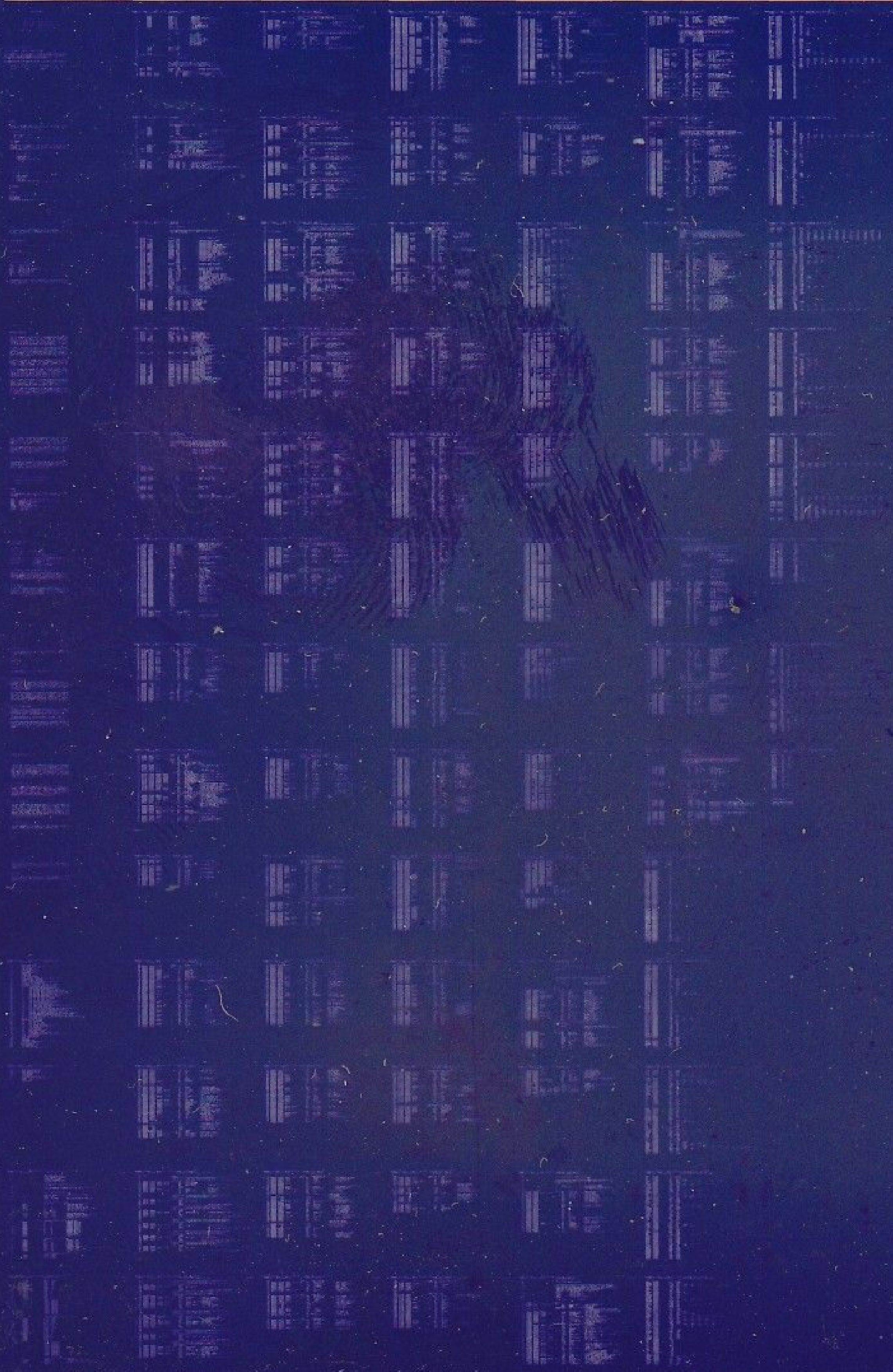
ADV11-A

ADV11 PERF TEST
CVADACO

AH-8175C-MC
FICHE 1 OF 1

FEB 1981
COPYRIGHT© 76-80
MADE IN USA



IDENTIFICATION

Product Code: AC-8174C-MC

Diagnostic Code: MAINDEC-11-CVADA-C

Product Name: CVADACO ADV11 Perf Test

Date: Aug. 8, 1980

Maintainer: Diagnostic Group

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.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by digital or its affiliated companies.

The following are trademarks of Digital Equipment Corporation:

```
DIGITAL    PDP      UNIBUS     MASSBUS
DL         DECUS    DECTAPE
```

1.0 ABSTRACT

Version "C" includes the patch to correct an error in the differential linearity test. Version "C" also allows the program to execute properly on an "F-11" cpu.

This diagnostic has two starting addresses:

200 standard tolerances
204 restart
210 tighter tolerances for the option test area's burn in.

This diagnostic tests the ADV11 with or without the test connector.

When starting the diagnostic, a set of tests is listed and this statement is printed out: "Type the letter and carriage return of the desired test:". The following chart indicates which letter corresponds to which test:

W: The entire Wraparound test (requires test connector)

a. Analog subtests
b. Noise test
c. Interchannel Settling test
d. Differential Linearity and Relative Accuracy test

C: Calibration test only

P: Print values test only

L: Logic Subtests only

A: Auto test (requires test connector)

A. Logic subtests
ts B. Analog subtes
C. Noise Test
D. Interchannel Settling Test
E. Differential Linearity and Relative Accuracy Test

2.0 REQUIREMENTS

2.1 Equipment

LSI-11 or F-11 computer with 8K of memory
I/O Terminal
ADV11 Module
VT55 OR VT105 Terminal supported for graphic output (optional)
Test connector (70-12894)

2.2 Storage

This program uses all 8K of memory and is not "chainable" on an 8K CPU. The orogram is "chainable" on 12K or greater CPU. The program will destroy "absolute loader" on an 8K CPU, if "W" or "A" is selected.

3.0 LOADING PROCEDURE

Procedure for loading normal binary tapes should be followed.

4.0 STARTING PROCEDURE

4.1 Control Switch Settings

Standard PDP-11 Format

SW15=1	Halt on error
SW14=1	Loop on test
SW13=1	Inhibit error typeouts
SW12=1	Halt for VT55/VT105 GRAPHIC DISPLAY TERMINAL
SW11=1	Inhibit iterations
SW10=1	Bell on error
SW9 =1	Loop on error
SW8 =1	Loop on test in SWR <7:0>

Location 200 is the starting address of the diagnostic for standard tolerances. 204 is the restart address. 210 is the starting address of the diagnostic for the option test area's burn in test.

5.0 OPERATING PROCEDURE

Start the diagnostic at 200 or 210. The program heading and the list of tests available, will be printed out followed by a message "Type letter and <CR> for test:". Then type the letter you want, according to the table listed and depress return. If started at the option test area's starting address, the program will not ask for the test but will run the logic test.

Two control characters, ^A and ^C, are set aside for interrupting a test and transferring control to either the beginning of the diagnostic (^C) or to the beginning of the specific test which was in progress (^A). During the logic tests while a reset is being performed, ^C or ^A will not be executed until after the reset has been completed, therefore continue typing ^C or ^A until it is successful.

For machines without a hardware switch register, location SWREG (176) is used as a software switch register. To modify the contents of SWREG, type ^G. The program responds with the current contents of SWREG and a slash. Type the desired new contents of SWREG followed by a carriage return.

If "W" is typed, the program will type "XX ADV11's FOUND". Where XX is the number of ADV11's in octal. If the number is greater than 1, the test will be run successively on each ADV11. The program will run through the analog subtests, the Noise test, the Interchannel Settling test, and the Differential Linearity and Relative Accuracy test. The Test connector is required.

If "C" is typed, the program will run the calibration routine and loop on the test until it is calibrated and a carriage return typed. If a certain ADV11 is to be calibrated, its status register address must be loaded into $BASE (1250), and its vector address must be loaded into the low byte of $VECT1 (1244).

If "P" is typed, the program will run the print values routine and will loop on that test until the operator halts it. If a certain ADV11 is to be tested, its status register address must be loaded into $BASE (1250), and its vector address must be loaded into the low byte of $VECT1 (1244).

If "A" is typed, the program will execute the logic tests, analog tests, noise, settle and differential linearity. At the beginning of the test the program will type "XX ADV11'S FOUND". Where XX is the number of ADV11's in octal. If the number is greater than 1 the test will run successively on each ADV11.

If "L" is typed, the program will execute the logic tests, printing "END PASS" when it has completed an entire pass. At the beginning of the test the program will type "XX ADV11'S FOUND". Where XX is the number of ADV11's in octal. If the number is greater than 1, the test will be run successively on each ADV11.

5.1 Inhibiting Auto-Size Feature

This program will automatically auto-size and test each ADV11 it detects on the system. To inhibit this feature, set bit 15 of location $ENVM (1214). Also, load location $BASE (1250) with the ADV11's status register address and the low byte of location $VECT1 (1244) with the ADV11's vector address.

5.2 End of Pass Typeouts

At end of pass, the following typeout will occur:

```
"ENDPASS  GOOD UNITS  0000000000000011
```

This indicates that units 1 and 2 have run without failure.

6.0 ERRORS

This program uses the Diagnostic "SYSMAC" package for error reporting and typeout. The error information consists of the following:

```
ERRPC:    Location at which an error was detected.
STREG:    Address of the status register.
ADBUFF:   Address of the buffer
CHANL:    Channel value
NOMINAL:  Expected correct data
TOLERANCE: The acceptable deviation from the nominal
ACTUAL:   Actual data
EXPECTED: Expected correct data
```

7.0 MISCELLANEOUS

7.1 Execution Time

Execution time for each of the tests is:

Calibration:	5 conversions/min @110 baud
Print Values:	8 conversions/8 seconds @ 110 baud
Wraparound Test:	7 minutes first pass; 25 minutes for successive passes
Logic Test:	1 minute
Auto Test:	8 minutes first pass, 26 minutes for successive passes

7.2 Status Register and Vector Addresses and Priority

When testing more than one ADV11, the difference in addresses is 4 for bus address and 10 for vector address. These values are in VADR (bus address) (1336) and VVCT (vector address) (1340). The first ADV11's status register address must be in $BASE (1250), its vector address must be in the low byte of $VECT1 (1244).

7.3 Switch Register

If a hardware switch register is present and the operator desires to use a software switch register and the ^G feature; it is necessary to load the starting address, set the hardware switch register to all ones (-1), and depress start. The program will then run with the software switch register.

7.4 VT55 Graphic Output

The screen display may be halted for examination by setting bit 12 of the switch register. Then, type "P" to complete the program's execution.

8.0 RESTRICTIONS

8.1 Testing

The Test Connector must be present when running the auto test and the wraparound test.

8.2 Starting Restriction

If a free-running clock, such as 60Hz from the power supply, is attached to the BEVNT bus line on both Rev level C/D and E systems, an interrupt to location 100 will occur when using the "G" and "L" commands prior to executing the first instruction. Therefore this program can not disable the BEVNT bus line by inhibiting interrupts.

User systems requiring a free-running clock attached to the BEVNT bus line can temporarily avoid this situation by setting the PSW(RS) to 200, instead of using the "G" command, load the PC (R7) with the starting address and use the proceed "P" command. Before using the "L" command, the PSW(RS) can be set to 200 to avoid receiving the BEVNT interrupt after loading the ABS loader.

8.3 Possible Program "BOMBS"

The first two tests of this program check to see if the ADV11 responds to the expected address. If the ADV11 does not respond, a buss error occurs. Also bus errors can occur during the time the program sizes to see how many ADV11's are on your system.

For more information on the next subject, see JAN. 1976 LSI-11 ENGINEERING BULLETIN issued by The Digital Components Group.

Bus errors may alter the preset contents of location 4 before the trap is executed, thereby transferring program control to area in the program that was not set up to handle the trap. If this happens, the program will "BOMB" and possibly rewrite parts of itself.

9.0 PROGRAM DESCRIPTION

9.1 Logic Tests

These 21 logic subtests run sequentially without further operator intervention. Its purpose is to check that each of the status register bits that are read/write can be loaded and properly read back; that initialize clears the external start enable bit, the done bit, the interrupt enable bit, the overflow bit, the error flag, and the A/D start bit. It also checks that the A/D done flag sets at end of conversion and clears when the converted value is read. It checks the interrupt logic and the correct setting of the error flag.

9.2 Calibration Routine

If "C" is typed, the program will ask for a channel. Type channel number followed by a carriage return. The program will ask you if you want offset or gain. Apply voltage requested to selected channel. Adjust pot requested for 0.00 LSB typeout. Type carriage return when adjusted. The last typeout will be checked for 0.00 LSB with a tolerance of 0.04 LSB if outside, the program will ask you to adjust the same pot again.

9.3 Print Values Routine

This test begins when the operator types "P". It then loads the channel from the switch register bits 0-7 and does a conversion on that channel. If SWR bit 13 is down (0), it prints out the converted value on the teletype; otherwise, if SWR bit 13 is up (1), it puts the converted value in the display register. The operator may change the channel at any time during the test, however the new values from the new channel will not be printed until the next line of 8 values is printed. The 8 values on each line correspond to only one channel.

