTER FOR FALCON TWEAKER. ;;GPA
(1) 002324 104403 INSTR ;INPUT ADDRESS OF DEVICE CSR
(1) 002326 007124 MREGAD ;MESSAGE "CONTROL REGISTER ADDRESS-"
(1) 002330 104405 PARAM ;CONVERT STRING TO OCTAL
(1) 002332 160000 160000 ;LOW LIMIT
(1) 002334 163770 163770 ;HIGH LIMIT

```

(1)	002336	001174		\$BASE				:LOCATION TO BE FILLED
(1)	002340	007		.BYTE	7			:LSB MASK
(1)	002341	001		.BYTE	1			:NUMBER OF LOCATIONS
(1)				GETVEL=	.			: POINTER FOR FALCON TWEAKER. ;:GPA
(1)	002342	104403		INSTR				:INPUT ADDRESS OF DEVICE VECTOR
(1)	002344	007102		MVECTOR				:MESSAGE 'VECTOR ADDRESS-'
(1)	002346	104405		PARAM				:CONVERT STRING TO OCTAL
(1)	002350	000300		300				:LOW LIMIT
(1)	002352	000770		770				:HIGH LIMIT
(1)	002354	002040		DZVRIV				:LOCATIONS TO BE FILLED
(1)	002356	003		.BYTE	3			:LSB MASK
(1)	002357	004		.BYTE	4			:NUMBER OF LOCATIONS
(1)	002360	004737	007704	JSR		PC,DZVLEV		:GO BUILD DEVICE POINTERS
(1)								
(1)	002364	104403		INSTR				:INPUT WHICH TEST YOU ARE RUNNING
(1)	002366	007311		MWHICH				:ECHO OR CABLE
(1)	002370	104416		PAWCH				:SET FLAG
(1)	002372	007046		WCHFLG				:THIS FLAG
(1)	002374	104403		INSTR				:INPUT BAUD RATE
(1)	002376	007232		MSPEED				:MESSAGE 'BAUD RATE-'
(1)	002400	104415		PARMD				:CONVERT DECIMAL STRING TO OCTAL
(1)	002402	000062		50.				:LOW LIMIT
(1)	002404	022600		9600.				:HIGH LIMIT
(1)	002406	007064		LINESP				:LOCATION TO BE FILLED
(1)	002410	000		.BYTE	0			:LSB MASK
(1)	002411	001		.BYTE	1			:NUMBER OF LOCATIONS
(1)	002412	104413		LINEX:		DEVICE.CLR		:CLEAR DEVICE
(1)	002414	005037	007050	CLR		STFLG		:CLEAR PROGRAM START FLAG
(1)	002420	104403		INSTR				:INPUT LINE NUMBER
(1)	002422	007222		MLINE				:MESSAGE 'LINE NUMBER-'
(1)	002424	104405		PARAM				:CONVERT STRING TO OCTAL
(1)	002426	000000		0				:LOW LIMIT
(1)	002430	000003		3				:HIGH LIMIT
(1)	002432	001374		SAVLIN				:LOCATION TO BE FILLED
(1)	002434	000		.BYTE	0			:LSB MASK
(1)	002435	001		.BYTE	1			:NUMBER OF LOCATIONS
(1)	002436	004537	006652	JSR		R5,SET		
(1)								
(1)	002442	106427	000200	XBEGIN:	MTPS	#MASK		:LOCK OUT INTERRUPTS
(1)	002446	012706	001120	MOV		#STACK,SP		:SET UP PROCESSOR STACK
(1)	002452	005037	007052	CLR		LOCKUP		:CLEAR TIMEOUT
(1)	002456	005737	007046	TST		WCHFLG		:ECHO OR CABLE TEST ?
(1)	002462	001413		BEQ		2\$:ECHO
(1)	002464	012737	010420	MOV		#TST2,\$LPADR		:CABLE TEST
(1)	002472	005737	007050	TST		STFLG		:ARE YOU LOOPING ?
(1)	002476	001017		BNE		1\$:YES
(1)	002500	005137	007050	COM		STFLG		:NO
(1)	002504	104402	007404	TYPE		,MCABLE		:TYPE CABLE TEST
(1)	002510	000412		BR		1\$		
(1)	002512	012737	010044	MOV		#TST1,\$LPADR		:SET UP ECHO TEST
(1)	002520	005737	007050	TST		STFLG		:ARE YOU LOOPING ?
(1)	002524	001004		BNE		1\$:YES
(1)	002526	005137	007050	COM		STFLG		:NO
(1)	002532	104402	007357	TYPE		,MTERM		:TYPE ECHO TEST
(1)	002536			1\$:				

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09
CNDZCA.P11 10-DEC-82 15:31

PAGE 77-20
PROGRAM INITIALIZATION AND START UP.

SEQ 0034

(1) 002536 000177 176510
(1)
7822

RESTART:JMP @SLPADR ;START TESTING,THIS LOCATION IS ALSO
;USED BY THE POWER UP ROUTINE
;;GPA PRGEND DZV11,<END PASS CNDZC-A >,10.

```

7823          :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 XBEGIN
(3)
(3) 002542    $EOP:
(5) 002542    CLR      $ERRPC      :CLEAR LAST ERROR PC
(5) 002546    CLR      $ERFLG      :CLEAR ERROR FLAG
(5) 002552    TYPE     ,MEPASS     :TYPE END PASS
(5) 002556    TYPE     ,MCSR      :TYPE CSR
(5) 002562    CNVRT   ,XCSR      :SHOW IT
(5) 002566    TYPE     ,MVECX     :TYPE VECTOR
(5) 002572    CNVRT   ,XVEC      :SHOW IT
(5) 002576    INC      $PASS      :RAISE PASS COUNT
(5) 002602    TYPE     ,MPASSX    :TYPE PASSES
(5) 002606    CNVRT   ,XPASS     :SHOW IT
(5) 002612    DEC      $PASS      :RESTORE PASS COUNT
(5) 002616    TYPE     ,MERRX    :TYPE ERRORS
(5) 002622    CNVRT   ,XERR      :SHOW IT
(3) 002626    CLR      $TIMES     :ZERO THE NUMBER OF ITERATIONS
(3) 002632    INC      $PASS      :INCREMENT THE PASS NUMBER
(3) 002636    BIC      #100000,$PASS :DON'T ALLOW A NEG. NUMBER
(3) 002644    DEC      (PC)+      :LOOP?
(3) 002646    $EOPCT: .WORD      1
(3) 002650    BGT      $DOAGN     :YES
(3) 002652    MOV      (PC)+,@(PC)+ :RESTORE COUNTER
(3) 002654    $ENDCT: .WORD      1
(3) 002656    $GET42: MOV      @#42,R0 :GET MONITOR ADDRESS
(3) 002660    BEQ      $DOAGN     :BRANCH IF NO MONITOR
(3) 002664    RESET   :CLEAR THE WORLD
(3) 002666    C00005 :GO TO MONITOR
(3) 002670    JSR      PC,(R0)    :SAVE ROOM
(3) 002672    NOP     :FOR
(3) 002674    NOP     :ACT11
(3) 002676    NOP     :ACT11
(3) 002700    $DOAGN:
(3) 002700    JMP      @(PC)+     :RETURN
(3) 002702    $RTNAD: .WORD      XBEGIN
(3)
(2)
(2) 002704    XCSR:   1
(2) 002706    .BYTE  6,2
(2) 002710    DZVCSR
(2) 002712    XVEC:   1
(2) 002714    .BYTE  3,2
(2) 002716    DZVRIV
(2) 002720    XPASS:  1
(2) 002722    .BYTE  6,2
  
```

(2) 002724 001126
 (2) 002726 000001
 (2) 002730 006 002
 (2) 002732 001256
 (1)
 (1)
 (1) 002734 011605
 (1) 002736 012537 003120
 (1) 002742 012537 003122
 (1) 002746 012537 003124
 (1) 002752 112537 003126
 (1) 002756 112537 003127
 (1) 002762 010516
 (1) 002764 005005
 (1) 002766 012704 007536
 (1) 002772 122714 000015
 (1) 002776 001424
 (1) 003000 121427 000060
 (1) 003004 002421
 (1) 003006 121427 000071
 (1) 003012 003016
 (1) 003014 142714 000060
 (1) 003020 005002
 (1) 003022 152402
 (1) 003024 060205
 (1) 003026 122714 000015
 (1) 003032 001410
 (1) 003034 006305
 (1) 003036 010502
 (1) 003040 006305
 (1) 003042 006305
 (1) 003044 060205
 (1) 003046 000754
 (1) 003050 104404
 (1) 003052 000744
 (1)
 (1)
 (1)
 (1) 003054 020537 003122
 (1) 003060 101373
 (1) 003062 020537 003120
 (1) 003066 103770
 (1) 003070 133705 003126
 (1) 003074 001365
 (1)
 (1)
 (1)
 (1) 003076 013704 003124
 (1) 003102 010524
 (1) 003104 062705 000002
 (1) 003110 105337 003127
 (1) 003114 001372
 (1) 003116 000002
 (1) 003120 000000
 (1) 003122 000000
 (1) 003124 000000

XERR: \$PASS
 1
 .BYTE 6,2
 \$ERTTL

 .PARMD: :CONVERT DECIMAL ASCII STRING TO OCTAL
 MOV (SP),R5
 MOV (R5)+,6\$
 MOV (R5)+,7\$
 MOV (R5)+,8\$
 MOV (R5)+,9\$
 MOVB (R5)+,10\$
 MOVB (R5)+,10\$
 MOV R5,(SP)
 2\$: CLR R5
 MOV #INBUF,R4
 CMPB #15,(R4)
 BEQ 3\$
 1\$: CMPB (R4),#0
 BLT 3\$
 CMPB (R4),#9
 BGT 3\$
 BICB #0,(R4)
 CLR R2
 BISB (R4)+,R2
 ADD R2,R5
 CMPB #15,(R4)
 BEQ 4\$
 ASL R5 :X2
 MOV R5,R2 :SAVE X2
 ASL R5 :X4
 ASL R5 :X8
 ADD R2,R5 :TIMES 10
 BR 1\$
 3\$: INSTER
 BR 2\$

 :TEST TO SEE IF NUMBER IS WITHIN LIMITS
 4\$: CMP R5,7\$
 BHI 3\$
 CMP R5,6\$
 BLO 3\$
 BITB 9\$,R5
 BNE 3\$

 :STORE NUMBER AT SPECIFIED ADDRESS
 5\$: MOV 8\$,R4
 MOV R5,(R4)+
 ADD #2,R5
 DECB 10\$
 BNE 5\$
 RTI
 6\$: 0
 7\$: 0
 8\$: 0

```
(1) 003126 000 9$: .BYTE 0
(1) 003127 000 10$: .BYTE 0
(1)
(1) ;CHECK FOR FREEZE ON CURRENT DATA
(1) ;-----
(1) 003130 032777 001000 176146 .SCOP1: BIT #SW09,@SWR ;IS SW09=1(SET)?
(1) 003136 001405 BEQ 1$ ;BR IF NOT SET.
(1) 003140 005737 001364 TST LOCK ;IS THERE A TIGHT LOOP SPECIFIED?
(1) 003144 001402 BEQ 1$ ;IF NO, RETURN
(1) 003146 013716 001364 MOV LOCK,(SP) ;IF YES, GOTO THE ADDRESS IN LOCK.
(1) 003152 000002 1$: RTI ;GO BACK.
(1)
(1) 003154 032777 010000 176122 .TYPE: BIT #SW12,@SWR ;INHIBIT ALL PRINTOUT??
(1) 003162 001403 BEQ $TYPE ;IF NOT, GO TYPE
(1) 003164 062716 000002 ADD #2,(SP) ;SKIP OVER MESSAGE POINTER
(1) 003170 000002 RTI ;RETURN TO WHERE PROCEDURE WAS INVOKED
(2) .SBTTL TYPE ROUTINE
(2)
(2) ;*****
(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) 003172 105737 001323 $TYPE: TSTB $TPFLG ;:IS THERE A TERMINAL?
(2) 003176 100002 BPL 1$ ;:BR IF YES
(2) 003200 000000 HALT ;:HALT HERE IF NO TERMINAL
(2) 003202 000430 BR ;:LEAVE
(2) 003204 010046 1$: MOV R0,-(SP) ;:SAVE R0
(2) 003206 017600 000002 MOV @2(SP),R0 ;:GET ADDRESS OF ASCIZ STRING
(2) 003212 122737 000001 001140 CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
(2) 003220 001011 BNE 62$ ;:NO,GO CHECK FOR APT CONSOLE
(2) 003222 132737 000100 001141 BITB #APTPOOL,$ENVM ;:SPOOL MESSAGE TO APT
(2) 003230 001405 BEQ 62$ ;:NO,GO CHECK FOR CONSOLE
(2) 003232 010037 003242 MOV R0,61$ ;:SETUP MESSAGE ADDRESS FOR APT
(2) 003236 004737 003462 JSR PC,$ATY3 ;:SPOOL MESSAGE TO APT
(2) 003242 000000 61$: .WORD 0 ;:MESSAGE ADDRESS
(2) 003244 132737 000040 001141 62$: BITB #APTCLSP,$ENVM ;:APT CONSOLE SUPPRESSED
(2) 003252 001003 BNE 60$ ;:YES,SKIP TYPE OUT
(2) 003254 112046 2$: MOVB (R0)+,-(SP) ;:PUSH CHARACTER TO BE TYPED ONTO STACK
(2) 003256 001005 BNE 4$ ;:BR IF IT ISN'T THE TERMINATOR
(2) 003260 005726 TST (SP)+ ;:IF TERMINATOR POP IT OFF THE STACK
(2) 003262 012600 60$: MOV (SP)+,R0 ;:RESTORE R0
(2) 003264 062716 000002 3$: ADD #2,(SP) ;:ADJUST RETURN PC
(2) 003270 000002 RTI ;:RETURN
(2) 003272 122716 000011 4$: CMPB #HT,(SP) ;:BRANCH IF <HT>
```

```

(2) 003276 001430          BEQ      8$
(2) 003300 122716 000200  CMPB    #CRLF,(SP)    ;;BRANCH IF NOT <CRLF>
(2) 003304 001006          BNE     5$
(2) 003306 005726          TST    (SP)+         ;;POP <CR><LF> EQUIV
(2) 003310 104402          TYPE                    ;;TYPE A CR AND LF
(2) 003312 001357          *CRLF
(2) 003314 105037 003450  CLRB    $CHARCNT     ;;CLEAR CHARACTER COUNT
(2) 003320 000755          BR     2$            ;;GET NEXT CHARACTER
(2) 003322 004737 003404  5$:    JSR    PC,$TYPEC  ;;GO TYPE THIS CHARACTER
(2) 003326 123726 001322  6$:    CMPB    $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
(2) 003332 001350          BNE     2$            ;;IF NO GO GET NEXT CHAR.
(2) 003334 013746 001320  MOV     $NULL,-(SP)   ;;GET # OF FILLER CHARS. NEEDED
(2)                                ;;AND THE NULL CHAR.
(2) 003340 105366 000001  7$:    DECB    1(SP)      ;;DOES A NULL NEED TO BE TYPED?
(2) 003344 002770          BLT     6$            ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2) 003346 004737 003404  JSR    PC,$TYPEC
(2) 003352 105337 003450  DECB    $CHARCNT     ;;DO NOT COUNT AS A COUNT
(2) 003356 000770          BR     7$            ;;LCOP
(2)
(2)                                ;HORIZONTAL TAB PROCESSOR
(2)
(2) 003360 112716 000040  8$:    MOVB    #' ,(SP)   ;;REPLACE TAB WITH SPACE
(2) 003364 004737 003404  9$:    JSR    PC,$TYPEC  ;;TYPE A SPACE
(2) 003370 132737 000007 003450 BITB    #7,$CHARCNT  ;;BRANCH IF NOT AT
(2) 003376 001372          BNE     9$            ;;TAB STOP
(2) 003400 005726          TST    (SP)+         ;;POP SPACE OFF STACK
(2) 003402 000724          BR     2$            ;;GET NEXT CHARACTER
(2) 003404 105777 175704  $TYPEC: TSTB    @STPS   ;;WAIT UNTIL PRINTER IS READY
(2) 003410 100375          BPL     $TYPEC
(2) 003412 116677 000002 175676 MOVB    2(SP),@STPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)
(2) 003420 122766 000015 000002 CMPB    #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
(2) 003426 001003          BNE     1$            ;;BRANCH IF NO
(2) 003430 105037 003450  CLRB    $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
(2) 003434 000406          BR     $TYPEX        ;;EXIT
(2) 003436 122766 000012 000002 1$:    CMPB    #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
(2) 003444 001402          BEQ     $TYPEX        ;;BRANCH IF YES
(2) 003446 105227          INCB    (PC)+        ;;COUNT THE CHARACTER
(2) 003450 000000          $CHARCNT: .WORD 0    ;;CHARACTER COUNT STORAGE
(2) 003452 000207          $TYPEX: RTS         PC
(2)
(2)
(2)                                .SBTTL  APT COMMUNICATIONS ROUTINE
(2)
(2)                                ;*****
(3) 003454 112737 000001 003720  $ATY1: MOVB    #1,$FFLG  ;;TO REPORT FATAL ERROR
(2) 003462 112737 000001 003716  $ATY3: MOVB    #1,$MFLG  ;;TO TYPE A MESSAGE
(2) 003470 000403          BR     $ATYC
(2) 003472 112737 000001 003720  $ATY4: MOVB    #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
(2) 003500          $ATYC:
(4) 003500 010046          MOV     R0,-(SP)     ;;PUSH R0 ON STACK
(4) 003502 010146          MOV     R1,-(SP)     ;;PUSH R1 ON STACK
(2) 003504 105737 003716  TSTB    $MFLG        ;;SHOULD TYPE A MESSAGE?
(2) 003510 001450          BEQ     5$            ;;IF NOT: BR
(2) 003512 122737 000001 001140  CMPB    #APTENV,$ENV  ;;OPERATING UNDER APT?

```

```

(2) 003520 001031          BNE      3$          ;;IF NOT: BR
(2) 003522 132737 000100 001141 BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(2) 003530 001425          BEQ      3$          ;;IF NOT: BR
(2) 003532 017600 000004          MOV     @4(SP),R0    ;;GET MESSAGE ADDR.
(2) 003536 062766 000002 000004 ADD     #2,4(SP)    ;;BUMP RETURN ADDR.
(2) 003544 005737 001120          1$: TST    $MSGTYPE    ;;SEE IF DONE W/ LAST XMISSION?
(2) 003550 001375          BNE     1$          ;;IF NOT: WAIT
(2) 003552 010037 001134          MOV     R0,$MSGAD
(2)          ;;PUT ADDR IN MAILBOX
(2)          2$: TSTB   (R0)+    ;;FIND END OF MESSAGE
(2) 003556 105720          BNE     2$
(2) 003560 001376          SUB     $MSGAD,R0   ;;SUB START OF MESSAGE
(2) 003562 163700 001134          ASR    R0          ;;GET MESSAGE LNGTH IN WORDS
(2) 003566 006200          MOV     R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
(2) 003570 010037 001136          MOV     #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
(2) 003574 012737 000004 001120 BR      5$
(2) 003602 000413          3$: MOV     @4(SP),4$  ;;PUT MSG ADDR IN JSR LINKAGE
(2) 003604 017637 000004 003630 ADD     #2,4(SP)    ;;BUMP RETURN ADDRESS
(2) 003612 062766 000002 000004 MOV     177776,-(SP) ;;PUSH 177776 ON STACK
(4) 003620 013746 177776          JSR    PC,$TYPE    ;;CALL TYPE MACRO
(2) 003624 004737 003172          4$: .WORD 0
(2) 003630 000000          5$:
(2) 003632 105737 003720          10$: TSTB   $FFLG      ;;SHOULD REPORT FATAL ERROR?
(2) 003636 001416          BEQ     12$        ;;IF NOT: BR
(2) 003640 005737 001140          TST    $ENV        ;;RUNNING UNDER APT?
(2) 003644 001413          BEQ     12$        ;;IF NOT: BR
(2) 003646 005737 001120          11$: TST    $MSGTYPE    ;;FINISHED LAST MESSAGE?
(2) 003652 001375          BNE     11$        ;;IF NOT: WAIT
(2) 003654 017637 000004 001122 MOV     @4(SP),$FATAL ;;GET ERROR #
(2) 003662 062766 000002 000004 ADD     #2,4(SP)    ;;BUMP RETURN ADDR.
(2) 003670 005237 001120          INC    $MSGTYPE    ;;TELL APT TO TAKE ERROR
(2) 003674 105037 003720          12$: CLRB   $FFLG      ;;CLEAR FATAL FLAG
(2) 003700 105037 003717          CLRB   $LFLG      ;;CLEAR LOG FLAG
(2) 003704 105037 003716          CLRB   $MFLG      ;;CLEAR MESSAGE FLAG
(4) 003710 012601          MOV     (SP)+,R1   ;;POP STACK INTO R1
(4) 003712 012600          MOV     (SP)+,R0   ;;POP STACK INTO R0
(2) 003714 000207          RTS    PC          ;;RETURN
(2) 003716 000          $MFLG: .BYTE 0    ;;MESSG. FLAG
(2) 003717 000          $LFLG: .BYTE 0
(2)          ;;LOG FLAG
(2) 003720 000          $FFLG: .BYTE 0    ;;FATAL FLAG
(2)          .EVEN
(2)          APTSIZE=200
(2)          APTENV=001
(2)          APTSPCOL=100
(2)          APTCSUP=040
(1)          ;STRING INPUT ROUTINE
(1)          ;-----
(1) 003722 010346          .INSTR: MOV     R3,-(SP) ;;SAVE R3 ON STACK
(1) 003724 010446          MOV     R4,-(SP)    ;;SAVE R4 ON STACK
(1) 003726 017637 000004 003744 MOV     @4(SP),.MSG  ;;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1) 003734 062766 000002 000004 ADD     #2,4(SP)    ;;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1) 003742 104402          .INST1: TYPE      ;;PRINT THE MESSAGE
(1) 003744 000000          .MSG: 0           ;;MESSAGE IS POINTED TO FROM HERE

```



```

(1) ;-----
(1)
(1) 004166 020537 004240 LIMITS: CMP R5,HILIM ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
(1) 004172 101373 BHI PARERR ;IF YES, GO PRINT THE MESSAGE AGAIN
(1) 004174 020537 004236 CMP R5,LOLIM ;IS THE RESULT LOWER THAN ALLOWED?
(1) 004200 103770 BLO PARERR ;IF YES, GO PRINT THE MESSAGE AGAIN
(1) 004202 133705 004244 BITB LOBITS,R5 ;ARE ANY INCORRECT BITS SET IN THE RESULT?
(1) 004206 001365 BNE PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)
(1) ;STORE NUMBER AT SPECIFIED ADDRESS
(1)
(1) 004210 013704 004242 1$: MOV DEVADR,R4 ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1) 004214 010524 MOV R5,(R4)+ ;STORE THE RESULT
(1) 004216 062705 000002 ADD #2,R5 ;CALCULATE THE NEXT DATUM
(1) 004222 105337 004245 DECB ADCRNT ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1) 004226 001372 BNE 1$ ;IF NOT, GO STORE THE NEXT DATUM
(1) 004230 012604 MOV (SP)+,R4 ;RESTORE R4
(1) 004232 012605 MOV (SP)+,R5 ;RESTORE R5
(1) 004234 000002 RTI ;RETURN TO THE MAIN PROGRAM
(1)
(1) 004236 000000 LOLIM: 0 ;LOWEST ACCEPTABLE VALUE
(1) 004240 000000 HILIM: 0 ;HIGHEST ACCEPTABLE
(1) 004242 000000 DEVADR: 0 ;LOCATION WHERE RESULT WILL BE STORED
(1) 004244 000 LOBITS: .BYTE 0 ;INCORRECT BITS MASK
(1) 004245 000 ADCRNT: .BYTE 0 ;COUNT OF ITEMS TO BE STORED
(1)
(1) ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1) ;-----
(1) 004246 016637 000004 001404 .SAV05: MOV 4(SP),SAVPC ;SAVE R7 (PC)
(1)
(1) ;SAVE R0-R5
(1)
(1) 004254 010537 001340 SV05: MOV R5,$REG5 ;SAVE R5
(1) 004260 010437 001336 MOV R4,$REG4 ;SAVE R4
(1) 004264 010337 001334 MOV R3,$REG3 ;SAVE R3
(1) 004270 010237 001332 MOV R2,$REG2 ;SAVE R2
(1) 004274 010137 001330 MOV R1,$REG1 ;SAVE R1
(1) 004300 010037 001326 MOV R0,$REG0 ;SAVE R0
(1) 004304 000002 RTI ;LEAVE.
(1)
(1) ;RESTORE R0-R5
(1)
(1) 004306 013700 001326 .RES05: MOV $REG0,R0 ;RESTORE R0
(1) 004312 013701 001330 MOV $REG1,R1 ;RESTORE R1
(1) 004316 013702 001332 MOV $REG2,R2 ;RESTORE R2
(1) 004322 013703 001334 MOV $REG3,R3 ;RESTORE R3
(1) 004326 013704 001336 MOV $REG4,R4 ;RESTORE R4
(1) 004332 013705 001340 MOV $REG5,R5 ;RESTORE R5
(1) 004336 000002 RTI ;LEAVE
(1)
(1) ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(1) ;-----
(1)
(1) 004340 104402 001357 .CONVR: TYPE $CRLF ;PRINT A CARRIAGE RETURN
(1) 004344 010046 .CNVRT: MOV R0,-(SP) ;SAVE R0

```