9.4 Differential Linearity

This test determine if a change in the input voltage represents a similar change in the resulting converted binary value, by measuring the width of each state correct to 0.01 LSB.

9.5 Settling Test

The purpose of this test is to check that the time needed to settle and correctly report a new input value after switching channels does not exceed the expected amount of time for such a change.

9.6 Noise Test

This test measures the internal short-term repeatability noise within the A/D. RMS noise.equals 1 standard deviation of the Gaussian curve, PEAK noise equals 2.3 standard deviation of the Gaussian curve.

9.7 Analog Tests

These 6 subtests check the channels and their output.

```
    22          BASIC DEFINITIONS
    23          OPERATIONAL SWITCH SETTINGS
    31          TRAP CATCHER
   (1)          STARTING ADDRESS(ES)
    35          ACT11 HOOKS
    37          APT PARAMETER BLOCK
    38          COMMON TAGS
   (2)          APT MAILBOX-ETABLE
   (1)          ERROR POINTER TABLE
    76          MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
   126          CONTROL A AND C DECODERS
   159          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
   164          INITIALIZE THE COMMON TAGS
   169          DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
   185          DIALOGUE TO DETERMINE WHICH TEST TO RUN
   271          T1      FLOAT A ONE THRU MULTIPLEXER BITS
   280          T2      LOAD AND READ BACK ERROR I.E. BIT14
   284          T3      LOAD AND READ BACK INTERRUPT ENABLE BIT6
   290          T4      LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
   295          T5      LOAD AND READ BACK EXTERNAL START ENABLE BIT4
   299          T6      LOAD AND READ BACK MAINT. TST BIT2
   304          T7      LOAD AND READ BACK ENABLE I.D. BIT3
   308          T10     TEST I.D. BIT (BIT 12) CLEARED
   318          T11     TEST I.D. BIT (BIT 12) SET
   328          T12     LOAD AND READ BACK ERROR FLAG BIT15
   332          T13     TEST INIT CLEARS BITS 2-6,8-11,14
   342          T14     TEST INIT CLEARS ERROR FLAG
   349          T15     TEST  DONE FLAG  SETS AND BIT0 CLEARS ON END OF CONV.
   360          T16     TEST INIT CLEARS DONE FLAG
   370          T17     TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
   377          T20     TEST ALL '0'S RESULTS USING MAINT. ADTST. BIT
   388          T21     TEST ALL '1'S RESULT USING MAINT. ADTST. BIT
   400          T22     GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
   422          T23     TEST INTERRUPT OCCURS WHEN ERROR AND I.E.E. IS SET
   435          T24     TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
   448          T25     TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
   475          WRAPAROUND TEST SECTION
   477          T26     TEST CH0 GROUND
   486          T27     TEST CH1 +4.5 VOLT
   494          T30     TEST CH2 -4.5 VOLT
   501          T31     TEST GROUND ON CHANNELS 4 - 17
   513          T32     TEST VERNIER OFFSET DAC ON CH0
   526          T33     OFFSET ON CH0
   541          T34     TEST RAMP RANGE, CH3
   569          T35     NOISE TEST, 1 EDGE
   597          T36     INTERCHANNEL SETTLING TEST, 1 EDGE
   618          T37     DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
   714          PRINT VALUES ROUTINE
   749          LOGIC TEST SECTION
   758          AUTO TEST
   774          WRAPAROUND TEST
   783          DETERMINE IF MORE ADV11'S TO BE TESTED
  1366          END OF PASS ROUTINE
  1368          ASCII MESSAGES
  1463          TTY INPUT ROUTINE
  1465          READ AN OCTAL NUMBER FROM THE TTY
  1467          SCOPE HANDLER ROUTINE
  1468          ERROR HANDLER ROUTINE
  1469          ERROR MESSAGE TYPEOUT ROU¯INE
  1471          TYPE ROUTINE
  1472          APT COMMUNICATIONS ROUTINE
  1474          BINARY TO OCTAL (ASCII) AN) TYPE
  1475          BINAPY TO ASCII AND TYPE R)UTINE
  1477          TRAP DECODER
   (3)          TRAP TABLE
```

```
     1                           ;DEVELOPED USING SYSMAC.C3
    21                           .TITLE  MAINDEC-11-CVADA-C
   (1)                           ;*COPYRIGHT (C) 1980
   (1)                           ;*DIGITAL EQUIPMENT CORP.
   (1)                           ;*MAYNARD, MASS. 01754
   (1)                           ;*
   (1)                           ;*PROGRAM BY G. STEVENS MOD. R.SHOOP
   (1)                           ;*
   (1)                           ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
   (1)                           ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
   (1)                           ;*
    22                           .SBTTL  BASIC DEFINITIONS
   (1)
   (1)                           ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
   (1)        001100             STACK=  1100
   (1)                           .EQUIV  EMT,ERROR        ;;BASIC DEFINITION OF ERROR CALL
   (1)                           .EQUIV  IOT,SCOPE        ;;BASIC DEFINITION OF SCOPE CALL
   (1)
   (1)                           ;*MISCELLANEOUS DEFINITIONS
   (1)        000011             HT=     11               ;;CODE FOR HORIZONTAL TAB
   (1)        000012             LF=     12               ;;CODE FOR LINE FEED
   (1)        000015             CR=     15               ;;CODE FOR CARRIAGE RETURN
   (1)        000200             CRLF=   200              ;;CODE FOR CARRIAGE RETURN-LINE FEED
   (1)        177776             PS=     177776           ;;PROCESSOR STATUS WORD
   (1)                           .EQUIV  PS,PSW
   (1)        177774             STKLMT= 177774           ;;STACK LIMIT REGISTER
   (1)        177772             PIRQ=   177772           ;;PROGRAM INTERRUPT REQUEST REGISTER
   (1)        177570             DSWR=   177570           ;;HARDWARE SWITCH REGISTER
   (1)        177570             DDISP=  177570           ;;HARDWARE DISPLAY REGISTER
   (1)
   (1)                           ;*GENERAL PURPOSE REGISTER DEFINITIONS
   (1)        000000             R0=     %0               ;;GENERAL REGISTER
   (1)        000001             R1=     %1               ;;GENERAL REGISTER
   (1)        000002             P2=     %2               ;;GENERAL REGISTER
   (1)        000003             R3=     %3               ;;GENERAL REGISTER
   (1)        000004             R4=     %4               ;;GENERAL REGISTER
   (1)        000005             R5=     %5               ;;GENERAL REGISTER
   (1)        000006             R6=     %6               ;;GENERAL REGISTER
   (1)        000007             R7=     %7               ;;GENERAL REGISTER
   (1)        000006             SP=     %6               ;;STACK POINTER
   (1)        000007             PC=     %7               ;;PROGRAM COUNTER
   (1)
   (1)                           ;*PRIORITY LEVEL DEFINITIONS
   (1)        000000             PR0=    0                ;;PRIORITY LEVEL 0
   (1)        000040             PR1=    40               ;;PRIORITY LEVEL 1
   (1)        000100             PR2=    100              ;;PRIORITY LEVEL 2
   (1)        000140             PR3=    140              ;;PRIORITY LEVEL 3
   (1)        000200             PR4=    200              ;;PRIORITY LEVEL 4
   (1)        000240             PR5=    240              ;;PRIORITY LEVEL 5
   (1)        000300             PR6=    300              ;;PRIORITY LEVEL 6
   (1)        000340             PR7=    340              ;;PRIORITY LEVEL 7
   (1)
   (1)                           ;*"SWITCH REGISTER" SWITCH DEFINITIONS
   (1)        100000             SW15=   100000
   (1)        040000             SW14=   40000
   (1)        020000             SW13=   20000
   (1)            010000                  SW12=   10000
   (1)            004000                  SW11=   4000
   (1)            002000                  SW10=   2000
   (1)            001000                  SW09=   1000
   (1)            000400                  SW08=   400
   (1)            000200                  SW07=   200
   (1)            000100                  SW06=   100
   (1)            000040                  SW05=   40
   (1)            000020                  SW04=   20
   (1)            000010                  SW03=   10
   (1)            000004                  SW02=   4
   (1)            000002                  SW01=   2
   (1)            000001                  SW00=   1
   (1)                                    .EQUIV  SW09,SW9
   (1)                                    .EQUIV  SW08,SW8
   (1)                                    .EQUIV  SW07,SW7
   (1)                                    .EQUIV  SW06,SW6
   (1)                                    .EQUIV  SW05,SW5
   (1)                                    .EQUIV  SW04,SW4
   (1)                                    .EQUIV  SW03,SW3
   (1)                                    .EQUIV  SW02,SW2
   (1)                                    .EQUIV  SW01,SW1
   (1)                                    .EQUIV  SW00,SW0
   (1)
   (1)                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
   (1)            100000                  BIT15=  100000
   (1)            040000                  BIT14=  40000
   (1)            020000                  BIT13=  20000
   (1)            010000                  BIT12=  10000
   (1)            004000                  BIT11=  4000
   (1)            002000                  BIT10=  2000
   (1)            001000                  BIT09=  1000
   (1)            000400                  BIT08=  400
   (1)            000200                  BIT07=  200
   (1)            000100                  BIT06=  100
   (1)            000040                  BIT05=  40
   (1)            000020                  BIT04=  20
   (1)            000010                  BIT03=  10
   (1)            000004                  BIT02=  4
   (1)            000002                  BIT01=  2
   (1)            000001                  BIT00=  1
   (1)                                    .EQUIV  BIT09,BIT9
   (1)                                    .EQUIV  BIT08,BIT8
   (1)                                    .EQUIV  BIT07,BIT7
   (1)                                    .EQUIV  BIT06,BIT6
   (1)                                    .EQUIV  BIT05,BIT5
   (1)                                    .EQUIV  BIT04,BIT4
   (1)                                    .EQUIV  BIT03,BIT3
   (1)                                    .EQUIV  BIT02,BIT2
   (1)                                    .EQUIV  BIT01,BIT1
   (1)                                    .EQUIV  BIT00,BIT0
   (1)
   (1)                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
   (1)            000004                  ERRVEC= 4                ;;TIME OUT AND OTHER ERRORS
   (1)            000010                  RESVEC= 10               ;;RESERVED AND ILLEGAL INSTRUCTIONS
   (1)            000014                  TBITVEC=14               ;;"T" BIT
```

```
  (1)          000014                          TRTVEC= 14                ;;TRACE TRAP
  (1)          000014                          BPTVEC= 14                ;;BREAKPOINT TRAP (BPT)
  (1)          000020                          IOTVEC= 20                ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
  (1)          000024                          PWRVEC= 24                ;;POWER FAIL
  (1)          000030                          EMTVEC= 30                ;;EMULATOR TRAP (EMT) **ERROR**
  (1)          000034                          TRAPVEC=34                ;;"TRAP" TRAP
  (1)          000060                          TKVEC=  60                ;;TTY KEYBOARD VECTOR
  (1)          000064                          TPVEC=  64                ;;TTY PRINTER VECTOR
  (1)          000240                          PIRQVEC=240               ;;PROGRAM INTERRUPT REQUEST VECTOR
   23                                          .SBTTL  OPERATIONAL SWITCH SETTINGS
  (1)                                          ;*
  (1)                                          ;*      SWITCH                  USE
  (1)                                          ;*      ------            --------------------
  (1)                                          ;*        15              HALT ON ERROR
  (1)                                          ;*        14              LOOP ON TEST
  (1)                                          ;*        13              INHIBIT ERROR TYPEOUTS
  (1)                                          ;*        12              HALT FOR VT55 DISPLAY
  (1)                                          ;*        11              INHIBIT ITERATIONS
  (1)                                          ;*        10              BELL ON ERROR
  (1)                                          ;*         9              LOOP ON ERROR
  (1)                                          ;*         8              LOOP ON TEST IN SWR<7:0>
   24          170400                          ABASE=  170400
   25          100400                          AVECT1= 100400
   26          000200                          APRIOR= 200
   27
   28          000100                                  .=100
   29  000100  000104  000200  000002                  .WORD   104,200,2
   30
   31                                          .SBTTL  TRAP CATCHER
  (1)
  (1)          000000                                  .=0
  (1)                                          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
  (1)                                          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
  (1)                                          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
  (1)          000174                                  .=174
  (1)  000174  000000                          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
  (1)  000176  000000                          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
  (1)                                          .SBTTL  STARTING ADDRESS(ES)
  (1)  000200  000137  001644                          JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
   32  000204  000137  002262                          JMP     @#BEG2          ;RESTART ADDRESS
   33  000210  000137  001652                          JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
   35                                  .SBTTL  ACT11 HOOKS
  (1)
  (2)                                  ;;*****************************************************************
  (1)                                  ;HOOKS REQUIRED BY ACT11
  (1)              000214                      $SVPC=.                 ;SAVE PC
  (1)              000046                      .=46
  (1)   000046     012022                      $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
  (1)              000052                      .=52
  (1)   000052     000000                      .WORD   0               ;;2)SET LOC.52 TO ZERO
  (1)              000214                      .=$SVPC                 ;; RESTORE PC
   36              001000              .=1000
   37                                  .SBTTL  APT PARAMETER BLOCK
  (1)
  (2)                                  ;;*****************************************************************
  (1)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  (2)                                  ;;*****************************************************************
  (1)              001000                      .$X=.   ;;SAVE CURRENT LOCATION
  (1)              000024                      .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
  (1)   000024     000200                      200     ;;FOR APT START UP
  (1)              000044                      .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
  (1)   000044     001000                      $APTHDR ;;POINT TO APT HEADER BLOCK
  (1)              001000                      .=.$X   ;;RESET LOCATION COUNTER
  (2)                                  ;;*****************************************************************
  (1)                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
  (1)                                  ;INTERFACE SPEC.
  (1)
  (1)   001000                         $APTHD:
  (1)   001000     000000              $HIBTS: .WORD   0        ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
  (1)   001002     001174              $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
  (1)   001004     000454              $TSTM:  .WORD   300.     ;;RUN TIM OF LONGEST TEST
  (1)   001006     000074              $PASTM: .WORD   60.      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  (1)   001010     000454              $UNITM: .WORD   300.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  (1)   001012     000031                      .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
   38                                  .SBTTL  COMMON TAGS
  (1)
  (2)                                  ;;*************************************************************
  (1)                                  ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
  (1)                                  ;*USED IN THE PROGRAM.
  (1)
  (1)          001100                          .=1100
  (1)  001100                          $CMTAG:                                  ;;START OF COMMON TAGS
  (1)  001100  000000                          .WORD   0
  (1)  001102     000                  $TSTNM: .BYTE   0                        ;;CONTAINS THE TEST NUMBER
  (1)  001103     000                  $ERFLG: .BYTE   0                        ;;CONTAINS ERROR FLAG
  (1)  001104  000000                  $ICNT:  .WORD   0                        ;;CONTAINS SUBTEST ITERATION COUNT
  (1)  001106  000000                  $LPADR: .WORD   0                        ;;CONTAINS SCOPE LOOP ADDRESS
  (1)  001110  000000                  $LPERR: .WORD   0                        ;;CONTAINS SCOPE RETURN FOR ERRORS
  (1)  001112  000000                  $ERTTL: .WORD   0                        ;;CONTAINS TOTAL ERRORS DETECTED
  (1)  001114     000                  $ITEMB: .BYTE   0                        ;;CONTAINS ITEM CONTROL BYTE
  (1)  001115     001                  $ERMAX: .BYTE   1                        ;;CONTAINS MAX. ERRORS PER TEST
  (1)  001116  000000                  $ERRPC: .WORD   0                        ;;CONTAINS PC OF LAST ERROR INSTRUCTION
  (1)  001120  000000                  $GDADR: .WORD   0                        ;;CONTAINS ADDRESS OF 'GOOD' DATA
  (1)  001122  000000                  $BDADR: .WORD   0                        ;;CONTAINS ADDRESS OF 'BAD' DATA
  (1)  001124  000000                  $GDDAT: .WORD   0                        ;;CONTAINS 'GOOD' DATA
  (1)  001126  000000                  $BDDAT: .WORD   0                        ;;CONTAINS 'BAD' DATA
  (1)  001130  000000                          .WORD   0                        ;;RESERVED--NOT TO BE USED
  (1)  001132  000000                          .WORD   C
  (1)  001134     000                  $AUTOB: .BYTE   0                        ;;AUTOMATIC MODE INDICATOR
  (1)  001135     000                  $INTAG: .BYTE   0                        ;;INTERRUPT MODE INDICATOR
  (1)  001136  000000                          .WORD   0
  (1)  001140  177570                  SWR:    .WORD   DSWR                     ;;ADDRESS OF SWITCH REGISTER
  (1)  001142  177570                  DISPLAY: .WORD  DDISP                    ;;ADDRESS OF DISPLAY REGISTER
  (1)  001144  177560                  $TKS:   177560                           ;;TTY KBD STATUS
  (1)  001146  177562                  $TKB:   177562                           ;;TTY KBD BUFFER
  (1)  001150  177564                  $TPS:   177564                           ;;TTY PRINTER STATUS REG. ADDRESS
  (1)  001152  177566                  $TPB:   177566                           ;;TTY PRINTER BUFFER REG. ADDRESS
  (1)  001154     000                  $NULL:  .BYTE   0                        ;;CONTAINS NULL CHARACTER FOR FILLS
  (1)  001155     002                  $FILLS: .BYTE   2                        ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
  (1)  001156     012                  $FILLC: .BYTE   12                       ;;INSERT FILL CHARS. AFTER A "LINE FEED"
  (1)  001157     000                  $TPFLG: .BYTE   0                        ;;"TERMINAL AVAILABLE" FLAG (BIT<07>-0=YES)
  (1)  001160  000000                  $TIMES: 0                                ;;MAX. NUMBER OF ITERATIONS
  (1)  001162  000000                  $ESCAPE:0                                ;;ESCAPE ON ERROR ADDRESS
  (1)  001164  177607  000377          $BELL:  .ASCIZ  <207><377><377>          ;;CODE FOR BELL
  (1)  001170     077                  $QUES:  .ASCII  /?/                      ;;QUESTION MARK
  (1)  001171     015                  $CRLF:  .ASCII  <15>                     ;;CARRIAGE RETURN
  (1)  001172  000012                  $LF:    .ASCIZ  <12>                     ;;LINE FEED
  (2)                                  ;;*************************************************************
  (2)                                  .SBTTL  APT MAILBOX-ETABLE
  (2)
  (3)                                  ;;*************************************************************
  (2)                                  .EVEN
  (2)  001174                          $MAIL:                        ;;APT MAILBOX
  (2)  001174  000000                  $MSGTY: .WORD   AMSGTY        ;;MESSAGE TYPE CODF
  (2)  001176  000000                  $FATAL: .WORD   AFATAL        ;;FATAL ERROR NUMBER
  (2)  001200  000000                  $TESTN: .WORD   ATESTN        ;;TEST NUMBER
  (2)  001202  000000                  $PASS:  .WORD   APASS         ;;PASS COUNT
  (2)  001204  000000                  $DEVCT: .WORD   ADEVCT        ;;DEVICE COUNT
  (2)  001206  000000                  $UNIT:  .WORD   AUNIT         ;;I/O UNIT NUMBER
  (2)  001210  000000                  $MSGAD: .WORD   AMSGAD        ;;MESSAGE ADDRESS
```

```
   (2)  001212  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
   (2)  001214                          $ETABLE:                ;;APT ENVIRONMENT TABLE
   (2)  001214     000                  $ENV:    .BYTE  AENV    ;;ENVIRONMENT BYTE
   (2)  001215     000                  $ENVM:   .BYTE  AENVM   ;;ENVIRONMENT MODE BITS
   (2)  001216  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
   (2)  001220  000000                  $USWR:   .WORD  AUSWR   ;;USER SWITCHES
   (2)  001222  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
   (2)                                  ;*                      BITS 15-11=CPU TYPE
   (2)                                  ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
   (2)                                  ;*                              11/70=06,PDQ=07,Q=10
   (2)                                  ;*                      BIT 10=REAL TIME CLOCK
   (2)                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
   (2)                                  ;*                      BIT  8=MEMORY MANAGEMENT
   (2)  001224     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
   (2)  001225     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
   (2)                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
   (2)                                  ;*                              900 NSEC CORE=001
   (2)                                  ;*                              300 NSEC BIPOLAR=002
   (2)                                  ;*                              500 NSEC MOS=003
   (2)  001226  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
   (2)                                  ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
   (2)  001230     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
   (2)  001231     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
   (2)  001232  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
   (2)  001234     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
   (2)  001235     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
   (2)  001236  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
   (2)  001240     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
   (2)  001241     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
   (2)  001242  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
   (2)  001244  100400                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
   (2)  001246  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
   (2)  001250  170400                  $BASE:   .WORD  ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
   (2)  001252  000000                  $DEVM:   .WORD  ADEVM   ;;DEVICE MAP
   (2)  001254  000000                  $CDW1:   .WORD  ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
   (2)  001256                          $ETEND:
   (2)                                  .MEXIT
```

```
(1)                                 .SBTTL  ERROR POINTER TABLE
(1)
(1)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                 ;*      DT              ;;POINTS TO THE DATA
(1)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1) 001256                          $ERRTB:



                                    ;ITEM   1
    001256  014325                          EM1                     ;STATUS REG. ERROR
    001260  014445                          DH1                     ;ERRPC STREG EXPECTED ACTUAL
    001262  014624                          DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
    001264  014664                          DF1


                                    ;ITEM   2
    001266  014347                          EM2                     ;FAILED TO INTERRUPT
    001270  014564                          DH3                     ;ERRPC STREG ACTUAL
    001272  014654                          DT3                     ;$ERRPC, STREG,  $BDDAT
    001274  014664                          DF1

                                    ;ITEM   3
    001276  014373                          EM3                     ;UNEXPECTED INTERRUPT
    001300  014564                          DH3                     ;ERRPC STREG
    001302  014654                          DT3                     ;$ERRPC, STREG
    001304  014664                          DF1

                                    ;ITEM   4
    001306  014420                          EM4                     ;ERROR ON A/D CHANNEL
    001310  014501                          DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
    001312  014636                          DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT
    001314  014664                          DF1
```

```
 76                                      .SBTTL      MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
 77  001316  170400                      STREG:  ABASE                      ;ADDRESS OF STATUS REGISTER
 78  001320  170401                      ADST1:  ABASE+1                    ;UPPER BYTE OF STATUS REG.
 79  001322  170402                      ADBUFF: ABASE+2                    ;ADDRESS OF A/D BUFFER
 80  001324  100400                      VECTOR: AVECT1                     ;VECTOR ADDRESS
 81  001326  000200                      BASEBR: APRIOR                     ;INTERRUPT PRIORITY LEVEL
 82  001330  100402                      VECTR1: AVECT1+2
 83  001332  100404                      VECTR2: AVECT1+4                   ;ERROR VECTOR ADDRESS
 84  001334  100406                      VECTR3: AVECT1+6
 85  001336  000004                      VADR:   4                          ;INCREMENT FOR BUS ADDRESS
 86  001340  000010                      VVCT:   10                         ;INCREMENT FOR VECTOR ADDRESS
 87  001342  000000                      BASECH: 0                          ;BASE CHANNEL
 88  001344  000060                      KBVECT: 60
 89  001346  000000                      WIDE:   0                          ;NO. OF WIDE STATES
 90  001350  000000                      NARROW: 0                          ;NO. OF NARROW STATES
 91  001352  000000                      FIRST:  0
 92  001354  000000                      SKIPST: 0                          ;NO. OF SKIPPED STATES
 93  001356  000000                      TEMP:   0                          ;WORK AREA
 94  001360  000000                      CH1:    0                          ;FIRST CHANNEL
 95  001362  000000                      CH2:    0                          ;SECOND CHANNEL
 96  001364  000000                      NBEXT:  0                          ;NO. OF ADV11'S TO BE TESTED
 97  001366  000000                      NMBEXT: 0                          ;NO. OF ADV11'S TO BE TESTED
 98  001370  000000                      DUMMY:  0                          ;DUMMY CHANNEL
 99  001372  000000                      CHANL:  0                          ;CHANNEL VALUE
100  001374  000000                      TADDR:  0                          ;TEST ADDRESS
101  001376  000000                      RNA:    0                          ;RANDOM
102  001400  000000                      RNB:    0                          ;NUMBER
103  001402  000000                      RNC:    0                          ;VALUES
104  001404  000000                      RMS:    0                          ;RMS NOISE VALUE
105  001406  000000                      PEAK:   0                          ;PEAK NOISE VALUE
106  001410  000000                      FLAG:   0                          ;VT55 FLAG
107  001412  000000                      SPREAD: 0                          ;DEVIATION FROM THE NOMINAL
108  001414  000000                      DAC:    0                          ;SAR VALUE
109  001416  000000                      DELAY:  0                          ;TIME DELAY COUNTER
110  001420  000000                      EDGE:   0                          ;EDGE VALUE
111  001422  000000                      BITPNT: 0
112  001424  000000                      MIN:    0                          ;MIN VALUE
113  001426  000000                      WFTEST: 0                          ;OPTION TEST AREA FLAG
114  001430  000000                      MAX:    0                          ;MAX VALUE
115  001432  000000                      PERCNT: 0                          ;PERCENT FOR SAR ROUTINE
116  001434  000000                      OUT:    0
117  001436  000000                      GUNITS: 0
118  001440  000001                      TSTBIT: 1
119
120  001442                              UNEXP:
(1)  001442  012737  001456  001162              MOV     #1$,$ESCAPE    ;;ESCAPE TO 1$ ON ERROR
121  001450  005237  001103                      INC     $ERFLG
122  001454  104003                              ERROR   3
123  001456  005037  001162              1$:     CLR     $ESCAPE        ;RETURN ESCAPE TO NORMAL
124  001462  000002                              RTI                    ;UNEXPECTED INTERRUPT
```

```
                                         .SBTTL        CONTROL A AND C DECODERS
     001464  010046                      ISERV:  MOV     R0,-(SP)                        ;SAVE R0
     001466  017700  177454                      MOV     @$TKB,R0                        ;GET CHARACTER