```

(1) 004534 006300          ASL    R0          ;POSITION OFFSET FOR TABLE INDEXING
(1) 004536 016000 001742  MOV    .TRPTAB(R0),R0 ;PLACE INDEXED ADDRESS OF TABLE IN R0
(1) 004542 000200          RTS    R0          ;TRANSFER TO THAT ADDRESS AND RESTORE OLD R0
(1)
(1)                          ;DEVICE CLEAR ROUTINE
(1)                          ;ISSUE A DEVICE CLEAR
(1)
(1)                          ;-----
(1) 004544          .DEVICE.CLR:
(1) 004544 052777 000020 175236  BIS    #DCLR,@DZVCSR ;SET DCLR
(1) 004552 032777 000020 175230 1$: BIT    #DCLR,@DZVCSR ;DID IT CLEAR?
(1) 004560 001374          BNE    1$          ;BR IF NO
(1) 004562 000002          RTI          ;EXIT ROUTINE
(1)
(1)                          ;ROUTINE TO HANDLE MAINTENANCE BIT SETTING WITH DEVICE CLEAR
(1)                          ;-----
(1) 004564 104413          .DCLASM:DEVICE.CLR ;ISSUE A DEVICE CLEAR
(1) 004566 153777 001424 175214  BISB  MNTFLG,@DZVCSR ;LOAD THE MAINTENANCE BIT IF IT IS I MODE
(1) 004574 000002          RTI          ;RETURN TO CALLING ROUTINE
(1)
(1)                          .DELAY:
(1) 004576 010046          MOV    R0,-(SP)     ;SAVE R0
(1) 004600 013700 004614  MOV    DLYCNT,R0   ;SET COUNT
(1) 004604 005300          1$: DEC    R0      ;DELAY
(1) 004606 001376          BNE    1$          ;
(1) 004610 012600          MOV    (SP)+,R0   ;RESTORE R0
(1) 004612 000002          RTI          ;LEAVE ROUTINE
(1) 004614 000001  DLYCNT: .WORD    1 ;PATCHABLE LOC FOR MORE TIME
(1)
(1)                          ;ADVANCE TO NEXT TEST HANDLER
(1)                          ;-----
(1) 004616 013716 001362  .ADVANCE:MOV    NEXT,(SP) ;CRUNCH STACK WITH ADDRESS OF NEXT TEST
(1) 004622 005037 001364  CLR    LOCK       ;RESET TIGHT LOOP ADDRESS
(1) 004626 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) 004630 106302          .SHIFT:ASLB    R2   ;POINT TO THE NEXT LINE
(1) 004632 032702 000020  BIT    #BIT4,R2   ;HAVE WE PASSED ALL LINE POINTERS?
(1) 004636 001402          BEQ    1$         ;IF NOT, RETURN TO THE TEST
(1) 004640 022626          POP2SP ;REMOVE THE TRAP CALL FROM THE STACK
(1) 004642 104400          ADVANCE ;GO TO THE NEXT TEST
(1) 004644 000002  1$: RTI          ;RETURN TO THE PRESENT TEST
(1)

```

```

(1)                                     ;LINE PARAMETER REGISTER SETUP ROUTINE
(1)
(1) 004646 010146 .LPRSET:MOV R1,-(SP) ;SAVE CONTENTS OF R1
(1) 004650 010246 MOV R2,-(SP) ;SAVE CONTENTS OF R2
(1) 004652 013701 001370 MOV PAR,R1 ;MOVE DEFAULT PARAM. INTO R1
(1) 004656 012702 000001 MOV #1,R2 ;INIT. FOR LINE 1
(1) 004662 010177 17132 1$: MOV R1,@DZVLP ;LOAD PARAM. REGISTER
(1) 004666 005201 INC R1 ;SET R1 FOR NEXT LINE
(1) 004670 106302 ASLB R2 ;SET R2 FOR NEXT LINE
(1) 004672 032702 000020 BIT #BIT4,R2 ;ALL LINES DONE?
(1) 004676 001771 BEQ 1$ ;IF NO LOAD NEXT LINE
(1) 004700 012602 MOV (SP)+,R2 ;RELOAD R2
(1) 004702 012601 MOV (SP)+,R1 ;RELOAD R1
(1) 004704 000002 RTI ;RETURN
(1)
(1)                                     ;ROUTINE TO ZERO DATA BUFFER
(1)
(1) 004706 010046 .BUFSET:MOV R0,-(SP) ;SAVE CONTENTS OF P0
(1) 004710 012700 001426 MOV #TDO,R0 ;SET R0 TO TOP OF BUFFER
(1) 004714 005020 1$: CLR (R0)+ ;CLEAR BUFFER LOCATION
(1) 004716 022700 001446 CMP #STOP,R0 ;IS BUFFER ALL CLEARED
(1) 004722 001374 BNE 1$ ;IF NOT CLEAR NEXT LOCATION
(1) 004724 012600 MOV (SP)+,R0 ;RELOAD R0
(1) 004726 000002 RTI ;RETURN
(1)
(1)                                     ;ERROR HANDLER
(1) -----
(1)
(1) 004730 004737 005356 $ERROR: JSR PC,SERV.G ;FIND OUT IF <^G> WAS HIT
(1) 004734 032777 010000 174342 BIT #SW12,@SWR ;BELL ON ERROR?
(1) 004742 001406 BEQ XB ;BR IF NO BELL
(1) 004744 105777 174344 TSTB @STPS ;TTY READY.
(1) 004750 109003 BPL XB ;DON'T WAIT IF TTY NOT READY.
(1) 004752 112777 000207 174336 MOVB #207,@STPB ;PUSH A BELL AT THE TTY.
(1) 004760 032777 020000 174316 XB: BIT #SW13,@SWR ;DELETE ERROR PRINT OUT?
(1) 004766 001113 BNE HALTS ;BR IF NO PRINT OUT WANTED.
(1) 004770 021637 001262 CMP (SP),$ERRPC ;WAS THIS ERROR FOUND LAST TIME?
(1) 004774 001404 BEQ 1$ ;BR IF YES
(1) 004776 011637 001262 MOV (SP),$ERRPC ;RECORD BEING HERE
(1) 005002 105037 001247 CLRB $ERFLG ;PREPARE HEADER
(1) 005006 104407 1$: SAVO5 ;SAVE ALL PROC REGISTERS
(1) 005010 011605 MOV (SP),R5 ;GET THE PC OF ERROR
(1) 005012 162705 000002 SUB #2,R5 ;GET ADDRESS OF TRAP CALL
(1) 005016 011504 MOV (R5),R4 ;GET ERROR INSTRUCTION
(1) 005020 110437 001260 MOVB R4,$ITEMB ;COPY TEST NUMBER FOR APT HANDLING
(1) 005024 006304 ASL R4 ;MULT BY TWO
(1) 005026 061504 ADD (R5),R4 ;DOUBLE IT
(1) 005030 006304 ASL R4 ;MULT AGAIN
(1) 005032 042704 177001 BIC #177001,R4 ;CLEAR JUNK
(1) 005036 062704 011110 ADD #.ERRTAB,R4 ;GET POINTER
(1) 005042 012437 005166 MOV (R4)+,ERRMSG ;GET ERROR MESSAGE
(1) 005046 012437 005200 MOV (R4)+,DATAHD ;GET DATA HEADRER
(1) 005052 011437 005212 MOV (R4),DATABP ;GET DATA TABLE
(1) 005056 105737 001247 TSTB $ERFLG ;TYPE HEADER
(1) 005062 001403 BEQ TYPMSG ;BR IF YES
(1) 005064 005737 005212 TST DATABP ;DOES DATA TABLE EXIST?

```

```

(1) 005070 001044          BNE      TYPDAT      ;BR IF YES.
(1) 005072 104402 001357  TYPMSG: TYPE     , $CRLF    ;TYPE A CARRIAGE RETURN
(1) 005076 104402 001357  TYPE     , $CRLF    ;AND TYPE ANOTHER
(1) 005102 005737 001364  TST      LOCK
(1) 005106 001402          BEQ      1$
(1) 005110 104402 006271  TYPE     , MASTEK
(1) 005114 104402 006257  1$:     TYPE     , MTSTN
(1) 005120 104412 005350  CNVRT   , XTSTN      ;SHOW IT
(1) 005124 104402 006347  TYPE     , MERRPC    ;TYPE PC.
(1) 005130 104412 005342  CNVRT   , ERTABO    ;SHOW IT
(1) 005134 104402 006221  TYPE     , MCSRX
(1) 005140 104412 002704  CNVRT   , XCSR
(1) 005144 104402 001357  TYPE     , $CRLF
(1) 005150 112737 177777  G01247 MOVB    #-1, $ERFLG  ;GIVE A CR/LF
(1) 005156 005737 005166  TST      ERRMSG    ;NO MORE HEADER UNLESS NO DATA TABLE.
(1) 005162 001402          BEQ      WTBS.FM   ;IS THERE AN ERROR MESSAGE?
(1) 005164 104402          TYPE
(1) 005166 000000  ERRMSG: 0          ;BR IF NO.
(1) 005170          WTBS.FM: ;TYPE
(1) 005170 005737 005200  TST      DATAHD   ;ERROR MESSAGE
(1) 005174 001402          BEQ      TYPDAT   ;DATA HEADER?
(1) 005176 104402          TYPE
(1) 005200 000000  DATAHD: 0        ;BR IF NO
(1) 005202 005737 005212  TYPDAT: TST      DATABP ;TYPE
(1) 005206 001402          BEQ      RESREG   ;DATA HEADER
(1) 005210 104411          CNVRT
(1) 005212 000000  DATABP: 0          ;DATA TABLE?
(1) 005214 104410          RESREG: RES05 ;BR IF NO.
(1) 005216 122737 000001 001140 HALTS:  CMPB    #APTENV, $ENV ;SHOW
(1) 005224 001007          BNE      1$        ;DATA TABLE
(1) 005226 113737 001260 005240 MOVB    $ITEMB, 5$ ;RESTORE PROC REGISTERS
(1) 005234 004737 003472  JSR     PC, $ATY4  ;IS APT RUNNING?
(1) 005240 000000  5$:     .WORD    0    ;SKIP APT CALL IF NOT
(1) 005242 000777          BR      10$       ;COPY ERROR NUMBER
(1) 005244 022737 002670 000042 15$:    CMP     #SENDAD, @#42 ;CALL APT SERVICE
(1) 005252 001403          BEQ      20$       ;ERROR NUMBER STUCK HERE
(1) 005254 005777 174024          TST     @SWR      ;LOCK UP HERE
(1) 005260 100004          BPL     EXITER    ;CHECK TO SEE IF IN ACT-11 MODE
(1) 005262 016677 000002 174016 20$:    MOV     2(SP), @DISPLAY ;IF SO, HANDLE ACCORDINGLY
(1) 005270 000000          HALT
(1) 005272 005237 001256  EXITER: INC     $ERTTL ;HALT ON ERROR?
(1) 005276 004737 005356          JSR     PC, $SERV.G ;BR IF NO HALT ON ERROR
(1) 005302 032777 000400 173774 BIT     #SW08, @SWR ;SHOW ERROR PC IN DATA DISPLAY
(1) 005310 001007          BNE      1$        ;UPDATE ERROR COUNT
(1) 005312 032777 002000 173764 BIT     #SW10, @SWR ;FIND OUT IF ^G WAS TYPED
(1) 005320 001407          BEQ      2$        ;GOTO TOP OF TEST?
(1) 005322 013737 001362 001252 1$:     MOV     NEXT, $LPADR ;BR IF YES
(1) 005330 012706 001120          MOV     #STACK, SP ;GOTO NEXT TEST?
(1) 005334 000177 173712          JMP     @ $LPADR  ;BR IF NO
(1) 005340 000002          RTI
(1) 005342 000001  ERTABO: 1          ;SET FOR NEXT TEST
(1) 005344 006      002          .BYTE   6, 2      ;RESET SP
(1) 005346 001404          SAVPC
(1) 005350 000001          XTSTN: 1          ;GOTO SPECIFIED TEST
(1) 005352 002      002          .BYTE   2, 2
(1) 005354 001246          $TSTNM
  
```

APT COMMUNICATIONS ROUTINE

```
(1) 005356 017746 173730 SERV.G: MOV @STKB,-(SP) ;OTHERWISE, GET THE LAST CHARACTER TYPED
(1) 005362 042716 000200 BIC #BIT7,(SP) ;STRIP PARITY(EIGHTH) BIT
(1) 005366 122726 000007 CMPB #7,(SP)+ ;IS IT ^G?
(1) 005372 001076 BNE 6$ ;IF NOT, IGNORE INPUT
(1) 005374 032777 004000 173706 BIT #4000,@STKS ;RX BUSY?
(1) 005402 001365 BNE SERV.G ;BR IF YES
(1) 005404 005404 GETSWR= . ;:GPA
(1) 005404 017737 173674 005612 .MOV @SWR,90$ ;SAVE (SWR).
(1) 005412 104402 005572 1$: TYPE ,89$ ;TYPE HEADER FOR OLD SWITCH REGISTER
(1) 005416 104412 005604 CNVRT ,88$ ;TYPE THE NUMBER ITSELF
(1) 005422 104402 005614 TYPE ,91$ ;AFTER HAVING CONVERTED IT TO ASCII
(1) 005426 105037 005620 CLRB 92$ ;CLEAR SWR CHANGE FLAG
(1) 005432 005077 173646 CLR @SWR ;CLEAR THE SOFTWARE SWITCH REGISTER
(1) 005436 105777 173646 3$: TSTB @STKS ;WAIT FOR DONE.
(1) 005442 100375 BPL 3$ ;CONTINUE WAITING FOR IT
(1) 005444 017746 173642 MOV @STKB,-(SP) ;PUT THE CHARACTER ON THE STACK
(1) 005450 042716 000200 BIC #BIT7,(SP) ;STRIP PARITY BIT
(1) 005454 122726 000015 CMPB #15,(SP)+ ;IS IT THE CARRIAGE RETURN CHAR?
(1) 005460 001433 BEQ 4$ ;IF SO, GO PRINT CRLF
(1) 005462 105777 173626 2$: TSTB @STPS ;IS THE OUTPUT BUFFER AVAILABLE
(1) 005466 100375 BPL 2$ ;IF NOT, WAIT FOR IT TO BE READY
(1) 005470 105237 005620 INCB 92$ ;INDICATE THAT THE SWR WAS CHANGED
(1) 005474 014677 173616 MOV -(SP),@STPB ;PLACE THE CHARACTER THERE(ECHO BACK)
(1) 005500 000241 CLC ;GET READY TO ROTATE
(1) 005502 006177 173576 ROL @SWR ;MOVE THE EXISTING BITS OVER
(1) 005506 006177 173572 ROL @SWR ;TO MAKE ROOM FOR THE INCOMING
(1) 005512 006177 173566 ROL @SWR ;THREE BITS FROM THIS CHARACTER
(1) 005516 103735 BCS 1$ ;ERROR
(1) 005520 022627 000060 CMP (SP)+,#60 ;IS IT LOWER THAN 0?
(1) 005524 002732 BLT 1$ ;IF SO, GO ASK AGAIN
(1) 005526 026627 177776 000067 CMP -2(SP),#67 ;IS IT HIGHER THAN 7?
(1) 005534 003326 BGT 1$ ;IF SO, GO ASK AGAIN
(1) 005536 042746 177770 BIC #^C<7>,-(SP) ;ISOLATE INFORMATION BITS
(1) 005542 052677 173536 BIS (SP)+,@SWR ;ADD THEM TO THE SWITCH REGISTER
(1) 005546 000733 BR 3$ ;GO CHECK FOR THE NEXT CHARACTER
(1) 005550 105737 005620 4$: TSTB 92$ ;HAS THE SWR BEEN CHANGED?
(1) 005554 001003 BNE 5$ ;IF YES GO TYPE CRLF
(1) 005556 013777 005612 173520 MOV 90$,@SWR ;IF NOT RESTORE SWR
(1) 005564 104402 001357 5$: TYPE ,SCRLF ;TYPE A CARRIAGE RETURN AND LINE FEED
(1) 005570 000207 6$: RTS PC ;RETURN TO CALLING PROCEDURE
(1) 005572 020200 051450 051127 89$: .ASCIZ <200>? (SWR)=/?
(1) 005600 036451 000057 .EVEN
(1) 005604 000001 88$: 1
(1) 005606 006 000 .BYTE 6,0
(1) 005610 005612 90$: .WORD 0
(1) 005612 000000 90$: .ASCIZ ?/=/?
(1) 005614 036457 000057 92$: .BYTE 0
(1) 005620 000 .EVEN
(1) 005622 005622 .SBTTL POWER DOWN AND UP ROUTINES
(2)
(2)
(3)
(2)
(2) 005622 012737 005766 000024 ;*****
:POWER DOWN ROUTINE
$PWRDN: MOV #$ILLUP,@#PWRVEC ;:SET FOR FAST UP
```

```

(2) 005630 012737 000300 000026      MOV    #PR6,@#PWRVEC+2  ;;PRIO:6
(4) 005630 010046      MOV    R0,-(SP)         ;;PUSH R0 ON STACK
(4) 005640 010146      MOV    R1,-(SP)         ;;PUSH R1 ON STACK
(4) 005642 010246      MOV    R2,-(SP)         ;;PUSH R2 ON STACK
(4) 005644 010346      MOV    R3,-(SP)         ;;PUSH R3 ON STACK
(4) 005646 010446      MOV    R4,-(SP)         ;;PUSH R4 ON STACK
(4) 005650 010546      MOV    R5,-(SP)         ;;PUSH R5 ON STACK
(4) 005652 017746      MOV    @SWR,-(SP)       ;;PUSH @SWR ON STACK
(2) 005656 010637 005772      MOV    SP,$SAVR6        ;;SAVE SP
(2) 005662 012737 005674 000024      MOV    #SPWRUP,@#PWRVEC ;;SET UP VECTOR
(2) 005670 000000      HALT
(2) 005672 000776      BR     -2               ;;HANG UP
(2)
(3)
(2)
(2) 005674 012737 005766 000024      $PWRUP: MOV    #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(2) 005702 013706 005772      MOV    $SAVR6,SP        ;;GET SP
(2) 005706 005037 005772      CLR    $SAVR6           ;;WAIT LOOP FOR THE TTY
(2) 005712 005237 005772      1$:   INC    $SAVR6       ;;WAIT FOR THE INC
(2) 005716 001375      BNE    1$                ;;OF WORD
(4) 005720 012677 173360      MOV    (SP)+,@SWR        ;;POP STACK INTO @SWR
(4) 005724 012605      MOV    (SP)+,R5          ;;POP STACK INTO R5
(4) 005726 012604      MOV    (SP)+,R4          ;;POP STACK INTO R4
(4) 005730 012603      MOV    (SP)+,R3          ;;POP STACK INTO R3
(4) 005732 012602      MOV    (SP)+,R2          ;;POP STACK INTO R2
(4) 005734 012601      MOV    (SP)+,R1          ;;POP STACK INTO R1
(4) 005736 012600      MOV    (SP)+,R0          ;;POP STACK INTO R0
(2) 005740 012737 005622 000024      MOV    #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(2) 005746 012737 000300 000026      MOV    #PR6,@#PWRVEC+2  ;;PRIO:6
(2) 005754 104402      TYPE                                ;;REPORT THE POWER FAILURE
(2) 005756 005774      $PWRMG: .WORD    MPFAIL        ;;POWER FAIL MESSAGE POINTER
(2) 005760 012716      MOV    (PC)+,(SP)        ;;RESTART AT RESTART
(2) 005762 002536      $PW^AD: .WORD    RESTART       ;;RESTART ADDRESS
(2) 005764 000002      RTI
(2) 005766 000000      $SILLUP: HALT            ;;THE POWER UP SEQUENCE WAS STARTED
(2) 005770 000776      BR     -2                ;;BEFORE THE POWER DOWN WAS COMPLETE
(2) 005772 000000      $SAVR6: 0                ;;PUT THE SP HERE
(2) 005774 050200 051127 043040      MPFAIL: .ASCIZ    <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 006037 200 047105 020104      MEPASS: .ASCIZ    <200>/END PASS CNDZC-A /
(2) 006063 200 052522 047116      MR:     .ASCIZ    <200>/RUNNING /
(2) 006077 200 051120 043517      MERR2:  .ASCIZ    <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 006146 044600 051516 043125      MERR3:  .ASCIZ    <200>/INSUFFICIENT DATA!/
(2) 006172 046200 041517 020113      MLOCK:  .ASCIZ    <200>/LOCK ON SELECTED TEST/
(2) 006221 103 051123 020072      MCSRX:  .ASCIZ    /CSR: /
(2) 006227 126 041505 020072      MVECX:  .ASCIZ    /VEC: /
(2) 006235 120 051501 042523      MPASSX: .ASCIZ    /PASSES: /
(2) 006246 051105 047522 051522      MERRX:  .ASCIZ    /ERRORS: /
(2) 006257 124 051505 020124      MTSTN:  .ASCIZ    /TEST NO: /
(2) 006271 052 000040      MASTEK: .ASCIZ    /* /
(2) 006274 051600 052105 051440      MNEW:   .ASCIZ    <200>/SET SWITCH REG TO DZV11'S DESIRED ACTIVE./
(2) 006347 120 035103 000040      MERRPC: .ASCIZ    /PC: /
(2) 006354 046600 050101 047440      XHEAD:  .ASCIZ    <200>/MAP OF DZV11 STATUS/<200>
(2) 006402 044600 046114 043505      MBADLN: .ASCIZ    <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)
(2) 006444 000002      .EVEN
(2) 006446 006 003      XSTATQ: 2
      .BYTE 6,3
  
```

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09 PAGE 77-34
CNDZCA.P11 10-DEC-82 15:31 POWER DOWN AND UP ROUTINES

J 4

SEQ 0048

(2)	006450	001344			\$TMP1	
(2)	006452	006	002		.BYTE	6.2
(2)	006454	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) 006456 017605 000000 .SETFLG:MOV @ (SP),R5 ;PICK UP ADDRESS OF TAG
(2) 006462 042737 000040 007536 BIC #40,INBUF ;STRIP LOWER CASE
(2) 006470 122737 000105 007536 CMPB #'E,INBUF ;IS IT EXTERNAL LOOP BACK ?
(2) 006476 001005 BNE 4$ ;NO
(2) 006500 013715 006570 MOV 1$, (R5) ;YES STORE INFO
(2) 006504 105037 001424 CLRB MNTFLG ;SET MAINT BIT =0
(2) 006510 000422 BR 7$ ;GET OUT
(2) 006512 122737 000111 007536 4$: CMPB #'I,INBUF ;IS IT INTERNAL LOOP BACK ?
(2) 006520 001006 BNE 5$ ;NO
(2) 006522 013715 006572 MOV 2$, (R5) ;YES STORE INFO
(2) 006526 112737 000010 001424 MOVB #MAINT,MNTFLG ;SET UP THE MAINTENANCE FLAG LOADER
(2) 006534 000410 BR 7$ ;GET OUT
(2) 006536 122737 000123 007536 5$: CMPB #'S,INBUF ;IS IT STAGGERED LOOP BACK ?
(2) 006544 001007 BNE 6$ ;WHAT ?
(2) 006546 013715 006574 MOV 3$, (R5) ;YES STORE INFO
(2) 006552 105037 001424 CLRB MNTFLG ;ZERO BITS
(2) 006556 062716 000002 7$: ADD #2, (SP) ;POP AROUND
(2) 006562 000002 RTI
(2) 006564 104404 6$: INSTER ;RETRY
(2) 006566 000733 BR .SETFLG ;DIT 0
(2) 006570 000200 1$: .WORD 200 ;EXTERNAL = E
(2) 006572 000000 2$: .WORD 0 ;INTERNAL = I
(2) 006574 100000 3$: .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) 006576 017605 000000 .PAWCH: MOV @ (SP),R5
(2) 006602 142737 000040 007536 BICB #40,INBUF ;SET FOR LOWER CASE INPUT
(2) 006610 122737 000105 007536 CMPB #'E,INBUF ;IS IT 'E' ?
(2) 006616 001002 BNE 1$
(2) 006620 105015 CLRB (R5) ;000
(2) 006622 000406 BR 2$
(2) 006624 122737 000103 007536 1$: CMPB #'C,INBUF ;IS IT 'C' ?
(2) 006632 001005 BNE 3$
(2) 006634 112715 177777 MOVB #-1, (R5) ;3177
(2) 006640 062715 000002 2$: ADD #2, (SP)
(2) 006644 000002 RTI
(2) 006646 104404 3$: INSTER ;RETRY
(2) 006650 000752 BR .PAWCH