     001472  042700  177600                      BIC     #177600,R0
     001476  120027  000003                      CMPB    R0,#3                           ;IS IT ^C?
     001502  001010                              BNE     1$
     001504  104401  012114                      TYPE    ,CMSG                           ;ECHO CHARACTER
     001510  012706  001100                      MOV     #STACK,SP
     001514  004737  011356                      JSR     PC,RST                          ;RESET & SET INTRPT. EN.
     001520  000137  002262                      JMP     BEG2
     001524  120027  000001          1$:         CMPB    R0,#1                           ;IS IT ^A?
     001530  001010                              BNE     2$
     001532  104401  012107                      TYPE    ,AMSG                           ;ECHO CHARACTER
     001536  012706  001100                      MOV     #STACK,SP
     001542  004737  011356                      JSR     PC,RST                          ;RESET & SET INTRPT. EN.
     001546  000177  177622                      JMP     @TADDR                          ;RETURN TO TEST
     001552  120027  000007          2$:         CMPB    R0,#7                           ;IS IT ^G?
     001556  001027                              BNE     NONE
     001560  023727  001140  177570              CMP     SWR,#177570                     ;HARDWARE SWREG?
     001566  001423                              BEQ     NONE
     001570  104401  012121                      TYPE    ,GMSG                           ;ECHO CHARACTER
     001574  017746  177340                      MOV     @SWR,-(SP)              ;;SAVE @SWR FOR TYPEOUT
(1)                                                                              ;;TYPE SWREG
(1)  001600  104403                              TYPOS                           ;;GO TYPE--OCTAL ASCII
(1)  001602     006                              .BYTE   6                       ;;TYPE 6 DIGITS
(1)  001603     001                              .BYTE   1                       ;;TYPE LEADING ZEROS
     001604  104401  012301                      TYPE    ,SLASH
     001610  104410                              RDOCT                                   ;READ NEW VALUE
     001612  012677  177322                      MOV     (SP)+,@SWR                      ;LOAD NEW SWREG VALUE
     001616  012600                      POPR0:  MOV     (SP)+,R0
     001620  022776  000001  000000      RETURN: CMP     #1,@0(SP)               ;DOES IT RETURN TO A WAIT?
     001626  001002                              BNE     RET2                    ;NO
     001630  062716  000002              RET1:   ADD     #2,(SP)                 ;BUMP RETURN ADDRESS
     001634  000002                      RET2:   RTI
     001636  104401  012105              NONE:   TYPE    ,QUEST                          ;TYPE "?"
     001642  000765                              BR      POPR0
```

```
  159                                  .SBTTL         INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
  160  001644  005037  001426          BEGIN:  CLR     WFTEST
  161  001650  000403                          BR      RBEG
  162  001652  012737  000001  001426  BEGIN2: MOV     #1,WFTEST
  163  001660  000005                  RBEG:   RESET
  164                                  .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001662  012706  001100                  MOV     #$CMTAG,R6        ;;FIRST LOCATION TO BE CLEARED
  (1)  001666  005026                          CLR     (R6)+             ;;CLEAR MEMORY LOCATION
  (1)  001670  022706  001140                  CMP     #SWR,R6 ;;DONE?
  (1)  001674  001374                          BNE     .-6               ;;LOOP BACK IF NO
  (1)  001676  012706  001100                  MOV     #STACK,SP         ;;SETUP THE STACK POINTER
  (1)                                  ;;INITIALIZE A FEW VECTORS
  (1)  001702  012737  015262  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001710  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001716  012737  015540  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001724  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001732  012737  017130  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001740  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  001746  013737  011742  011734          MOV     $ENDCT,$EOPCT     ;;SETUP END-OF-PROGRAM COUNTER
  (1)  001754  005037  001160                  CLR     $TIMES            ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  001760  005037  001162                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  001764  112737  000001  001115          MOVB    #1,$ERMAX         ;;ALLOW ONE ERROR PER TEST
  (1)  001772  012737  001772  001106          MOV     #.,$LPADR         ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  002000  012737  002000  001110          MOV     #.,$LPERR         ;;SETUP THE ERROR LOOP ADDRESS
  (2)                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  002006  013746  000004                  MOV     @#ERRVEC,-(SP)    ;;SAVE ERROR VECTOR
  (2)  002012  012737  002046  000004          MOV     #64$,@#ERRVEC     ;;SET UP ERROR VECTOR
  (2)  002020  012737  177570  001140          MOV     #DSWR,SWR         ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  002026  012737  177570  001142          MOV     #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
  (2)  002034  022777  177777  177076          CMP     #-1,@SWR          ;;TRY TO REFERENCE HARDWARE SWR
  (2)  002042  001012                          BNE     66$               ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  002044  000403                          BR      65$               ;;BRANCH IF NO TIMEOUT
  (2)  002046  012716  002054          64$:    MOV     #65$,(SP)         ;;SET UP FOR TRAP RETURN
  (2)  002052  000002                          RTI
  (2)  002054  012737  000176  001140  65$:    MOV     #SWREG,SWR        ;;POINT TO SOFTWARE SWR
  (2)  002062  012737  000174  001142          MOV     #DISPREG,DISPLAY
  (2)  002070  012637  000004          66$:    MOV     (SP)+,@#ERRVEC    ;;RESTORE ERROR VECTOR
  (1)
  (2)  002074  005037  001202                  CLR     $PASS             ;;CLEAR PASS COUNT
  (2)  002100  132737  000200  001215          BITB    #APTSIZE,$ENVM    ;;TEST USER SIZE UNDER APT
  (2)  002106  001403                          BEQ     67$               ;;YES,USE NON-APT SWITCH
  (2)  002110  012737  001216  001140          MOV     #$SWREG,SWR       ;;NO,USE APT SWITCH REGISTER
  (2)  002116                          67$:
```

```
002116  005037  001410                         CLR     FLAG            ;CLEAR VT55 FLAG
002122  005737  000042                         TST     @#42                    ;IS IT CHAINED?
002126  001033                                 BNE     REST1
                                       .SBTTL        DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
002130  042777  000100  177006                 BIC     #100,@$TKS
002136  104401  014002                         TYPE    ,CO             ;TYPE ASCIZ STRING
002142  004737  002432                         JSR     PC,VTFLG        ;GET A CHARACTER
002146  020027  000033                         CMP     R0,#33
002152  001017                                 BNE     NOVT55          ;NO VT55 PRESENT
002154  004737  002432                         JSR     PC,VTFLG        ;GET A CHARACTER
002160  020027  000057                         CMP     R0,#57
002164  001012                                 BNE     NOVT55          ;NO VT55 PRESENT
002166  004737  002432                         JSR     PC,VTFLG        ;GET A CHARACTER
002172  020027  000103                         CMP     R0,#103
002176  001403                                 BEQ     VT55            ;VT55 IS PRESENT
002200  020027  000105                         CMP     R0,#105
002204  001002                                 BNE     NOVT55
002206  005237  001410                 VT55:   INC     FLAG
```

```
                                     .SBTTL         DIALOGUE TO DETERMINE WHICH TEST TO RUN
002212  104401  014145               NOVT55: TYPE   ,HEAD1
002216  000005                       REST1:  RESET
002220  004737  006344                       JSR    PC,FIXONE               ;INITIALIZE ADDPESSES
002224  013700  001344                       MOV    KBVECT,R0
002230  012720  001464                       MOV    #ISERV,(R0)+
002234  012710  000340                       MOV    #340,(R0)
002240  012737  062341  001376               MOV    #62341,RNA              ;RANDOM NO, VARIABLES
002246  012737  142315  001400               MOV    #142315,RNB
002254  012737  127623  001402               MOV    #127623,RNC
002262  012706  001100               BEG2:   MOV    #STACK,SP       ;RESET STACK POINTER INCASE RESTARTED
002266  000005                               RESET                  ;RESTART ADDRESS
002270  005737  000042                       TST    @#42                    ;IS IT CHAINED?
002274  001405                               BEQ    1$
002276  000137  006064               2$:     JMP    BEGL                    ;GO TO LOGIC TESTS
002302  005737  001426                       TST    WFTEST                  ;TEST FOR OPTION TEST
002306  001373                               BNE    2$              ;;
002310  104401  013617               1$:     TYPE   ,MSG71
002314  104407                       TRYAG:  RDLIN
002316  052777  000100  176620               BIS    #100,@$TKS
002324  005046                               CLR    -(SP)                   ;CLEAR PSW
002326  012746  002334                       MOV    #1$,-(SP)
002332  000002                               RTI
002334  012600                       1$:     MOV    (SP)+,R0                ;READ ANSWER
002336  142710  000040                       BICB   #40,(R0)
002342  121027  000101                       CMPB   (R0),#'A                ;IS IT A?
002346  001002                               BNE    2$              ;;NO, TRY C
002350  000137  006122                       JMP    BEGINA                  ;GO TO AUTO TEST
002354  121027  000103               2$:     CMPB   (R0),#'C                ;IS IT C?
002360  001002                               BNE    3$              ;;NO, TRY P
002362  000137  005420                       JMP    BEGINC                  ;GO TO CALIBRATION TEST
002366  121027  000120               3$:     CMPB   (R0),#'P                ;IS IT P?
002372  001002                               BNE    4$              ;;NO, TRY L
002374  000137  005674                       JMP    BEGINP                  ;GO TO DISPLAY CONVERSIONS TEST
002400  121027  000114               4$:     CMPB   (R0),#'L                ;IS IT L?
002404  001002                               BNE    5$              ;;NO, TRY W
002406  000137  006064                       JMP    BEGL                    ;GO TO LOGIC TFSTS
002412  121027  000127               5$:     CMPB   (R0),#'W                ;IS IT W?
002416  001002                               BNE    6$              ;;NO,TRY AGAIN
002420  000137  006204                       JMP    BEGINW                  ;GO TO WRAPAROUND TEST
002424  104401  012105               6$:     TYPE   ,QUEST
002430  000731                               BR     TRYAG                   ;WAIT FOR CHARACTER
```

```
     002432  005000                          VTFLG:  CLR     R0               ;TEST FOR PRESENCE
     002434  105777  176504                  1$:     TSTB    @$TKS            ;OF VT55
     002440  100404                                  BMI     2$               ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
     002442  005300                                  DEC     R0
     002444  001373                                  BNE     1$               ;;
     002446  005726                                  TST     (SP)+            ;POP A WORD OFF STACK
     002450  000660                                  BR      NOVT55           ;;NO VT55 PRESENT
     002452  017700  176470                  2$:     MOV     @$TKB,R0
     002456  042700  177600                          BIC     #177600,R0               ;TEST VT55 CODE
     002462  000207                                  RTS     PC

     002464  005037  001202                  TESTAD: CLR     $PASS            ;CLEAR PASS COUNT
     002470  )05037  001436                          CLR     GUNITS           ;CLEAR UNIT ERROR BITS
     002474  012737  000001  001440                  MOV     #1,TSTBIT        ;INITIALIZE MODULE ERROR TEST BIT
     002502  012737  000001  001356                  MOV     #1,TEMP          ;SET UP FOR ONLY ONE A/D
     002510  105737  001215                          TSTB    $ENVM            ;TESTING ONLY ONE A/D?
     002514  100411                                  BMI     3$               ;;YES
     002516  012737  000004  001356                  MOV     #4,TEMP          ;SET UP MAX NO OF A/D'S
     002524  005737  001426                          TST     WFTEST           ;IS IT IN OPTION TEST
     002530  001403                                  BEQ     3$               ;;NOT IN OPTION TEST
     002532  012737  000020  001356                  MOV     #16.,TEMP        ;SET UP OPTION MAX NO OF A/D'S
     002540  013737  001250  001126  3$:             MOV     $BASE,$BDDAT     ;SETUP TO TEST FOR ADV11'S
     002546  013746  000004                          MOV     @#ERRVEC,-(SP)   ;SAVE ERRVEC
     002552  012737  002624  000004                  MOV     #2$,ERRVEC       ;SET UP FOR TIME OUT ERROR
     002560  005037  001364                          CLR     NBEXT            ;CLEAR ADV11 COUNTER
     002564  005777  176336                  1$:     TST     @$BDDAT          ;ADDRESS ADV11
     002570  005237  001364                          INC     NBEXT            ;INCREMENT ADV11 COUNTER
     002574  053737  001440  001436                  BIS     TSTBIT,GUNITS    ;SET A/D BIT UNDER TEST
     002602  006337  001440                          ASL     TSTBIT .         ;SET TEST BIT FOR NEXT UNIT
     002606  005337  001356                          DEC     TEMP             ;REACHED MAX?
     002612  001405                                  BEQ     4$               ;;REACHED MAX NO OF A/D'S
     002614  063737  001336  001126                  ADD     VADR,$BDDAI      ;GET NEXT ADV11
     002622  000760                                  BR      1$               ;;TRY NEXT ADV11
     002624  022626                          2$:     CMP     (SP)+,(SP)+      ;POP 2 WORDS OFF STACK
     002626                                  4$:
(1)  002626  013746  001364                          MOV     NBEXT,-(SP)      ;;SAVE NBEXT FOR TYPEOUT
(1)                                                                           ;;TYPE NUMBER OF ADV11'S
(1)  002632  104403                                  TYPOS                    ;;GO TYPE--OCTAL ASCII
(1)  002634     002                                  .BYTE   2                ;;TYPE 2 DIGIT(S)
(1)  002635     000                                  .BYTE   0                ;;SUPPRESS LEADING ZEROS
     002636  104401  013:57                          TYPE    ,MSG50
     002642  005337  001364                          DEC     NBEXT            ;ADJUST ADV11 COUNT
     002646  013737  001364  001366                  MOV     NBEXT,NMBEXT     ;KEEP COUNT OF NUMBER
     002654  012637  000004                          MOV     (SP)+,ERRVEC     ;RESTORE ERRVEC
     002660  012737  000001  001440                  MOV     #1,TSTBIT        ;INITIALIZE MODULE ERROR TEST BIT