```

```

(2) ;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
(2) ;LINE NUMBER (SAVLIN) FOR DZVLPR, DZVTCR AND DZVCSR
(2) ;REGISTER USAGE.
(2)
(2) 006652 013737 001374 007070 SET: MOV SAVLIN,NUMLIN ;SAVE SAVLIN
(2) 006660 013700 001374 MOV SAVLIN,R0 ;COPY THE LINE NUMBER FOR LOOP CONTROL
(2) 006670 012702 000001 CLR NUMTCR ;SET A DEFAULT OF LINE 0 OR NO LINES
(2) 006674 005300 XTCCR1: MOV #1,R2 ;SET A BIT POINTER TO THE FIRST LINE
(2) 006676 100402 DEC R0 ;REDUCE THE INDICATOR.IS IT MINUS YET?
(2) 006700 006302 BMI SET1 ;IF SO, R2 POINTS TO THE RIGHT LINE
(2) 006702 000774 ASL R2 ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
(2) 006704 012701 006746 SET1: MOV #TABLE2,R1 ;GO SEE IF THIS LINE IS THE ONE
(2) 006710 010237 007072 MOV R2,NUMTCR ;COPY THE CORRECT BIT POINTER
(2) 006714 022137 007064 1$: CMP (R1)+,LINESP
(2) 006720 001407 BEQ 2$
(2) 006722 005721 TST (R1)+ ;IS IT THE END OF TABLE?
(2) 006724 001373 BNE 1$ ;NO
(2) 006726 104402 007174 TYPE ,MINVAL ;INVALID BAUD RATE,BEGIN AGAIN
(2) 006732 012705 002374 MOV #BAUD,R5 ;JUMP TO BAUD THRU R5
(2) 006736 000402 BR 3$
(2) 006740 011137 007066 2$: MOV (R1),SPEED ;SET UP BAUD RATE
(2) 006744 000205 3$: RTS R5
  
```

```

(2)
(2)
(2)
(2)
(2) TABLE2: ;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR)
(2) 006746 000062 .WORD 50 ;50 BAUD
(2) 006750 010070 .WORD 10070 ;
(2) 006752 000113 .WORD 75 ;75 BAUD
(2) 006754 010470 .WORD 10470 ;
(2) 006756 000156 .WORD 110 ;110 BAUD
(2) 006760 011070 .WORD 11070 ;TWO STOP BITS
(2) 006762 000207 .WORD 135 ;134.5 BAUD
(2) 006764 011470 .WORD 11470 ;TWO STOP BITS
(2) 006766 000226 .WORD 150 ;150 BAUD
(2) 006770 012070 .WORD 12070 ;TWO STOP BITS
(2) 006772 000454 .WORD 300 ;300 BAUD
(2) 006774 012430 .WORD 12430 ;ONE STOP BIT
(2) 006776 001130 .WORD 600 ;600 BAUD
(2) 007000 013030 .WORD 13030 ;ONE STOP BIT
(2) 007002 002260 .WORD 1200 ;1200 BAUD
(2) 007004 013430 .WORD 13430 ;ONE STOP BIT
(2) 007006 003410 .WORD 1800 ;1800 BAUD
(2) 007010 014030 .WORD 14030 ;ONE STOP BIT
(2) 007012 003720 .WORD 2000 ;2000 BAUD
(2) 007014 014430 .WORD 14430 ;ONE STOP BIT
(2) 007016 004540 .WORD 2400 ;2400 BAUD
(2) 007020 015030 .WORD 15030 ;ONE STOP BIT
(2) 007022 007020 .WORD 3600 ;3600 BAUD
(2) 007024 015430 .WORD 15430 ;ONE STOP BIT
(2) 007026 011300 .WORD 4800 ;4800 BAUD
(2) 007030 016030 .WORD 16030 ;ONE STOP BIT
(2) 007032 016040 .WORD 7200 ;7200 BAUD
(2) 007034 016430 .WORD 16430 ;ONE STOP BIT
(2) 007036 022600 .WORD 9600 ;9600 BAUD
  
```

```

(2) 007040 017030 .WORD 17030
(2) 007042 177777 000000 .WORD -1,0 ;TABLE TERMINATOR
(2)
(2) 007046 000000 WCHFLG:0 ;ECHO OR CABLE FLAG
(2) 007050 000000 STFLG: 0 ;PROGRAM START FLAG
(2) 007052 000000 LOCKUP: 0 ;TIMEOUT FLAG
(2) 007054 000000 LAST: 0 ;LAST ERROR PC
(2) 007056 000000 TDATA: 0
(2) 007060 000000 RDATA: 0
(2) 007062 000000 BYTCNT: 0
(2) 007064 000156 LINESP: 110. ;DEFAULT BAUD RATE
(2) 007066 011070 SPEED: 11070 ;DEFAULT 110 BAUD, 8 BITS/CHAR,
;FDX, 2 SJJP BITS
(2) 007070 000000 NUMLIN: 0
(2)
(2) 007072 000001 NUMTCR: 1 ;DEFAULT VALUE, TCR BIT 0
(2) 007074 000200 PRIO: 200 ;DEFAULT DEVICE PRIORITY
;MASK OUT INTERRUPTS
(2) 007076 000000 RECDAT: 0
(2) 007100 000000 TBUF: 0
(2) 007102 053200 041505 047524 MVECTO: .ASCIZ <200>/VECTOR ADDRESS- /
(2) 007124 041600 047117 051124 MREGAD: .ASCIZ <200>/CONTROL REGISTER ADDRESS- /
(2) 007160 050200 051501 020123 MPASS: .ASCIZ <200>/PASS DONE./
(2) 007174 044600 053116 046101 MINVAL: .ASCIZ <200>/INVALID BAUD RATE - /
(2) 007222 046200 047111 035105 MLINE: .ASCIZ <200>/LINE: /
(2) 007232 041200 052501 020104 MSPEED: .ASCIZ <200>/BAUD RATE - /
(2) 007250 052200 050131 020105 MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZV11 TERMINAL /
(2) 007311 200 044127 041511 MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /
(2) 007357 200 042524 046522 MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /
(2) 007404 041600 041101 042514 MCABLE: .ASCIZ <200>/CABLE TEST /
(2) 007421 377 177415 005377 MQUICK: .ASCII <377><15><377><377><12><377><377>
(2) 007430 044124 020105 052521 .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/
(2) 007525 377 177415 005377 .ASCII <377><15><377><377><12><377><377><377><0>
(2)
(2) .EVEN
(2)
(2) ;BUFFERS FOR INPUT-OUTPUT
(2)
(2) 007536 000000 INBUF: 0
(2) 007600 007600 .=.+40
(2) 007600 000000 TEMP: 0
(2) 007642 007042 .=.+40
(2) 007642 000000 MDATA: 0
(2) 007704 007704 .=.+40
  
```

```
(1) ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
(1) 007704 013700 002040 DZVLEV: MOV DZVRIV,RO ;PLACE THE BASE VECTOR ADDRESS IN RO
(1) 007710 062700 000002 ADD #2,RO ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
(1) 007714 010037 002042 MOV RO,DZV'IS ;STORE IT HERE
(1) 007720 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
(1) 007724 010037 002044 MOV RO,DZV'IV ;STORE IT HERE
(1) 007730 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
(1) 007734 010037 002046 MOV RO,DZ' 'S ;STORE IT HERE

(1) ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
(1) ;OF THE DEVICE
(1) 007740 013700 001174 MOV $BASE,RO ;COPY THE ADDRESS BEING LOADED
(1) 007744 010037 002010 MOV RO,DZVCSR ;XXX0
(1) 007750 005200 INC RO
(1) 007752 010037 002012 MOV RO,HDZVCSR ;XXX1
(1) 007756 005200 INC RO
(1) 007760 010037 002014 MOV RC,DZVRBUF ;XXX2
(1) 007764 010037 002020 MOV RO,DZVLPR ;XXX2
(1) 007770 005200 INC RO
(1) 007772 010037 002016 MOV RO,HDZVRBUF ;XXX3
(1) 007776 010037 002022 MOV RO,HDZVLPR ;XXX3
(1) 010002 005200 INC RO
(1) 010004 010037 002024 MOV RO,DZVT'CR ;XXX4
(1) 010010 005200 INC RO
(1) 010012 010037 002026 MOV RO,HDZVT'CR ;XXX5
(1) 010016 005200 INC RO
(1) 010020 010037 002030 MOV RO,DZVMSR ;XXX6
(1) 010024 010037 002034 MOV RO,DZVTDR ;XXX6
(1) 010030 005200 INC RO
(1) 010032 010037 002032 MOV RO,HDZVMSR ;XXX7
(1) 010036 010037 002036 MOV RO,HDZVTDR ;XXX7
(1) 010042 000207 RTS PC
```

```

7827
7828
7829
7830
7831
7832
7833 010044 104413
7834 010046 012737 000001 001246
7835 010054 013777 007072 171742
7836 010062 013737 007070 001370
7837 010070 053737 007066 001370
7838 010076 013777 001370 171714
7839 010104 012777 000040 171676
7840 010112 005004
7841 010114 012705 007421
7842 010120 005777 171664
7843 010124 100404
7844 010126 104414
7845 010130 005304
7846 010132 001372
7847 010134 104003
7848 010136 005004
7849 010140 112577 171670
7850 010144 001365
7851 010146 004737 005356
7852 010152 122777 000377 171124
7853 010160 001755
7854 010162 012737 002542 001362
7855 010170 012777 010244 171642
7856 010176 012777 000200 171636
7857 010204 106427 000000
7858 010210 012777 000140 171572
7859 010216 104402 007250
7860 010222 105777 171062
7861 010226 100375
7862 010230 106427 000200
7863 010234 004737 005356
7864 010240 000137 002412
7865
7866
7867
7868 010244 105777 171540
7869 010250 100401
7870 010252 104004
7871 010254 017737 171534 007076
7872 010262 100401
7873 010264 104023
7874 010266 032737 020000 007076
7875 010274 001401
7876 010276 104025
7877
7878 010300 113737 007076 007100
7879 010306 113737 007076 007536
7880 010314 042737 177600 007536
7881 010322 042737 176377 007076
7882 010330 000337 007076

:***** ECHO TEST *****
:*THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
:*(IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
:*ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)

TST1: DEVICE.CLR ;CLEAR DZV11
MOV #1,$TSTNM
MOV NUMTCR,@DZVTCR ;SET TCR BIT
MOV NUMLIN,PAR ;SET PARAMETERS
BIS SPEED,PAR ;SET BAUD RATE
MOV PAR,@DZVLPR ;LOAD PARAM.
MOV #MSENAB,@DZVCSR ;SET SCANN ENABLE
CLR R4
4$: MOV #MQUICK,R5 ;SET MESSAGE BUFFER
3$: TST @DZVCSR ;TRDY?
BMI 2$ ;BR IF YES
DELAY ;WAIT
DEC R4
BNE 3$
ERROR 3 ;NO TRDY SET! WHY?
2$: CLR R4 ;RESET COUNTER TO 0
MOVB (R5)+,@DZVTDR ;LOAD CHAR
BNE 3$
JSR PC,SERV.G ;<^G>?
CMPB #377,@SWR ;SWR SET TO 377?
BEQ 4$ ;IF YES LOOP ON QUICK MESSAGE
MOV #SEOP,NEXT
MOV #INTSVC,@DZVRIV ;SET UP INTERRUPT SERVICE
MOV #MASK,@DZVRIS ;AND LEVEL
MTPS #CLEAR ;ALLOW INTERRUPTS
MOV #RIE!MSENAB,@DZVCSR ;SET RECEIVER INTERRUPT ENABLE
TYPE ,MCHAR ;TYPE 'ANY CHARACTER'
1$: TSTB @STKS ;IF SOMEBODY HITS A KEY- GET NEW LINE #
BPL 1$ ;LOOP HERE
MTPS #MASK ;MASK FURTHER INTERRUPTS
JSR PC,SERV.G ;MAKE SURE IT WASN'T <^G>
JMP LINEX

;THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
INTSVC: TSTB @DZVCSR ;TEST REC. FLAG
BMI .+4
ERROR 4 ;ERROR - INTERRUPT NOT CAUSED BY FLAG
MOV @DZVRBUF,RECDAT
BMI .+4
ERROR 23 ;NON- VALID CHARACTER
BIT #BIT13,RECDAT ;CHECK FOR FRAMING ERROR
BEQ .+4 ;BR IF NO ERROR
ERROR 25 ;EITHER SOMEBODY HIT THE
;'BREAK KEY' OR YOU HAVE AN ERROR!
MOVB RECDAT,TBUF ;MOVE CHARACTER TO OUTPUT AREA
MOVB RECDAT,INBUF ;MOVE CHARACTER TO CHECK FOR ^C
BIC #^C<177>,INBUF ;STRIP JUNK PLUS PARITY
BIC #176377,RECDAT ;SAVE ONLY LINE NUMBER
SWAB RECDAT