     002666  000207                                  RTS     PC
```

```
  270  002670                             BEGINL:
  271                                     ;;*****************************************************************
  (3)                                     ;*TEST 1         FLOAT A ONE THRU MULTIPLEXER BITS
  (3)                                     ;;*****************************************************************
  (2)  002670  012737  002670  001106     TST1:   MOV     #TST1,$LPADR
  272  002676  012737  002670  001110             MOV     #TST1,$LPERR
  273  002704  012737  000400  001124             MOV     #BIT8,$GDDAT             ;LOAD FIRST BIT
  274  002712  104412                     2$:     CHKIT
  275  002714  104001                             ERROR   1                        ;FAILED TO LOAD + READ BIT
  276  002716  006337  001124             1$:     ASL     $GDDAT                   ;GET NEXT BIT
  277  002722  023727  001124  010000             CMP     $GDDAT,#BIT12            ;FINISHED?
  278  002730  001370                             BNE     2$               ;;NO,GO TO NEXT TEST
  279
  280                                     ;;*****************************************************************
  (3)                                     ;*TEST 2         LOAD AND READ BACK ERROR I.E. BIT14
  (3)                                     ;;*****************************************************************
  (2)  002732  000004                     TST2:   SCOPE
  281  002734  012737  040000  001124             MOV     #BIT14,$GDDAT
  282  002742  104412                             CHKIT
  283  002744  104001                             ERROR   1                        ;FAILED TO LOAD + READ ERROR I.E.
  284                                     ;;*****************************************************************
  (3)                                     ;*TEST 3         LOAD AND READ BACK INTERRUPT ENABLE BIT6
  (3)                                     ;;*****************************************************************
  (2)  002746  000004                     TST3:   SCOPE
  285  002750  012777  001442  176346             MOV     #UNEXP,@VECTOR           ;SETUP FOR UNEXPECTED INTERUPT
  286  002756  012737  000100  001124             MOV     #BIT6,$GDDAT             ;LOAD EXPECTED DATA
  287  002764  104412                             CHKIT
  288  002766  104001                             ERROR   1                        ;FAILED TO LOAD + READ INTERRUPT ENABLE
  289
  290                                     ;;*****************************************************************
  (3)                                     ;*TEST 4         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
  (3)                                     ;;*****************************************************************
  (2)  002770  000004                     TST4:   SCOPE
  291  002772  012737  000040  001124             MOV     #BIT5,$GDDAT             ;LOAD EXPECTED DATA
  292  003000  104412                             CHKIT
  293  003002  104001                             ERROR   1                        ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB
  294
  295                                     ;;*****************************************************************
  (3)                                     ;*TEST 5         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
  (3)                                     ;;*****************************************************************
  (2)  003004  000004                     TST5:   SCOPE
  296  003006  012737  000020  001124             MOV     #BIT4,$GDDAT     ;LOAD EXPECTED DATA
  297  003014  104412                             CHKIT
  298  003016  104001                             ERROR   1                ;FAILED TO LOAD + READ EXT. START ENABLE
  299                                     ;;*****************************************************************
  (3)                                     ;*TEST 6         LOAD AND READ BACK MAINT. TST BIT2
  (3)                                     ;;*****************************************************************
  (2)  003020  000004                     TST6:   SCOPE
  300  003022  012737  000004  001124             MOV     #BIT2,$GDDAT
  301  003030  104412                             CHKIT
  302  003032  104001                             ERROR   1                        ;FAILED TO LOAD + READ BACK MAINT. TST
```

```
                                          ;;*************************************************************
                                          ;*TEST 7        LOAD AND READ BACK ENABLE I.D. BIT3
                                          ;;*************************************************************
003034  000004                            TST7:   SCOPE
003036  012737  000010  001124                    MOV     #BIT3,$GDDAT
003044  104412                                    CHKIT
003046  104001                                    ERROR   1                       ;FAILED TO LOAD + READ ENABLE I.D. BIT
                                          ;;*************************************************************
                                          ;*TEST 10       TEST I.D. BIT (BIT 12) CLEARED
                                          ;;*************************************************************
003050  000004                            TST10:  SCOPE
003052  012777  000001  176236                    MOV     #1,@STREG       ;CLEAR I.D. ENABLE
003060  105777  176232                    1$:     TSTB    @STREG          ;WAIT FOR CONVERSION
003064  100375                                    BPL     1$              ;;CONVERSION IS NOT DONE YET
003066  017737  176230  001126                    MOV     @ADBUFF,$BDDAT  ;READ BUFFER
003074  005037  001124                            CLR     $GDDAT          ;CLEAR EXPECTED
003100  032737  010000  001126                    BIT     #BIT12,$BDDAT   ;IS I.D. BIT CLEARED?
003106  001401                                    BEQ     TST11           ;;YES - GOTO NEXT TEST
003110  104001                                    ERROR   1


                                          ;;*************************************************************
                                          ;*TEST 11       TEST I.D. BIT (BIT 12) SET
                                          ;;*************************************************************
003112  000004                            TST11:  SCOPE
003114  012777  000011  176174                    MOV     #BIT3!BIT0,@STREG       ;SET I.D. ENABLE BIT
003122  105777  176170                    1$:     TSTB    @STREG          ;WAIT FOR CONVERSION
003126  100375                                    BPL     1$              ;;CONVERSION IS NOT DONE YET
003130  017737  176166  001126                    MOV     @ADBUFF,$BDDAT  ;READ BUFFER
003136  012737  010000  001124                    MOV     #BIT12,$GDDAT   ;LOAD EXPECTED
003144  032737  010000  001126                    BIT     #BIT12,$BDDAT   ;IS I.D. BIT SET?
003152  001001                                    BNE     TST12           ;;YES - GOTO NEXT TEST
003154  104001                                    ERROR   1


                                          ;;*************************************************************
                                          ;*TEST 12       LOAD AND READ BACK ERROR FLAG BIT15
                                          ;;*************************************************************
003156  000004                            TST12:  SCOPE
003160  012737  100000  001124                    MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
003166  104412                                    CHKIT
003170  104001                                    ERROR   1               ;FAILED TO LOAD + READ ERROR FLAG
                                          ;;*************************************************************
                                          ;*TEST 13       TEST INIT CLEARS BITS 2-6,8-11,14
                                          ;;*************************************************************
003172  000004                            TST13:  SCOPE
003174  012737  000300  001160                    MOV     #300,$TIMES     ;;DO 300 ITERATIONS
003202  005037  001124                            CLR     $GDDAT          ;LOAD EXPECTED DATA
003206  012777  047574  176102            2$:     MOV     #47574,@STREG   ;SET STATUS REGISTER
003214  000005                                    RESET                   ;INITIALIZE
003216  052777  000100  175720                    BIS     #100,@$TKS      ;SET INTRPT. ENABLE
003224  017737  176066  001126                    MOV     @STREG,$BDDAT   ;READ STATUS REGISTER
003232  001401                                    BEQ     TST14           ;;NEXT TEST
003234  104001                                    ERROR   1               ;RESET FAILED TO CLEAR AD ST. REG. BITS
```

```
   342                                          ;;*******************************************************************
   (3)                                          ;*TEST 14       TEST INIT CLEARS ERROR FLAG
   (3)                                          ;;*******************************************************************
   (2)  003236  000004                          TST14:  SCOPE
   343  003240  012737  000300  001160                  MOV     #300,$TIMES             ;DO 300 ITERATIONS
   344  003246  012777  100000  176042                  MOV     #BIT15,@STREG           ;SET BIT 15
   345  003254  000005                                  RESET                           ;ISSUE INIT
   346  003256  052777  000100  175660                  BIS     #100,@$TKS                      ;SET INTRPT. EN. FOR KEYBOARD
   347  003264  104411                                  CHECK
   348  003266  104001                                  ERROR   1
   349                                          ;;*******************************************************************
   (3)                                          ;*TEST 15       TEST  DONE FLAG  SETS AND BIT0 CLEARS ON END OF CONV.
   (3)                                          ;;*******************************************************************
   (2)  003270  000004                          TST15:  SCOPE
   350  003272  012700  001000                          MOV     #BIT9,R0        ;STALL TIME COUNTER
   351  003276  005277  176014                          INC     @STREG          ;START CONVERSION
   352  003302  012737  000200  001124                  MOV     #BIT7,$GDDAT    ;LOAD EXPECTED
   353  003310  005300                          1$:     DEC     R0              ;STALL
   354  003312  001376                                  BNE     1$              ;TIME
   355  003314  042777  100000  175774                  BIC     #BIT15,@STREG   ;MASK OUT ERROR BIT
   356  003322  104411                                  CHECK
   357  003324  104001                                  ERROR   1               ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
   358  003326  017700  175770                          MOV     @ADBUFF,R0      ;CLEAR DONE FLAG FOR ITERATIONS
```

```
 360                                   ;;*****************************************************************
 (3)                                   ;*TEST 16        TEST INIT CLEARS DONE FLAG
 (3)                                   ;;*****************************************************************
 (2)  003332  000004                   TST16:   SCOPE
 (1)  003334  012737  000300  001160            MOV     #300,$TIMES       ;;DO 300 ITERATIONS
 361  003342  005037  001124                    CLR     $GDDAT            ;CLEAR EXPECTED
 362  003346  005277  175744                    INC     @STREG            ;START CONVERSION
 363  003352  105777  175740           2$:      TSTB    @STREG
 364  003356  100375                            BPL     2$
 365  003360  000005                            RESET
 366  003362  104411                            CHECK
 367  003364  104001                            ERROR   1                 ;DONE FLAG FAILED TO CLEAR
 368  003366  052777  000100  175550            BIS     #100,@$TKS                  ;SET INTRPT. EN. BIT
 369
 370                                   ;;*****************************************************************
 (3)                                   ;*TEST 17        TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
 (3)                                   ;;*****************************************************************
 (2)  003374  000004                   TST17:   SCOPE
 371  003376  005277  175714                    INC     @STREG            ;SET A/D START CONVERSION BIT
 372  003402  105777  175710           1$:      TSTB    @STREG            ;WAIT FOR FLAG
 373  003406  100375                            BPL     1$
 374  003410  017700  175706                    MOV     @ADBUFF,R0        ;READ CONVERTED VALUE
 375  003414  104411                            CHECK
 376  003416  104001                            ERROR   1                 ;DONE FLAG FAILED TO CLEAR
 377                                   ;;*****************************************************************
 (3)                                   ;*TEST 20        TEST ALL '0'S RESULTS USING MAINT. ADTST. BIT
 (3)                                   ;;*****************************************************************
 (2)  003420  000004                   TST20:   SCOPE
 378  003422  005037  001124                    CLR     $GDDAT            ;CLEAR EXPECTED VALUE
 379  003426  005037  001372                    CLR     CHANL             ;SET CHANL = 0
 380  003432  005037  001412                    CLR     SPREAD            ;SET SPREAD = 0
 381  003436  012777  000005  175652            MOV     #5,@STREG         ;CONVERT EVEN CHANNEL WITH MAINT. BIT SET
 382  003444  105777  175646           1$:      TSTB    @STREG            ;WAIT FOR DONE
 383  003450  100375                            BPL     1$
 384  003452  017737  175644  001126            MOV     @ADBUFF,$BDDAT    ;RESULTS TO BDDAT FOR CHECKING
 385  003460  001401                            BEQ     TST21             ;;GOTO NEXT TEST
 386  003462  104004                            ERROR   4                 ;DID NOT GET ALL '0'S RSULT WITH MAINT. ADTST
 387
 388                                   ;;*****************************************************************
 (3)                                   ;*TEST 21        TEST ALL '1'S RESULT USING MAINT. ADTST. BIT
 (3)                                   ;;*****************************************************************
 (2)  003464  000004                   TST21:   SCOPE
 389  003466  012737  007777  001124            MOV     #7777,$GDDAT      ;EXPECT ALL '1'S RESULT
 390  003474  012737  000001  001372            MOV     #1,CHANL          ;SET CHANL = 1
 391  003502  005037  001412                    CLR     SPREAD            ;SET SPREAD = 0
 392  003506  012777  000405  175602            MOV     #405,@STREG       ;CONVERT ODD CHANNEL WITH MAINT. BIT SET
 393  003514  105777  175576           1$:      TSTB    @STREG            ;WAIT FOR DONE
 394  003520  100375                            BPL     1$
 395  003522  017737  175574  001126            MOV     @ADBUFF,$BDDAT    ;RESULTS TO BDDAT FOR CHECKING
 396  003530  023737  001124  001126            CMP     $GDDAT,$BDDAT     ;EQUAL?
 397  003536  001401                            BEQ     TST22             ;;GOTO NEXT TEST
 398  003540  104004                            ERROR   4                 ;DID NOT GET ALL '1'S RESULT WITH MAINT. ADTST
```

```
   400                                     ;;*************************************************************************
   (3)                                     ;*TEST 22        GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
   (3)                                     ;;*************************************************************************
   (2)  003542  000004                     TST22:  SCOPE
   401                                             ;* "ENTERING TEST 22" TYPED OUT TO TELL YOU THE NEXT
   (1)                                             ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
   (1)                                             ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
   (1)                                             ;*EXECUTING TEST "22".
   (1)  003544  012700  000022                     MOV     #22,R0          ;GET TEST NO.
   (1)  003550  004737  011254                     JSR     PC,DUMW         ;PRINT MESSAGE
   402  003554  005046                             CLR     -(SP)           ;RESET PRIORITY
   403  003556  012746  003564                     MOV     #3$,-(SP)
   404  003562  000002                             RTI
   405  003564  012777  003640  175532  3$:        MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
   406  003572  012777  000200  175530             MOV     #200,@VECTR1    ;SET UP NEW PSW
   407  003600  012777  000101  175510             MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
   408  003606  105777  175504          2$:        TSTB    @STREG          ;WAIT FOR DONE
   409  003612  100375                             BPL     2$              ;FLAG TO SET
   410  003614  017737  175476  001126             MOV     @STREG,$BDDAT   ;READ STATUS REGISTER
   411  003622  012737  000300  001124             MOV     #BIT7!BIT6,$GDDAT ;GOOD DATA
   412  003630  104002                             ERROR   2               ;FAILED TO INTERRUPT ON DONE
   413  003632  004737  011326                     JSR     PC,DUMC         ;TYPE COMPLETED
   414  003636  000414                             BR      TST23           ;;BRANCH TO NEXT TEST
   415  003640  022626                  1$:        CMP     (SP)+,(SP)+     ;RESET STACK POINTER
   416  003642  012777  001442  175454             MOV     #UNEXP,@VECTOR  ;SET UP FOR UNEXPECTED INTERRUPT
   417  003650  005046                             CLR     -(SP)           ;CLEAR PSW
   418  003652  012746  003660                     MOV     #4$,-(SP)
   419  003656  000002                             RTI
   420  003660  004737  011326          4$:        JSR     PC,DUMC         ;TYPE COMPLETED
   421  003664  005777  175432                     TST     @ADBUFF         ;CLEAR DONE BIT
   422                                     ;;*************************************************************************
   (3)                                     ;*TEST 23        TEST INTERRUPT OCCURS WHEN ERROR AND I.E.E. IS SET
   (3)                                     ;;*************************************************************************
   (2)  003670  000004                     TST23:  SCOPE
   423                                             ;* "ENTERING TEST 23" TYPED OUT TO TELL YOU THE NEXT
   (1)                                             ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
   (1)                                             ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
   (1)                                             ;*EXECUTING TEST "23".
   (1)  003672  012700  000023                     MOV     #23,R0          ;GET TEST NO.
   (1)  003676  004737  011254                     JSR     PC,DUMW         ;PRINT MESSAGE
   424  003702  012777  003742  175422             MOV     #1$,@VECTR2     ;SETUP VECTOR ADDRESS
   425  003710  012777  140000  175400             MOV     #BIT15!BIT14,@STREG     ;CAUSE AN INTERRUPT
   426  003716  017737  175374  001126             MOV     @STREG,$BDDAT   ;BAD DATA
   427  003724  012737  140000  001124             MOV     #BIT15!BIT14,$GDDAT     ;GOOD DATA
   428  003732  104002                             ERROR   2
   429  003734  004737  011326                     JSR     PC,DUMC         ;TYPE COMPLETED
   430  003740  000627                             BR      TST20
   431  003742  022626                  1$:        CMP     (SP)+,(SP)+     ;POP STACK
   432  003744  004737  011326                     JSR     PC,DUMC
   433  003750  005077  175342                     CLR     @STREG
```

```
 435                                       ;;*****************************************************************
 (3)                                       ;*TEST 24        TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
 (3)                                       ;;*****************************************************************
 (2)  003754  000004                       TST24:  SCOPE
 436  003756  012777  000001  175332               MOV     #BIT0,@STREG    ;START CONVERSION
 437  003764  105777  175326               1$:     TSTB    @STREG          ;WAIT FOR
 438  003770  100375                               BPL     1$
 439  003772  012737  100200  001124       2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
 440  004000  012777  000001  175310               MOV     #BIT0,@STREG    ;START 2ND CONVERSION
 441  004006  012700  001000                       MOV     #BIT9,R0        ;WAIT FOR 2ND
 442  004012  005300                       3$:     DEC     R0              ;CONVERSION TO END
 443  004014  001376                               BNE     3$
 444  004016  104411                       4$:     CHECK
 445  004020  104001                               ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
 446                                                                       ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
 447  004022  017700  175274                       MOV     @ADBUFF,R0      ;CLEAR DONE FLAG
 448                                       ;;*****************************************************************
 (3)                                       ;*TEST 25        TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
 (3)                                       ;;*****************************************************************
 (2)  004026  000004                       TST25:  SCOPE
 449  004030  012737  100000  001124               MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
 450  004036  012777  000001  175252               MOV     #BIT0,@STREG    ;START CONVERSION
 451  004044  112777  000001  175244               MOVB    #BIT0,@STREG            ;START NEXT CONVERSION
 452  004052  112777  000001  175236               MOVB    #BIT0,@STREG            ;ONCE AGAIN IN CASE REFRESH INTERVENED
 453  004060  017737  175232  001126               MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
 454  004066  042737  077777  001126               BIC     #77777,$BDDAT           ;MASK OUT BIT 15
 455  004074  023737  001124  001126               CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
 456  004102  001401                               BEQ     1$              ;;BRANCH OVER ERROR
 457  004104  104001                               ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
 458                                                                       ;CONVERT BEGINS BEFORE FIRST DONE
 459  004106  105777  175204               1$:     TSTB    @STREG                  ;WAIT FOR DONE
 460  004112  100375                               BPL     1$
 461  004114  017700  175202                       MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
 462  004120  005077  175172                       CLR     @STREG                  ;CLEAR STATUS REGISTER
 463  004124  000004                               SCOPE
 464  004126  000207                               RTS     PC              ;RETURN TO TEST SECTION
 465
 466
 467                                       ;;SUBROUTINE FOR LOGIC TESTS;;
 468  004130  013777  001124  175160       TESTIT: MOV     $GDDAT,@STREG           ;LOAD EXPECTED VALUF
 469  004136  017737  175154  001126       TEST:   MOV     @STREG,$BDDAT           ;READ ST. REG.
 470  004144  023737  001124  001126               CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
 471  004152  001002                               BNE     RETERR          ;;ERROR RETURN
 472  004154  062716  000002                       ADD     #2,(SP)                 ;BUMP RETURN ADDRESS TO GET AROUND ERROR
 473  004160  000002                       RETERR: RTI
```

```
                                            .SBTTL          WRAPAROUND TEST SECTION
      004162                                WRAP:
                                            ;;*****************************************************************
(3)                                         ;*TEST 26       TEST CH0 GROUND
(3)                                         ;;*****************************************************************
(2)   004162  012737  000026  001102        TST26:  MOV     #$TN,$TSTNM
(1)   004170  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      004176  012737  004162  001110                MOV     #TST26,$LPERR
      004204  012737  004162  001106                MOV     #TST26,$LPADR
      004212  004537  011074                        JSR     R5,CONVRT               ;CONVERT 8 TIMES
      004216  000000                                0
      004220  004537  011206                        JSR     R5,COMPAR               ;COMPARE RESULTS
      004224  004000                                4000                            ;NOMINAL
      004226  011654                                V12                             ;TOLERANCE
      004230  104004                                ERROR   4                       ;ERROR ON A/D CHANNEL
                                            ;;*****************************************************************
(3)                                         ;*TEST 27       TEST CH1 +4.5 VOLT
(3)                                         ;;*****************************************************************
(2)   004232  000004                        TST27:  SCOPE
(1)   004234  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      004242  004537  011074                        JSR     R5,CONVRT               ;CONVERT 8 TIMES
      004246  000001                                1                               ;CHANNEL 1
      004250  004537  011206                        JSR     R5,COMPAR               ;COMPARE RESULTS
      004254  007344                                7344                            ;NOMINAL
      004256  011660                                V326                            ;TOLERANCE
      004260  104004                                ERROR   4                       ;ERROR ON A/D CHANNEL

                                            ;;*****************************************************************
(3)                                         ;*TEST 30       TEST CH2 -4.5 VOLT
(3)                                         ;;*****************************************************************
(2)   004262  000004                        TST30:  SCOPE
(1)   004264  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      004272  004537  011074                        JSR     R5,CONVRT               ;CONVERT 8 TIMES
      004276  000002                                2                               ;CHANNEL 2
      004300  004537  011206                        JSR     R5,COMPAR               ;COMPARE RESULTS
      004304  000434                                434                             ;NOMINAL
      004306  011660                                V326                            ;TOLERANCE
      004310  104004                                ERROR   4                       ;ERROR ON A/D CHANNEL
                                            ;;*****************************************************************
(3)                                         ;*TEST 31       TEST GROUND ON CHANNELS 4 - 17
(3)                                         ;;*****************************************************************
(2)   004312  000004                        TST31:  SCOPE
(1)   004314  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      004322  012737  000004  004334                MOV     #4,2$           ;SET UP FIRST CHANNEL
      004330  004537  011074                1$:     JSR     R5,CONVRT       ;CONVERT CHANNEL
      004334  000004                        2$:     4
      004336  004537  011206                        JSR     R5,COMPAR       ;TEST RESULTS
      004342  004000                                4000
      004344  011654                                V12
      004346  104004                                ERROR   4
      004350  005237  004334                        INC     2$              ;GET NEXT CHANNEL
      004354  022737  000017  004334                CMP     #17,2$          ;DONE?
      004362  001362                                BNE     1$              ;;NO
```

```
                                        ;;*************************************************************
                                        ;*TEST 32        TEST VERNIER OFFSET DAC ON CH0
                                        ;;*************************************************************
004364  000004                          TST32:  SCOPE
004366  012737  000001  001160                  MOV     #1,$TIMES        ;;DO 1 ITERATION
004374  005077  174722                          CLR     @ADBUFF                  ;SET VERNIER DAC = 0
004400  004537  011074                          JSR     R5,CONVRT                ;CONV. CH0, DIRECT VERNIER DAC
004404  000000                                  0
004406  013704  001356                          MOV     TEMP,R4                  ;SAVE VALUE IN R4
004412  012777  000377  174702  1$:             MOV     #377,@ADBUFF             ;SET VERNIER DAC = 377
004420  004537  011074                          JSR     R5,CONVRT                ;CONVERT IT
004424  000000                                  0
004426  160437  001356                          SUB     R4,TEMP                  ;TEMP=DIFF. BETWEEN VALUE & PREVIOUS
004432  004537  011206                          JSR     R5,COMPAR                ;COMPARE RESULTS
004436  000005                                  5
004440  011650                                  V2
004442  104004                                  ERROR   4
                                        ;;*************************************************************
                                        ;*TEST 33        OFFSET ON CH0
                                        ;;*************************************************************
004444  000004                          TST33:  SCOPE
004446  012737  000001  001160                  MOV     #1,$TIMES        ;;DO 1 ITERATION
004454  013737  001342  001372                  MOV     BASECH,CHANL             ;LOAD CHANNEL
004462  013737  001342  001370                  MOV     BASECH,DUMMY             ;LOAD DUMMY
004470  004737  005214                          JSR     PC,OFFSET        ;FIND OFFSET
004474  104401  014014                          TYPE    ,MOFSET          ;TYPE "OFFSET="