```

```
7883 010334 023737 001374 007076      CMP    SAVLIN,RECDAT    ;DOES THE LINE # COMPARE?
7884 010342 001407                      BEQ    2$
7885 010344 013737 007076 001374      MOV    RECDAT,SAVLIN   ;ADJUST LINE NO. FOR ERROR
7886 010352 104015                      ERROR  15              ;*WRONG LINE NUMBER
7887 010354 013737 007070 001374      MOV    NUMLIN,SAVLIN  ;CORRECT LINE NO. INDICATOR
7888 010362 123727 007536 000003 2$:  CMPB  INBUF,#3         ;IS IT A ^C ?
7889 010370 001004                      BNE   1$              ;NO
7890 010372 104413                      DEVICE.CLR
7891 010374 012716 002542              MOV    #SEOP,(SP)     ;CRUNCH STACK
7892 010400 000002                      RTI
7893 010402 005777 171402              1$:  TST   @DZVCSR        ;TRDY SET
7894 010406 100375                      BPL   1$              ;IF NOT THEN WAIT
7895 010410 113777 007100 171416      MOVB  TBUF,@DZVTDR    ;TRANSMIT THE CHARACTER
7896 010416 000002                      RTI
7897
7898
7899                                     ;***** CABLE TEST *****
7900                                     ;*THIS TEST TRANSMITS A BINARY COUNT PATTERN
7901                                     ;*VIA INTERRUPT MODE TO THE RECEIVER
7902                                     ;*...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
7903
7904 010420 106427 000200 001246      TST2: MTPS  #MASK      ;DISABLE INTERRUPTS
7905 010424 012737 000002 001362      MOV   #2,$STSTM
7906 010432 012737 002542 001362      MOV   #SEOP,NEXT
7907 010440 104413                      DEVICE.CLR
7908                                     ;*TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
7909                                     ;*WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
7910                                     ;*JUMPERS W1,W2,W3 AND W4 MUST BE INSTALLED ON THE
7911                                     ;*INTERFACE MODULE OTHERWISE AN ERROR REPORT WILL RESULT.
7912 010442 012737 010450 001364      MOV   #1$,LOCK       ;LOOP
7913 010450 113777 007072 171350 1$:  MOVB  NUMTCR,@HDZVTCR ;SET DTR
7914 010456 005005                      CLR   R5
7915 010460 153705 007072              BISB  NUMTCR,R5       ;BUILD EXPECTED
7916 010464 000305                      SWAB  R5              ;PUT IN HIGH BYTE
7917 010466 153705 007072              BISB  NUMTCR,R5
7918 010472 104414                      DELAY
7919 010474 017704 171330              MOV   @DZVMSR,R4     ;READY MODEM BITS
7920 010500 020504                      CMP   R5,R4          ;ARE THEY OK?
7921 010502 001401                      BEQ   2$              ;BR IF YES
7922 010504 104022                      ERROR  22             ;IS THE TEST CONNECTOR ON?
7923                                     ;HAS RIGHT LINE BEEN SELECTED?
7924                                     ;IF SO- YOU HAVE A PROBLEM!
7925                                     ;MODEM BITS NOT RIGHT
7926 010506 104401 001364 001370      2$:  SCOP1
7927 010510 104413 001364 001370      3$:  DEVICE.CLR
7928 010512 005037 001364 001370      CLR   LOCK           ;CLEAR SCOP1 LOCK ADDRESS
7929 010516 013737 007066 001370      MOV   SPEED,PAR      ;SET LINE SPEED
7930 010524 053737 007070 001370      BIS   NUMLIN,PAR     ;SELECT LINE #
7931 010532 052737 010000 001370      BIS   #RCVON,PAR     ;ENABLE THE RECEIVER FOR THIS LINE
7932 010540 013777 001370 171252      MOV   PAR,@DZVLPR    ;SET THE PARAMETERS AND TURN ON RECEIVER
7933 010546 012777 010670 171264      MOV   #INTREC,@DZVRIV ;SET UP INTR SERVICE
7934 010554 012777 000200 171260      MOV   #MASK,@DZVRIS  ;SET UP LEVEL
7935 010562 012777 011060 171254      MOV   #INTRAN,@DZVTIV ;SET UP INTR SERVICE
7936 010570 012777 000200 171250      MOV   #MASK,@DZVTIS  ;SET UP LEVEL
7937 010576 012777 040140 171204      MOV   #TIE!RIE!MSENAB,@DZVCSR ;SET TRANSMITTER INTERRUPT ENABLE
7938 010604 105037 001425                      CLRB  DONFLG         ;INIT INTERRUPT DONE INDICATOR
```

```

7939 010610 005001 CLR R1 ;RX DATA POINTER- SET TO 0
7940 010612 005002 CLR R2 ;TX DATA POINTER- SET TO 0
7941 010614 013777 007072 171202 MOV NUMTCR,@DZVTCR ;SET UP TCR BIT
7942 010622 106427 000000 MTPS #C'EAR ;ALLOW INTERRUPTS
7943
7944 ;YOU RETURN HERE AFTER EVERY RECEIVER INTERRUPT
7945 010626 105777 170456 SPIN: TSTB @STKS ;IF SOMEBODY HITS A KEY- GET A NEW LINE #
7946 010632 100004 BPL 1$ ;BRANCH IF NO KEY HIT
7947 010634 004737 005356 JSR PC,SERV.G ;MAKE SURE IT WASN'T <^G>
7948 010640 000137 002412 JMP LINEX ;SW02=1
7949 010644 105737 001425 1$: TSTB DONFLG ;ARE ALL RECEIVER INTER. DONE
7950 010650 001004 BNE QUIT$ ;IF YES GET OUT OF TIMING LOOP
7951 010652 005237 007052 INC LOCKUP ;INC TIMEOUT FLAG
7952 010656 001363 BNE SPIN ;IF NOT 0 RETURN SPINNING
7953 010660 104011 ERROR 11 ;*RECEIVER FAILED TO INTERRUPT CHECK CABLE/TERMINATOR
7954 010662 104413 QUIT$: DEVICE.CLR
7955 010664 000137 002542 JMP $EOP ;CALL FOR END OF PASS
7956 010670 005037 007052 INTREC: CLR LOCKUP ;CLEAR TIMEOUT FLAG
7957 010674 105777 171110 TSTB @DZVCSR ;TEST REC DONE
7958 010700 100401 BMI .+4 ;YES
7959 010702 104004 ERROR 4 ;*FALSE INTERRUPT
7960 010704 017737 171104 007076 MOV @DZVRBUF,RECDAT ;SAVE WORD
7961 010712 100401 BMI .+4
7962 010714 104023 ERROR 23 ;*NON VALID CHARACTER
7963 010716 032737 040000 007076 BIT #BIT14,RECDAT ;DATA OVERRUN ?
7964 010724 001401 BEQ .+4 ;NO
7965 010726 104024 ERROR 24 ;*YES
7966 010730 032737 020000 007076 BIT #BIT13,RECDAT ;FRAMING ERROR ?
7967 010736 001401 BEQ .+4 ;NO
7968 010740 104025 ERROR 25 ;*YES
7969 010742 032737 010000 007076 BIT #BIT12,RECDAT ;PARITY ERROR ?
7970 010750 001401 BEQ .+4 ;NO
7971 010752 104026 ERROR 26 ;*YES
7972 010754 110105 MOV# R1,R5 ;SET EXPECTED
7973 010756 113704 007076 MOV# RECDAT,R4 ;GET FOUND
7974 010762 042704 177400 BIC #^C<377>,R4 ;CLEAR HIGH BYTE
7975 010766 042705 177400 BIC #^C<377>,R5 ;CLEAR HIGH BYTE
7976 010772 020504 CMP R5,R4 ;OK?
7977 010774 001401 BEQ .+4
7978 010776 104005 ERROR 5 ;DATA ERROR
7979 011000 042737 176377 007076 BIC #176377,RECDAT ;SAVE ONLY LINE NUMBER
7980 011006 000337 007076 SWAT RECDAT
7981 011012 023737 001374 007076 CMP SAVLIN,RECDAT ;DOES THE LINE # COMPARE ?
7982 011020 001407 BEQ 4$ ;YES
7983 011022 013737 007076 001374 MOV RECDAT,SAVLIN ;ADJUST LINE NO. FOR ERROR
7984 011030 104015 ERROR 15 ;*WRONG LINE #
7985 011032 013737 007070 001374 MOV NUMLIN,SAVLIN ;READJUST LINE NO.
7986 011040 120127 000377 4$: CMPB R1,#377 ;LAST CHARACTER ?
7987 011044 001003 BNE 1$ ;NO
7988 011046 105237 001425 INCB DONFLG ;INDICATE RECEIVER INTERRUPTS DONE
7989 011052 000401 BR 2$
7990 011054 105201 1$: INCB R1 ;UPDATE EXPECTED DATA
7991 011056 000002 2$: RTI
7992
7993 011060 005777 170724 INTRAN: TST @DZVCSR ;TEST TRANSMIT FLAG
7994 011064 100401 BMI .+4
  
```

7995	011066	104003			ERROR	3		; *FALSE INTERRUPT
7996	011070	110277	170740		MOVB	R2,@DZVTDR		; TRANSMIT A CHARACTER
7997	011074	105202			INCB	R2		; UPDATE TX DATA
7998	011076	001003			BNE	1\$; BIT PATTERN DONE?
7999	011100	042777	040000	170702	BIC	#TIE,@DZVCSR		; IF YES THEN CLEAR TIE
8000	011106	000002			RTI			; IF NOT THEN RETURN

1\$:

			;ERROR TABLE	
			.ERRTAB:	
8002				
8003	011110	000000	0	;ERROR 0
8004	011112	000000	0	
8005	011114	000000	0	
8006				
8007	011116	011336	EM1	;ERROR
8008	011120	012666	DH1	
8009	011122	013066	DT1	
8010				
8011	011124	011411	EM2	;ERROR 2
8012	011126	012712	DH2	
8013	011130	013100	DT2	
8014				
8015	011132	011437	EM3	;ERROR 3
8016	011134	012745	DH3	
8017	011136	013116	DT3	
8018				
8019	011140	011476	EM4	;ERROR 4
8020	0111 2	012745	DH3	
8021	011144	013116	DT3	
8022				
8023	011146	011525	EM5	;ERROR 5
8024	011150	012757	DH4	
8025	011152	013124	DT4	
8026				
8027	011154	011554	EM6	;ERROR 6
8028	011156	012757	DH4	
8029	011160	013124	DT4	
8030				
8031	011162	011613	EM7	;ERROR 7
8032	011164	012745	DH3	
8033	011166	013116	DT3	
8034				
8035	011170	011654	EM8	;ERROR 10
8036	011172	012745	DH3	
8037	011174	013116	DT3	
8038				
8039	011176	011716	EM9	;ERROR 11
8040	011200	012745	DH3	
8041	011202	013116	DT3	
8042				
8043	011204	011754	EM10	;ERROR 12
8044	011206	012745	DH3	
8045	011210	013116	DT3	
8046				
8047	011212	012013	EM13	;ERROR 13
8048	011214	012745	DH3	
8049	011216	013116	DT3	
8050				
8051	011220	012044	EM14	;ERROR 14
8052	011222	012745	DH3	
8053	011224	013116	DT3	
8054				
8055	011226	012076	EM15	;ERROR 15
8056	011230	000000	0	
8057	011232	000000	0	

8058				
8059	011234	012140	EM16	
8060	011236	012745	DH3	
8061	011240	013116	DT3	
8062				
8063	011242	012212	EM17	:ERROR 17
8064	011244	012745	DH3	
8065	011246	013116	DT3	
8066				
8067	011250	012250	EM20	
8068	011252	012745	DH3	
8069	011254	013116	DT3	
8070				
8071	011256	012311	EM21	:ERROR 21
8072	011260	013006	DH5	
8073	011262	013142	DT5	
8074				
8075	011264	012341	EM22	:ERROR 22
8076	011266	012757	DH4	
8077	011270	013124	DT4	
8078				
8079	011272	012403	EM23	:ERROR 23
8080	011274	012745	DH3	
8081	011276	013116	DT3	
8082				
8083	011300	012433	EM24	
8084	011302	012745	DH3	
8085	011304	013116	DT3	
8086				
8087	011306	012461	EM25	
8088	011310	012745	DH3	
8089	011312	013116	DT3	
8090				
8091	011314	012511	EM26	
8092	011316	012745	DH3	
8093	011320	013116	DT3	
8094				
8095	011322	012540	EM27	
8096	011324	012745	DH3	
8097	011326	013116	DT3	
8098				
8099	011330	012606	EM30	
8100	011332	012745	DH3	
8101	011334	013116	DT3	

```

8103          :ERROR MESSAGES
8107 011336 047200 020117 052502 EM1: .ASCIZ <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
8108 011411      200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE?
8109 011437      200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
8110 011476 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
8111 011525      200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
8112 011554 042200 053132 030461 EM6: .ASCIZ <200>/DZV11 *RECEIVER BUFFER* ERROR/
8113 011613      200 051124 047101 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
8114 011654 052600 042516 050130 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
8115 011716 051200 041505 044505 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
8116 011754 052600 042516 050130 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
8117 012013      200 044523 047514 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
8118 012044 051600 046111 020117 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
8119 012076 040600 052103 047511 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
8120 012140 051200 040505 044504 EM16: .ASCIZ <200>/READING DZVRBUF DID NOT CLEAR SILO ALARM/
8121 012212 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
8122 012250 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
8123 012311      200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
8124 012341      200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
8125 012403      200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
8126 012433      200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
8127 012461      200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
8128 012511      200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
8129 012540 051600 046111 020117 EM27: .ASCIZ <200>/SILO ALARM FAILED TO CAUSE INTERRUPT/
8130 012606 046200 047111 020105 EM30: .ASCIZ <200>/LINE DID NOT RECEIVE FULL BINARY COUNT PATTERN/
8131
8132 012666 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZV11 REG/
8133 012712 042600 050130 041505 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
8134 012745      200 044514 042516 DH3: .ASCIZ <200>/LINE NO./
8135 012757      200 054105 042520 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
8136 013006 052200 020130 044514 DH5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/
8137
8138          013066          .EVEN
8142
8143 013066 000002          :DATA TABLES FOR ERROR MESSAGES
8144 013070      006      003 DT1: 2
8145 013072 001330          .BYTE 6,3
8146 013074      006      001 $REG1
8147 013076 001326          .BYTE 6,1
8148          $REG0
8149 013100 000003          DT2: 3
8150 013102      006      004 .BYTE 6,4
8151 013104 001340          $REG5
8152 013106      006      001 .BYTE 6,1
8153 013110 001336          $REG4
8154 013112      006      001 .BYTE 6,1
8155 013114 001326          $REG0
8156
8157 013116 000001          DT3: 1
8158 013120      003      001 .BYTE 3,1
8159 013122 001374          SAVLIN
8160
8161 013124 000003          DT4: 3
8162 013126      006      004 .BYTE 6,4
8163 013130 001340          $REG5
8164 013132      006      001 .BYTE 6,1
  