004500  004737  005270                          JSR     PC,TOFF          ;TYPE OFFSET
004504  004537  011206                          JSR     R5,COMPAR        ;IS RESULT WITHIN LIMITS?
004510  000000                                  0
004512  011656                                  V50D
004514  000401                                  BR      OFFERR           ;NO-ERROR
004516  000403                                  BR      OFFOK            ;YES-OK
004520  104401  012625                  OFFERR: TYPE    ,ERMSG
004524  000402                                  BR      TST34            ;;GO TO NEXT TEST
004526  104401  012303                  OFFOK:  TYPE    ,OKMSG
```

```
(3)                                       ;;*******************************************************************
(3)                                       ;*TEST 34        TEST RAMP RANGE, CH3
(3)                                       ;;*******************************************************************
(2)  004532  000004                       TST34:  SCOPE
     004534  012737  000001  001160               MOV     #1,$TIMES                       ;DO THIS ONCE
     004542  012703  007777                       MOV     #7777,R3                        ;INIT R3 VALUE
     004546  005004                               CLR     R4                              ;AND R4
     004550  012777  001400  174540               MOV     #1400,@STREG                    ;SETUP FOR CH3
     004556  012702  047040                       MOV     #20000.,R2                      ;SETUP FOR 20,000 CONVERSIONS
     004562  105277  174530           1$:         INCB    @STREG
     004566  105777  174524           2$:         TSTB    @STREG
     004572  100375                               BPL     2$
     004574  027704  174522                       CMP     @ADBUFF,R4
     004600  003402                               BLE     3$
     004602  017704  174514                       MOV     @ADBUFF,R4                      ;HIT A NEW HIGH
     004606  027703  174510           3$:         CMP     @ADBUFF,R3
     004612  002002                               BGE     4$
     004614  017703  174502                       MOV     @ADBUFF,R3                      ;HIT A NEW LOW
     004620  005302                      4$:         DEC     R2
     004622  001357                               BNE     1$
     004624  010337  001356                       MOV     R3,TEMP
     004630  004537  011206                       JSR     R5,COMPAR
     004634  000000                               0
     004636  011646                               V0
     004640  104004                               ERROR   4                    ;RAMP DIDN'T REACH LOW END OF RANGE
     004642  010437  001356                       MOV     R4,TEMP
     004646  004537  011206                       JSR     R5,COMPAR
     004652  007777                               7777
     004654  011646                               V0
     004656  104004                               ERROR   4                    ;RAMP DIDN'T REACH HIGH END OF RANGE
```

```
  569                                              ;;*****************************************************************
  (3)                                              ;*TEST 35        NOISE TEST, 1 EDGE
  (3)                                              ;;*****************************************************************
  (2)   004660  000004                             TST35:   SCOPE
  (1)   004662  012737  000001  001160                      MOV     #1,$TIMES           ;;DO 1 ITERATION
  570   004670  104401  012042                              TYPE    ,NOIMSG
  571   004674  005037  001372                              CLR     CHANL               ;LOAD CHANNEL 0
  572   004700  013737  001372  001370  1$:                 MOV     CHANL,DUMMY                 ;LOAD DUMMY CHANNEL
  573   004706  004737  006764                              JSR     PC,GETEDG                   ;GET EDGE VALUE
  574   004712  005037  001404                              CLR     RMS                         ;CLEAR RMS VLAUE
  575   004716  005037  001406                              CLR     PEAK                        ;CLEAR PEAK VALUE
  576   004722  004537  007144                              JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 16%
  577   004726  000020                                      16.
  578   004730  063737  001414  001404                      ADD     DAC,RMS             ;ADD RESULT TO RMS
  579   004736  004537  007144                              JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 84%
  580   004742  000124                                      84.
  581   004744  163737  001414  001404                      SUB     DAC,RMS             ;SUBTRACT RESULT FROM RMS
  582   004752  004537  007144                              JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 1%
  583   004756  000001                                      1
  584   004760  063737  001414  001406                      ADD     DAC,PEAK                    ;ADD RESULT TO PEAK
  585   004766  004537  007144                              JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 99%
  586   004772  000143                                      99.
  587   004774  163737  001414  001406                      SUB     DAC,PEAK                    ;SUBTRACT RESULT FROM PEAK
  588   005002  012737  000001  007142                      MOV     #1,EDGFLG
  589   005010  004737  010744                              JSR     PC,TYPRP                    ;TYPE RMS AND PEAK VALUES
  590   005014  005237  001372                              INC     CHANL                       ;GET NEXT CHANNEL
  591   005020  022737  000003  001372                      CMP     #3,CHANL                    ;CHANNEL 3?
  592   005026  001002                                      BNE     2$                  ;;NO
  593   005030  005237  001372                              INC     CHANL               ;CHANNEL 3 IS SKIPED
  594   005034  022737  000017  001372  2$:                 CMP     #17,CHANL           ;DONE?
  595   005042  001316                                      BNE     1$                  ;;NO
```

```
   597                                          ;;*****************************************************************
   (3)                                          ;*TEST 36       INTERCHANNEL SETTLING TEST, 1 EDGE
   (3)                                          ;;*****************************************************************
   (2)   005044  000004                         TST36:  SCOPE
   (1)   005046  012737  000001  001160                 MOV     #1,$TIMES       ;;DO 1 ITERATION
   598   005054  104401  012061                         TYPE    ,SETMSG         ;TYPE "SETTLING TEST"
   599   005060  012737  000001  001360                 MOV     #1,CH1          ;DO TEST BETWEEN CHANNEL 1 AND 2
   600   005066  012737  000002  001362                 MOV     #2,CH2
   601   005074  013737  001362  001372  1$:    MOV     CH2,CHANL
   602   005102  004737  006764                         JSR     PC,GETEDG       ;GET EDGE VALUES
   603   005106  005002                                 CLR     R2
   604   005110  004737  006722                         JSR     PC,SET1A        ;SCALING = .02 LSB
   605   005114  004737  006722                         JSR     PC,SET1A        ;MAKE IT .01 LSB
   606   005120  100001                                 BPL     2$
   607   005122  005402                                 NEG     R2              ;MAKE IT POSITIVE
   608   005124  010204                         2$:     MOV     R2,R4
   609   005126  012737  000001  007142                 MOV     #1,EDGFLG
   610   005134  004737  006572                         JSR     PC,TYPSET       ;TYPE SETTLING INFORMATION
   611   005140  022737  000002  001360                 CMP     #2,CH1          ;DONE?
   612   005146  001410                                 BEQ     TST37           ;;YES
   613   005150  013702  001360                         MOV     CH1,R2          ;SETTLE THE OTHER WAY
   614   005154  013737  001362  001360                 MOV     CH2,CH1
   615   005162  010237  001362                         MOV     R2,CH2
   616   005166  000742                                 BR      1$              ;;
   617   005170                                 3$:
   618                                          ;;*****************************************************************
   (3)                                          ;*TEST 37       DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
   (3)                                          ;;*****************************************************************
   (2)   005170  000004                         TST37:  SCOPE
   (1)   005172  012737  000001  001160                 MOV     #1,$TIMES       ;;DO 1 ITERATION
   619   005200  005737  001202                         TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
   620   005204  001402                                 BEQ     LEND
   621   005206  004737  007344                         JSR     PC,DIFLIN
   622   005212  000207                         LEND:   RTS     PC                      ;RETURN TO TEST SECTION
   623
   624   005214  012737  004001  001420         OFFSET: MOV     #4001,EDGE              ;4000,4001 EDGE
   625   005222  004537  007144                         JSR     R5,SARSUB
   626   005226  000062                                 50.
   627   005230  013737  001414  001356                 MOV     DAC,TEMP
   628   005236  012737  004000  001420                 MOV     #4000,EDGE              ;3777,4000 EDGE
   629   005244  004537  007144                         JSR     R5,SARSUB
   630   005250  000062                                 50.
   631   005252  063737  001414  001356                 ADD     DAC,TEMP
   632   005260  162737  000400  001356                 SUB     #400,TEMP
   633   005266  000207                                 RTS     PC
```

```
005270  013702  001356          TOFF:   MOV     TEMP,R2
005274  100402                          BMI     1$              ;;IS THE NUMBER POSITIVE?
005276  104401  012623                  TYPE    ,POSITV
005302  104413                  1$:     TYPDC
005304  104401  014027                  TYPE    ,MLSB           ;TYPE ASCIZ STRING
005310  000207                          RTS     PC
005312  005303                  TCHK:   DEC     R3              ;DECREMENT COUNT
005314  001005                          BNE     1$              ;;
005316  012703  000005                  MOV     #5,R3           ;RESET COUNT
005322  104401  001171                  TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
005326  000402                          BR      2$              ;;
005330  104401  012172          1$:     TYPE    ,SPACE          ;TYPE FOUR (4) SPACES
005334  005037  001416          2$:     CLR     DELAY           ;CLEAR DELAY
005340  005077  173600                  CLR     @$TKS           ;CLEAR INTERRUPT ENABLE
005344  105777  173574          3$:     TSTB    @$TKS           ;IS KEYBOARD FLAG SET?
005350  100404                          BMI     4$              ;;YES
005352  005237  001416                  INC     DELAY           ;IS DELAY ZERO?
005356  001372                          BNE     3$              ;;NO
005360  000416                          BR      6$              ;;
005362  005777  173560          4$:     TST     @$TKB           ;CLEAR FLAG
005366  012777  000100  173550          MOV     #100,@$TKS      ;SET INTERRUPT ENABLE
005374  004537  011206                  JSR     R5,COMPAR       ;TEST LAST CONVERSION
005400  000000                          0
005402  011652                          V4                      ;TOLERANCE .04 LSB
005404  000402                          BR      5$              ;;
005406  062716  000002                  ADD     #2,(SP)         ;BUMP RETURN ADDRESS
005412  062716  000002          5$:     ADD     #2,(SP)         ;BUMP RETURN ADDRESS 2 WORDS
005416  000207                  6$:     RTS     PC
005420  104401  012314          BEGINC: TYPE    ,CCHAN          ;ASK FOR CHANNEL
005424  104410                          RDOCT                   ;READ CHANNEL NUMBER
005426  012637  001372                  MOV     (SP)+,CHANL     ;STORE CHANNEL NUMBER
005432  013737  001372  001370          MOV     CHANL,DUMMY     ;LOAD DUMMY
005440  104401  012342          1$:     TYPE    ,SEL            ;SELECT OFFSET OR GAIN ADJUST
005444  104407                          RDLIN                   ;GET TEST
005446  012600                          MOV     (SP)+,R0        ;MOVE POINTER TO R0
005450  121027  000117                  CMPB    (R0),#'O        ;IS IT "O"?
005454  001406                          BEQ     AJOFF           ;;YES, GO TO ADJUST OFFSET
005456  121027  000107                  CMPB    (R0),#'G        ;IS IT "G"?
005462  001430                          BEQ     AJGAIN          ;;YES, GO TO ADJUST GAIN
005464  104401  001170                  TYPE    ,$QUES          ;TYPE "?"
005470  000763                          BR      1$              ;;
```

```
005472  104401  012455          AJOFF:  TYPE    ,IGND           ;GROUND CHANNEL
005476  104407                          RDLIN                   ;WAIT FOR CR
005500  005726                          TST     (SP)+           ;POP 1 WORD OFF STACK
005502  104401  012415          1$:     TYPE    ,XADJ           ;ADJUST MESSAGE
005506  104401  012514                  TYPE    ,CRWR           ;TYPE "TYPE CR WHEN READY"
005512  012703  000005                  MOV     #5,R3           ;SET UP COUNT
005516  004737  005214          2$:     JSR     PC,OFFSET       ;TEST AND TYPE OFFSET ERROR
005522  004737  005270                  JSR     PC,TOFF         ;TYPE OFFSET
005526  004737  005312                  JSR     PC,TCHK         ;CHECK FOR A CHARACTER AND DELAY
005532  000771                          BR      2$              ;;
005534  000762                          BR      1$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
005536  000005                          RESET
005540  000137  002262                  JMP     BEG2
005544  104401  012543          AJGAIN: TYPE    ,IVOLT          ;INPUT +5.115 VOLTS ON CHANNEL
005550  104401  012514                  TYPE    ,CRWR
005554  104407                          RDLIN                   ;WAIT FOR CR
005556  005726                          TST     (SP)+           ;POP 1 WORD OFF STACK
005560  104401  012607          1$:     TYPE    ,YADJ           ;ADJUST MESSAGE
005564  104401  012431                  TYPE    ,MOLSB          ;TYPE " FOR 0.00 LSB ERROR"
005570  104401  012514                  TYPE    ,CRWR
005574  012703  000005                  MOV     #5,R3           ;SET UP COUNT
005600  012737  007777  001420  2$:     MOV     #7777,EDGE      ;LOOK FOR 7776,7777 EDGE
005606  004537  007144                  JSR     R5,SARSUB
005612  000062                          50.
005614  013737  001414  001356          MOV     DAC,TEMP        ;SAVE DAC
005622  012737  007776  001420          MOV     #7776,EDGE      ;LOOK FOR 7775,7776 EDGE
005630  004537  007144                  JSR     R5,SARSUB
005634  000062                          50.
005636  063737  001414  001356          ADD     DAC,TEMP        ;ADD RESULTS
005644  162737  000400  001356          SUB     #400,TEMP       ;OFFSET RESULT
005652  004737  005270                  JSR     PC,TOFF         ;TYPE GAIN