```

8165 013134 001336
8166 013136 003 001
8167 013140 001374
8168
8169 013142 000004
8170 013144 003 005
8171 013146 001374
8172 013150 006 011
8173 013152 001340
8174 013154 006 007
8175 013156 001344
8176 013160 006 001
8177 013162 001402

\$REG4
.BYTE 3,1
SAVLIN
DTS: 4
.BYTE 3,5
SAVLIN
.BYTE 6,9.
\$REG5
.BYTE 6,7
\$TMP1
.BYTE 6,1
REGIST

;TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

8180
8181 013164 002450
8182 013166 001560
8183 013170 001120
8184 013172 000750
8185 013174 000660
8186 013176 000330
8187 013200 000150
8188 013202 000060
8189 013204 000040
8190 013206 000030
8191 013210 000020
8192 013212 000010
8193 013214 000001
8194 013216 000001
8195 013220 000001
8196 013222 000001

DLYTBL: 2450
1560
1120
750
660
330
150
60
40
30
20
10
1
1
1
1
1

;TIME FOR 50 BAUD
;TIME FOR 75 BAUD
;TIME FOR 110 BAUD
;TIME FOR 134 BAUD
;TIME FOR 150 BAUD
;TIME FOR 300 BAUD
;TIME FOR 600 BAUD
;TIME FOR 1200 BAUD
;TIME FOR 1800 BAUD
;TIME FOR 2000 BAUD
;TIME FOR 2400 BAUD
;TIME FOR 3600 BAUD
;TIME FOR 4800 BAUD
;TIME FOR 7200 BAUD
;TIME FOR 9600 BAUD
;TIME OF DELAY FOR 19200 BAUD

8197
8198
8199

;DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
;FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.

```
8277 ; INIT $BASE AND $VECT1 AND TWEAK THE '$GETPAR' CALLING
8278 ; SEQUENCE TO ACCEPT THE VALID FALCON RANGE.
8279
8280 013224 023727 001174 160010 FALCINI: CMP $BASE,#ABASE ; IS $BASE VIRGIN ?? ;:GPA
8281 013232 001003 BNE 1$ ; SKIP NEXT IF NOT ;:GPA
8282 013234 012737 174040 001174 MOV #174040,$BASE ; YES, SET ENGINEERING DEFAULT ;:GPA
8283 013242 023727 001170 000000 1$: CMP $VECT1,#AVECT1 ; IS $VECT1 VIRGIN ?? ;:GPA
8284 013250 001003 BNE 2$ ; SKIP NEXT IF NOT ;:GPA
8285 013252 012737 000370 001170 MOV #370,$VECT1 ; YES, SET ENGINEERING DEFAULT ;:GPA
8286 013260 012737 013324 002326 2$: MOV #3$,GETCSR+2 ; SUBSTITUTE CSR TEXT... ;:GPA
8287 013266 012737 174000 002332 MOV #174000,GETCSR+6 ;:GPA
8288 013274 012737 177770 002334 MOV #177770,GETCSR+10 ;...AND VALID RANGE. ;:GPA
8289 013302 012737 013366 002344 MOV #4$,GETVEC+2 ; SUBSTITUTE VECTOR TEXT... ;:GPA
8290 013310 005037 002350 CLR GETVEC+6 ;:GPA
8291 013314 012737 000370 002352 MOV #370,GETVEC+10 ;...AND VALID RANGE. ;:GPA
8292 013322 000207 RETURN ; RETURN TO CALLER. ;:GPA
8293
8294 013324 030600 052123 041440 3$: .ASCIZ <200>'1ST CSR ADDRESS (174000:177770) ' ;:GPA
013332 051123 040440 042104
013340 042522 051523 024040
013346 033461 030064 030060
013354 030472 033467 033467
013362 024460 000040
8295 013366 030600 052123 053040 4$: .ASCIZ <200>'1ST VECTOR ADDRESS (000:370) ' ;:GPA
013374 041505 047524 020122
013402 042101 011104 051505
013410 020123 030050 030060
013416 031472 030067 020051
013424 020040 000040
8296 .EVEN ;:GPA
8297
8298 ;$FREE= <1000-.>/2 ; FREE WORDS LEFT. ;:GPA
8299 .IF LT $FREE ;:GPA
8300 .ERROR ; VECTOR OVERLAY EXCEEDED ;:GPA
8301 .ENDC ;:GPA
8302
8303 .=$SVPC ;:GPA
8304 013430
8308 CORMAX:
8309 ;:THE FOLLOWING CALL TO CNMAC2.SML WAS ADDED TO INIT BRKVEC AND LKVEC
POINT=. ;SAVE POINTER
(1) 000100 013430 .=100
(1) 000102 000300 $CLKVEC ;LKVEC HANDLER
(1) 000140 000140 300 ;INTERRUPT HANDLER PRI
(1) 000140 170000 .=140 ;BRKVEC
(1) 000142 000300 170000 ;ODT START ADDRESS
(1) 013430 013430 300 ;PRIORITY
(1) 013430 104402 013436 .=POINT ;RESTORE POINTER
(1) 013434 000000 $CLKVEC: TYPE,CLKMES
(1) 013436 005015 045514 042526 CLKMES: .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
(1) 013444 020103 047111 042524 HALT
(1) 013452 051122 050125 020124
(1) 013460 020055 044504 041523
(1) 013466 047117 042516 052103
(1) 013474 046040 041524 000040
8310 013502 000001 KXTFLAG:1
```

CNDZC-A MACY11 30(1046) 14-DEC-82 10:09 PAGE 80-1

CNDZCA.P11 10-DEC-82 15:31

DZV11 DEVICE DIAGNOSTICS.

K 5

COPYRIGHT 1982 DIGITAL EQUIP. CORP.

SEQ 0062

8311

000001

.END

ABASE = 160010	7810#	7821	8280
ACDW1 = 000000	7821		
ACDW2 = 000000	7821		
ACPUOP= 000000	7821		
ACTIVE 001420	7821#		
ADDW0 = 000000	7821		
ADDW1 = 000000	7821		
ADDW10= 000000	7821		
ADDW11= 000000	7821		
ADDW12= 000000	7821		
ADDW13= 000000	7821		
ADDW14= 000000	7821		
ADDW15= 000000	7821		
ADDW2 = 000000	7821		
ADDW3 = 000000	7821		
ADDW4 = 000000	7821		
ADDW5 = 000000	7821		
ADDW6 = 000000	7821		
ADDW7 = 000000	7821		
ADDW8 = 000000	7821		
ADDW9 = 000000	7821		
ADEVCT= 000000	7821		
ADEVN = 000000	7821		
ADRCNT 004245	7823#*		
ADVANC= 104400	7821#	7823	
AENV = 000000	7821		
AENV# = 000000	7821		
AFATAL= 000000	7821		
AMADR1= 000000	7821		
AMADR2= 000000	7821		
AMADR3= 000000	7821		
AMADR4= 000000	7821		
AMAMS1= 000000	7821		
AMAMS2= 000000	7821		
AMAMS3= 000000	7821		
AMAMS4= 000000	7821		
AMSGAD= 000000	7821		
AMSGLG= 000000	7821		
AMSGTY= 000000	7821		
AMTYP1= 000000	7821		
AMTYP2= 000000	7821		
AMTYP3= 000000	7821		
AMTYP4= 000000	7821		
APASS = 000000	7821		
APRIOR= 000000	7821		
APTCSU= 000040	7823#		
APTENV= 000001	7823#		
APTSIZ= 000200	7823#		
APTSP0= 000100	7823#		
ASWREG= 000000	7821		
ATESTN= 000000	7821		
AUNIT = 000000	7821		
AJSWR = 000000	7821		
AVECT = 000300	7811#		
AVECT1= 000000	7821	8283	
AVECT2= 000000	7821		

BAUD	002374	7821#	7823						
BINWRD	004520	7823#							
BIT0	= 000001	7821#							
BIT00	= 000001	7821#							
BIT01	= 000002	7821#							
BIT02	= 000004	7821#							
BIT03	= 000010	7821#							
BIT04	= 000020	7821#							
BIT05	= 000040	7821#							
BIT06	= 000100	7821#							
BIT07	= 000200	7821#							
BIT08	= 000400	7821#							
BIT09	= 001000	7821#							
BIT1	= 000002	7821#							
BIT10	= 002000	7821#							
BIT11	= 004000	7821#							
BIT12	= 010000	7821#	7969						
BIT13	= 020000	7821#	7874	7966					
BIT14	= 040000	7821#	7963						
BIT15	= 100000	7821#							
BIT2	= 000004	7821#							
BIT3	= 000010	7821#							
BIT4	= 000020	7821#	7823						
BIT5	= 000040	7821#							
BIT6	= 000100	7821#							
BIT7	= 000200	7821#	7823						
BIT8	= 000400	7821#							
BIT9	= 001000	7821#							
BPTVEC	= 000014	7821#							
BRKVEC	= 000140	7821#							
BRK0	= 000400	7821#							
BRK1	= 001000	7821#							
BRK2	= 002000	7821#							
BRK3	= 004000	7821#							
BUFSET	= 104422	7821#							
BYTCNT	007062	7823#							
CHRCNT	004516	7823#*							
CLEAR	= 000000	7821#	7857*	7942*					
CLKMES	013436	8309#							
CNVRT	= 104412	7821#	7823						
CONVRT	= 104411	7821#	7823						
CORMAX	013430	8304#	8305						
CO0	= 000400	7821#							
CO1	= 001000	7821#							
CO2	= 002000	7821#							
CO3	= 004000	7821#							
CR	= 000015	7821#	7823						
CRLF	= 000200	7821#	7823						
DATABP	005212	7823#*							
DATAHD	005200	7823#*							
DCLASM	= 104417	7821#							
DCLR	= 000020	7821#	7823						
DDISP	= 177570	7821#							
DELAY	= 104414	7821#	7844	7918					
DEVADR	004242	7823#*							
DEVICE	= 104413	7821#	7823	7833	7890	7907	7927	7954	

DZVC7	001610	7821#				
DZVLEV	007704	7821	7823#			
DZVLPR	002020	7821#	7823*	7838*	7932*	
DZVMSR	002030	7821#	7823*	7919		
DZVNUM	001414	7821#				
DZVRBU	002014	7821#	7823*	7871	7960	
DZVRIS	002042	7821#	7823*	7856*	7934*	
DZVRIV	002040	7821#	7823	7855*	7933*	
DZVTCR	002024	7821#	7823*	7835*	7941*	
DZVTDR	002034	7821#	7823*	7849*	7895*	7996*
DZVTIS	002046	7821#	7823*	7936*		
DZVTIV	002044	7821#	7823*	7935*		
DZV.EN	001740	7821#				
DZV.MA	001500	7821#				
EIGHT =	000030	7821#				
EIGHTS=	000070	7821#				
EMTVEC=	000030	7821#				
EM1	011336	8007	8107#			
EM10	011754	8043	8116#			
EM13	012013	8047	8117#			
EM14	012044	8051	8118#			
EM15	012076	8055	8119#			
EM16	012140	8059	8120#			
EM17	012212	8063	8121#			
EM2	011411	8011	8108#			
EM20	012250	8067	8122#			
EM21	012311	8071	8123#			
EM22	012341	8075	8124#			
EM23	012403	8079	8125#			
EM24	012433	8083	8126#			
EM25	012461	8087	8127#			
EM26	012511	8091	8128#			
EM27	012540	8095	8129#			
EM3	011437	8015	8109#			
EM30	012606	8099	8130#			
EM4	011476	8019	8110#			
EM5	011525	8023	8111#			
EM6	011554	8027	8112#			
EM7	011613	8031	8113#			
EM8	011654	8035	8114#			
EM9	011716	8039	8115#			
ERRMSG	005166	7823#*				
ERRVEC=	000004	7821#				
ERTABO	005342	7823#				
EVEPAR=	000000	7821#				
EXITER	005272	7823#				
FALCIN	013224	7821*	8280#			
FIVE =	000000	7821#				
FIVES =	000040	7821#				
FRMERR=	020000	7821#				
GETCSR=	002324	7821#	8286*	8287*	8288*	
GETSWR=	005404	7821*	7823#			
GETVEC=	002342	7821#	8289*	8290*	8291*	
HALTS	005216	7823#				
HDRFLG	001423	7821#				
HDZVCS	002012	7821#	7823*			

HDZVLP	002022	7821#	7823*		
HDZVMS	002032	7821#	7823*		
HDZVRB	002016	7821#	7823*		
HDZVTC	002026	7821#	7823*	7913*	
HDZVTD	002036	7821#	7823*		
H'LIM	004240	7823#*			
HT =	000011	7821#	7823		
INBUF	007536	7823#*	7879*	7880*	7888
INIFLG	001422	7821#*			
INSTER=	104404	7821#	7823		
INSTR =	104403	7821#			
INSTR2	004040	7823#			
INTRAN	011060	7935	7993#		
INTREC	010670	7933	7956#		
INTSVC	010244	7855	7868#		
IOTVEC=	000020	7821#			
KXTFLA	013502	7821	8310#		
LAST	007054	7821*	7823#		
LF =	000012	7821#	7823		
LIMITS	004166	7823#			
LINE	001366	7821#			
LINESP	007064	7821	7823#		
LINEX	002412	7821#	7864	7948	
LINE0	001504	7821#			
LINE1	001516	7821#			
LINE10	001624	7821#			
LINE11	001636	7821#			
LINE12	001650	7821#			
LINE13	001662	7821#			
LINE14	001674	7821#			
LINE15	001706	7821#			
LINE16	001720	7821#			
LINE17	001732	7821#			
LINE2	001530	7821#			
LINE3	001542	7821#			
LINE4	001554	7821#			
LINE5	001566	7821#			
LINE6	001600	7821#			
LINE7	001612	7821#			
LKVEC =	000100	7821#			
LOBITS	004244	7823#*			
LOCK	001364	7821#	7823*	7912*	7928*
LOCKUP	007052	7821*	7823#	7951*	7956*
LOLIM	004236	7823#*			
LPRSET=	104421	7821#			
LPO =	000000	7821#			
LP1 =	000001	7821#			
LP2 =	000002	7821#			
LP3 =	000003	7821#			
MAINT =	000010	7821#	7823		
MANT0	001510	7821#			
MANT1	001522	7821#			
MANT10	001630	7821#			
MANT11	001642	7821#			
MANT12	001654	7821#			
MANT13	001666	7821#			

SW8	=	000400	7821#				
SW9	=	001000	7821#				
S110	=	001000	7821#				
S1200	=	003400	7821#				
S134	=	001400	7821#				
S150	=	002000	7821#				
S1800	=	004000	7821#				
S19200	=	007400	7821#				
S2000	=	004400	7821#				
S2400	=	005000	7821#				
S300	=	002400	7821#				
S3600	=	005400	7821#				
S4800	=	006000	7821#				
S50	=	000000	7821#				
S600	=	003000	7821#				
S7200	=	006400	7821#				
S75	=	000400	7821#				
S9600	=	007000	7821#				
TABLE2		006746	7823#				
TBITVE	=	000014	7821#				
TBUF		007100	7823#	7878*	7895		
TCR0	=	000001	7821#				
TCR1	=	000002	7821#				
TCR2	=	000004	7821#				
TCR3	=	000010	7821#				
TDATA		007056	7823#				
TD0		001426	7821#	7823			
TD1		001430	7821#				
TD2		001432	7821#				
TD3		001434	7821#				
TEIGHT		002106	7821#				
TEMP		007600	7823#				
TFIVE		002114	7821#				
TIE	=	040000	7821#	7937	7999		
TKVEC	=	000060	7821#				
TLO	=	000000	7821#				
TL1	=	000400	7821#				
TL2	=	001000	7821#				
TL3	=	001400	7821#				
TMTBL		002050	7821#				
TPVEC	=	000064	7821#				
TRAPVE	=	000034	7821#				
TRDY	=	100000	7821#				
TRTVEC	=	000014	7821#				
TRO		001436	7821#				
TR1		001440	7821#				
TR2		001442	7821#				
TR3		001444	7821#				
TSEVEN		002110	7821#				
TSIX		002112	7821#				
TST1		010044	7821	7833#			
TST2		010420	7821	7904#			
TWOSTO	=	000040	7821#				
TYPDAT		005202	7823#				
TYPE	=	104402	7821#	7823	7859	8309	
TYPMSG		005072	7823#				

T110	002054	7821#			
T1200	002066	7821#			
T134	002056	7821#			
T150	002060	7821#			
T1800	002070	7821#			
T2000	002072	7821#			
T2400	002074	7821#			
T300	002062	7821#			
T3600	002076	7821#			
T4800	002100	7821#			
T50	002050	7821#			
T600	002064	7821#			
T7200	002102	7821#			
T75	002052	7821#			
T9600	002104	7821#			
VEC1	002264	7821#			
WCHFLG	007046	7821	7823#		
WRDCNT	004514	7823#*			
WTBS.F	005170	7823#			
XBEGIN	002442	7821#	7823		
XBX	004760	7823#			
XCSR	002704	7823#			
XERR	002726	7823#			
XHEAD	006354	7823#			
XMTCNT	001400	7821#			
XMTLIN	001376	7821#			
XPASS	002720	7823#			
XSTATQ	006444	7823#			
XTCR1	006674	7823#			
XTSTN	005350	7823#			
XVEC	002712	7823#			
XX =	160210	7821#			
YY =	000500	7821#			
ZZ =	000020	7821#			
\$APTHD	001446	7821#			
\$ASTAT=	***** U	7823			
\$ATYC	003500	7823#			
\$ATY1	003454	7823#			
\$ATY3	003462	7823#			
\$ATY4	003472	7823#			
\$AUTOB	001300	7821#			
\$BASE	001174	7821#	7823	8280	8282*
\$BDADR	001266	7821#			
\$BDDAT	001272	7821#			
\$CDW1	001200	7821#			
\$CDW2	001202	7821#			
\$CHARC	003450	7823#*			
\$CLKVE	013430	8309#			
\$CMTAG	001244	7821#			
\$CM1 =	000006	7821#			
\$CM2 =	000014	7821#			
\$CM3 =	000006	7821#			
\$CM4 =	000005	7821#			
\$CPUOP	001146	7821#			
\$CRLF	001357	7821#	7823		
\$DDW0	001204	7821#			

\$DDW1	001206	7821#				
\$DDW10	001230	7821#				
\$DDW11	001232	7821#				
\$DDW12	001234	7821#				
\$DDW13	001236	7821#				
\$DDW14	001240	7821#				
\$DDW15	001242	7821#				
\$DDW2	001210	7821#				
\$DDW3	001212	7821#				
\$DDW4	001214	7821#				
\$DDW5	001216	7821#				
\$DDW6	001220	7821#				
\$DDW7	001222	7821#				
\$DDW8	001224	7821#				
\$DDW9	001226	7821#				
\$DEVCT	001130	7821#				
\$DEVN	001176	7821#				
\$DOAGN	002700	7823#				
\$E =	000002	6667#				
\$ENDAD	002670	7821	7823#			
\$ENDCT	002654	7823#				
\$ENV	001140	7821#	7823			
\$ENVN	001141	7821#	7823			
\$EOP	002542	7823#	7854	7891	7906	7955
\$EOPCT	002646	7823#				
\$ERFLG	001247	7821#*	7823*			
\$ERMAX	001261	7821#				
\$ERROR	004730	7821	7823#			
\$ERRPC	001262	7821#	7823*			
\$ERRTB	001362	7821#				
\$ERTTL	001256	7821#*	7823*			
\$ETABL	001140	7821#				
\$ETEND	001244	7821#				
\$FATAL	001122	7821#	7823*			
\$FFLG	003720	7823#*				
\$FILLC	001322	7821#	7823			
\$FILLS	001321	7821#	7823			
\$FLIP =	177777	5676#				
\$GDADR	001264	7821#				
\$GDDAT	001270	7821#				
\$GET42	002660	7823#				
\$HD =	000001	7821				
\$HIBTS	001446	7821#				
\$ICNT	001250	7821#				
\$ILLUP	005766	7823#				
\$INTAG	001301	7821#				
\$ITEMB	001260	7821#	7823*			
\$LF	001360	7821#	7823			
\$LFLG	003717	7823#*				
\$LPADR	001252	7821#*	7823*			
\$LPERR	001254	7821#				
\$MADR1	001152	7821#				
\$MADR2	001156	7821#				
\$MADR3	001162	7821#				
\$MADR4	001166	7821#				
\$MAIL	001120	7821#	7823			

\$MAMS1	001150	7821#				
\$MAMS2	001154	7821#				
\$MAMS3	001160	7821#				
\$MAMS4	001164	7821#				
\$MBADR	001450	7821#				
\$MFLG	003716	7823#*				
\$MSGAD	001134	7821#	7823*			
\$MSGLG	001136	7821#	7823*			
\$MSGTY	001120	7821#	7823*			
\$MTYP1	001151	7821#				
\$MTYP2	001155	7821#				
\$MTYP3	001161	7821#				
\$MTYP4	001165	7821#				
\$N =	000001	6666#				
\$NULL	001320	7821#	7823			
\$PASS	001126	7821#*	7823*			
\$PASTM	001454	7821#				
\$PWRAD	005762	7823#				
\$PWRDN	005622	7821	7823#			
\$PWRMG	005756	7823#				
\$PWRUP	005674	7823#				
\$QUES	001356	7821#	7823			
\$REGAD	001324	7821#				
\$REG0	001326	7821#	7823*	8147	8155	
\$REG1	001330	7821#	7823*	8145		
\$REG2	001332	7821#	7823*			
\$REG3	001334	7821#	7823*			
\$REG4	001336	7821#	7823*	8153	8165	
\$REG5	001340	7821#	7823*	8151	8163	8173
\$RTNAD	002702	7823#				
\$SAVR6	005772	7823#*				
\$SETUP=	000000	7823				
\$SVPC =	000040	7821#				
\$SWR =	164000	5683#	7821	7823		
\$SWREG	001142	7821#				
\$TESTN	001124	7821#				
\$TIMES	001354	7821#	7823*			
\$TKB	001312	7821#	7823			
\$TKS	001310	7821#	7823	7860	7945	
\$TMP0	001342	7821#				
\$TMP1	001344	7821#	7823	8175		
\$TMP2	001346	7821#	7823			
\$TMP3	001350	7821#				
\$TMP4	001352	7821#				
\$TN =	000001	7821#				
\$TPB	001316	7821#	7823*			
\$TPFLG	001323	7821#	7823			
\$TPS	001314	7821#	7823			
\$TSTM	001452	7821#				
\$TSTNM	001246	7821#*	7823	7834*	7905*	
\$TYPE	003172	7823#				
\$TYPEC	003404	7823#				
\$TYPEX	003452	7823#				
\$UNIT	001132	7821#				
\$UNITM	001456	7821#				
\$USWR	001144	7821#				

.\$ACT1	5064#	5682#	7821
.\$APT8	5109#	5682#	7821#
.\$APTH	5370#	5682#	7821
.\$APTY	5547#	5682#	7823
.\$ASTA	5417#		
.\$CATC	932#	5680#	
.\$CMTA	1047#	7821#	
.\$DB2D	4686#		
.\$DB2O	4812#		
.\$DIV	4587#		
.\$EOP	2214#	5680#	7823
.\$ERRO	2700#	5681#	
.\$ERRT	2896#		
.\$MULT	4523#		
.\$POWE	4229#	5681#	7823
.\$RAND	4307#		
.\$RDDE	3891#		
.\$RDOC	3797#		
.\$READ	3395#		
.\$R2AZ	4958#		
.\$SAVE	3969#		
.\$SB2D	4771#		
.\$SB2O	4874#		
.\$SCOP	2454#		
.\$SIZE	4361#		
.\$SUPR	4913#		
.\$TRAP	4073#	5681#	
.\$TYPB	3287#		
.\$TYPD	3209#		
.\$TYPE	2985#	5680#	7823
.\$TYPO	3112#		
.\$40CA	972#		

. ABS. 013504 000

ERRORS DETECTED: 0

CNDZCA,CNDZCA/CRF/NL:TOC=CNMAC2.SML,CNDZCA.P11

RUN-TIME: 13 13 1 SECONDS

RUN-TIME RATIO: 83/28=2.9

CORE USED: 50K (100 PAGES)