005656  004737  005312                  JSR     PC,TCHK         ;CHECK FOR CHARACTER AND DELAY
005662  000746                          BR      2$              ;;
005664  000735                          BR      1$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
005666  000005                          RESET
005670  000137  002262                  JMP     BEG2
```

```
                                           .SBTTL         PRINT VALUES ROUTINE
     005674  012737  005674  001374  BEGINP: MOV     #BEGINP,TADDR              ;TEST ADDRESS IN TADDR
     005702  005077  173410                  CLR     @STREG                     ;CLEAR STATUS REGISTER
     005706  104401  013723                  TYPE    ,HEAD5                     ;TYPE OUT HEADING
     005712  005046                          CLR     -(SP)                      ;CLEAR PSW
     005714  012746  005722                  MOV     #1$,-(SP)
     005720  000002                          RTI
     005722  017700  173212          1$:     MOV     @SWR,R0                    ;READ CHANNEL FROM SWITCH REG.
     005726  042700  177700                  BIC     #177700,R0                 ;ISOLATE MUX BITS
     005732  032777  020000  173200          BIT     #BIT13,@SWR                ;IS BIT 13 SET?
     005740  001005                          BNE     2$                ;;YES,SKIP TYPEOUT
     005742  104401  012167                  TYPE    ,CH
     005746  010046                          MOV     R0,-(SP)          ;;SAVE R0 FOR TYPEOUT
(1)                                                                    ;;TYPE CHANNEL
(1)  005750  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)  005752     002                          .BYTE   2                 ;;TYPE 2 DIGIT(S)
(1)  005753     000                          .BYTE   0                 ;;SUPPRESS LEADING ZEROS
     005754  012777  001620  173342  2$:     MOV     #RETURN,@VECTOR            ;ADDRESS AFTER INTRPT.
     005762  000300                          SWAB    R0                         .SWITCH BYTES
     005764  052700  000100                  BIS     #BIT6,R0
     005770  010077  173322                  MOV     R0,@STREG                  ;LOAD THE CHANNEL
     005774  012702  000010                  MOV     #10,R2                     ;TYPEOUT COUNTER
     006000  005277  173312          3$:     INC     @STREG                     ;START CONVERSION
     006004  000001                          WAIT                               ;WAIT FOR INTRPT.
     006006  017700  173310                  MOV     @ADBUFF,R0                 ;READ CONVERTED VALUE
     006012  032777  020000  173120          BIT     #BIT13,@SWR                ;IS BIT 13 SET?
     006020  001403                          BEQ     4$                         ;NOT SET, TYPE OUT LIST
     006022  010077  173114                  MOV     R0,@DISPLAY                        ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
     006026  000735                          BR      1$                         ;REPEAT CONVERSION
     006030  104401  012172          4$:     TYPE    ,SPACE
     006034  010046                          MOV     R0,-(SP)          ;;SAVE R0 FOR TYPEOUT
(1)                                                                    ;;PRINT OCTAL CONVERTED VALUE
(1)  006036  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)  006040     004                          .BYTE   4                 ;;TYPE 4 DIGIT(S)
(1)  006041     001                          .BYTE   1                 ;;TYPE LEADING ZEROS
     006042  012701  010000                  MOV     #10000,R1
     006046  005301                  5$:     DEC     R1
     006050  001376                          BNE     5$
     006052  005302                          DEC     R2                         ;DECREMENT THE COUNTER
     006054  001351                          BNE     3$                         ;NO CARRIAGE RETURN
     006056  104401  001171                  TYPE    ,$CRLF                     ;CARRIAGE RETURN
     006062  000717                          BR      1$                         ;REPEAT CONVERSION
```

```
                                        .SBTTL          LOGIC TEST SECTION
 006064  012737  0C6064  001374  BEGL:   MOV     #BEGL,TADDR              ;TEST ADDRESS
 006072  004737  002464                  JSR     PC,TESTAD                ;NO OF ADDITIONAL AD'S
 006076  004737  002670          1$:     JSR     PC,BEGINL                ;LOGIC TESTS
 006102  004737  006242                  JSR     PC,BUMPAD                ;MORE TO TEST?
 006106  000773                          BR      1$                       ;TEST NEXT A/D
 006110  012737  006076  011704          MOV     #1$,AGTST                ;ADDRESS FOR EOP
 006116  000137  011706                  JMP     $EOP                     ;TYPE END OF PASS

                                        .SBTTL          AUTO TEST
 006122  012737  006122  001374  BEGINA: MOV     #BEGINA,TADDR            ;TEST ADDRESS
 006130  004737  002464                  JSR     PC,TESTAD                ;NO. OF AD'S TO BE TESTED
 006134  004737  002670          1$:     JSR     PC,BEGINL                ;LOGIC TESTS
 006140  104401  013115                  TYPE    ,MEND                    ;TYPE END OF LOGIC TEST
 006144  013746  001316                  MOV     STREG,-(SP)              ;SAVE STREG FOR TYPEOUT
 006150  104403                          TYPOS                            ;TYPE OCTAL NUMBER
 006152     006                          .BYTE   6                        ;TYPE 6 DIGITS
 006153     001                          .BYTE   1                        ;TYPE LEADING ZEROS
 006154  104401  001171                  TYPE    ,$CRLF                   ;TYPE A CR,LF
 006160  004737  004162                  JSR     PC,WRAP
 006164  004737  006242                  JSR     PC,BUMPAD                ;TEST NEXT A/D
 006170  000761                          BR      1$                       ;TEST NEXT AD
 006172  012737  006134  011704          MOV     #1$,AGTST                ;ADDRESS FOR EOP
 006200  000137  011706                  JMP     $EOP                     ;TYPE END OF PASS

                                        .SBTTL          WRAPAROUND TEST
 006204  012737  006204  001374  BEGINW: MOV     #BEGINW,TADDR            ;TEST ADDRESS
 006212  004737  002464                  JSR     PC,TESTAD                ;NO. OF AD'S TO BE TESTED
 006216  004737  004162          1$:     JSR     PC,WRAP                  ;WRAPAROUND TESTS
 006222  004737  006242                  JSR     PC,BUMPAD                ;MORE A/D'S TO BE TESTED?
 006226  000773                          BR      1$                       ;YES-GO TEST NEXT ADV11
 006230  012737  006216  011704          MOV     #1$,AGTST
 006236  000137  011706                  JMP     $EOP                     ;INCREMENTS $PASS
```

```
                                        .SBTTL      DETERMINE IF MORE ADV11'S TO BE TESTED
 006242  005737  001364                 BUMPAD: TST     NBEXT                       ;ADDITIONAL AD'S?
 006246  001434                                 BEQ     FIXADR                      ;NO-INITIALIZE ADDRESSES
 006250  006337  001440                         ASL     TSTBIT              ;MOVE BIT TO NEXT MODULE
 006254  063737  001336  001316                 ADD     VADR,STREG                  ;SET UP NEW ST. REG.
 006262  063737  001336  001320                 ADD     VADR,ADST1                  ;SET UP NEW ADST1
 006270  063737  001336  001322                 ADD     VADR,ADBUFF                 ;SET UP NEW BUFFER ADDRESS
 006276  063737  001340  001324                 ADD     VVCT,VECTOR                 ;SET UP NEW VECTOR
 006304  063737  001340  001330                 ADD     VVCT,VECTR1
 006312  063737  001340  001332                 ADD     VVCT,VECTR2
 006320  063737  001340  001334                 ADD     VVCT,VECTR3
 006326  005077  172776                         CLR     @VECTR1
 006332  005337  001364                         DEC     NBEXT                       ;ONE LESS ADV11
 006336  000473                                 BR      BYPASS
 006340  062716  000002                 FIXADR: ADD     #2,(SP)
 006344  012737  000006  000004         FIXONE: MOV     #6,@#ERRVEC                 ;SET UP ERRVEC
 006352  012737  007336  000010                 MOV     #DELAY4,@#RESVEC            ;SETUP RESERVED INST. VECTOR
 006360  012737  000001  001440                 MOV     #1,TSTBIT           ;INITIALIZE MODULE ERROR TEST BIT
 006366  013737  001250  001316                 MOV     $BASE,STREG                 ;RELOAD INITIAL ADDRESSES
 006374  013737  001250  001320                 MOV     $BASE,ADST1
 006402  013737  001250  001322                 MOV     $BASE,ADBUFF
 006410  005237  001320                         INC     ADST1
 006414  062737  000002  001322                 ADD     #2,ADBUFF
 006422  013737  001244  001324                 MOV     $VECT1,VECTOR
 006430  042737  170000  001324                 BIC     #170000,VECTOR
 006436  113737  001245  001326                 MOVB    $VECT1+1,BASEBR
 006444  105037  001327                         CLRB    BASEBR+1                    ;CLEAR HIGH BYTE
 006450  013737  001324  001330                 MOV     VECTOR,VECTR1
 006456  062737  000002  001330                 ADD     #2,VECTR1
 006464  013737  001324  001332                 MOV     VECTOR,VECTR2
 006472  062737  000004  001332                 ADD     #4,VECTR2
 006500  013737  001324  001334                 MOV     VECTOR,VECTR3
 006506  062737  000006  001334                 ADD     #6,VECTR3
 006514  005077  172610                         CLR     @VECTR1
 006520  013737  001366  001364                 MOV     NMBEXT,NBEXT                ;RESET COUNTER
                                        ;;LOAD .+2 AND HALT TRAP CATCH;;
 006526  012700  000216                 BYPASS: MOV     #216,R0                     ;FILL .+2
 006532  012701  000214                         MOV     #214,R1                     ;LOAD HALT
 006536  020137  001344                 1$:     CMP     R1,KBVECT
 006542  001410                                 BEQ     2$
 006544  010021                                 MOV     R0,(R1)+
 006546  005021                                 CLR     (R1)+
 006550  010100                                 MOV     R1,R0
 006552  005720                                 TST     (R0)+
 006554  020027  001002                         CMP     R0,#1002
 006560  001366                                 BNE     1$
 006562  000207                                 RTS     PC                          ;TEST NEXT A/D
 006564  022021                         2$:     CMP     (R0)+,(R1)+
 006566  022021                                 CMP     (R0)+,(R1)+
 006570  000762                                 BR      1$
```

```
834  006572  104413                    TYPSET: TYPDC
835  006574  104401  012177                     TYPE    ,LSB
836  006600  013746  001362                     MOV     CH2,-(SP)         ;;SAVE CH2 FOR TYPEOUT
(1)                                                                       ;;TYPE CH
(1)  006604  104403                             TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)  006606     002                             .BYTE   2                 ;;TYPE 2 DIGIT(S)
(1)  006607     000                             .BYTE   0                 ;;SUPPRESS LEADING ZEROS
837  006610  104401  014035                     TYPE    ,MAT              ;TYPE ASCIZ STRING
838  006614  004737  007100                     JSR     PC,TYPEDG
839  006620  104401  012212                     TYPE    ,SETCH
840  006624  013746  001360                     MOV     CH1,-(SP)         ;;SAVE CH1 FOR TYPEOUT
(1)                                                                       ;;TYPE CH
(1)  006630  104403                             TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)  006632     002                             .BYTE   2                 ;;TYPE 2 DIGIT(S)
(1)  006633     000                             .BYTE   0                 ;;SUPPRESS LEADING ZEROS
841  006634  104401  012234                     TYPE    ,ATMSG
842  006640  013737  001360  006666             MOV     CH1,1$
843  006646  163737  001342  006666             SUB     BASECH,1$
844  006654  012777  000200  172440             MOV     #200,@ADBUFF
845  006662  004537  011074                     JSR     R5,CONVRT
846  006666  000000                     1$:     0
847  006670  013746  001356                     MOV     TEMP,-(SP)        ;;SAVE TEMP FOR TYPEOUT
(1)                                                                       ;;TYPE VALUE
(1)  006674  104403                             TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)  006676     004                             .BYTE   4                 ;;TYPE 4 DIGIT(S)
(1)  006677     001                             .BYTE   1                 ;;TYPE LEADING ZEROS
848  006700  020437  011666                     CMP     R4,VSET
849  006704  003003                             BGT     ERR
850  006706  104401  012303                     TYPE    ,OKMSG
851  006712  000207                             RTS     PC
852  006714  104401  012625             ERR:    TYPE    ,ERMSG
853  006720  000207                             RTS     PC
854
855                                     ;;SUBROUTINE FOR SETTLING TESTS;;
856  006722  013737  001362  001370     SET1A:  MOV     CH2,DUMMY                   ;LOAD DUMMY
857  006730  004537  007144                     JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 50%
858  006734  000062                             50.
859  006736  063702  001414                     ADD     DAC,R2            ;ADD RESULT TO R2
860  006742  013737  001360  001370             MOV     CH1,DUMMY                   ;CHANGE DUMMY VALUE
861  006750  004537  007144                     JSR     R5,SARSUB                   ;DO SAR ROUTINE AT 50%
862  006754  000062                             50.
863  006756  163702  001414                     SUB     DAC,R2            ;SUBTRACT RESULT FROM R2
864  006762  000207                             RTS     PC                ;RETURN
```

```
                                                 ;SUBROUTINE TO GET EDGE VALUE
                                                 ;CALL=JSR   PC,GETEDG
                                                 ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                                 ;RESULT IN EDGE, USES R0
      006764  012777  000200  172330  GETEDG: MOV     #200,@ADBUFF      ;LOAD VERNIER DAC
      006772  113700  001372                  MOVB    CHANL,R0                   ;GET CHANNEL
      006776  000300                          SWAB    R0                ;SET UP A.D STATUS REG.
      007000  052700  000100                  BIS     #100,R0           ;ENABLE INTRPT.
      007004  010077  172306                  MOV     R0,@STREG
      007010  012700  000100                  MOV     #100,R0           ;DAC SETTLING DELAY
      007014  005300                  1$:     DEC     R0
      007016  001376                          BNE     1$
      007020  005037  001420                  CLR     EDGE
      007024  012700  000010                  MOV     #10,R0
      007030  012777  001620  172266          MOV     #RETURN,@VECTOR            ;RETURN ADDRESS
      007036  005277  172254          CONV:   INC     @.TREG                     ;START CONVERSION
      007042  000001                          WAIT                               ;WAIT FOR INTERRUPT
      007044  067737  172252  001420          ADD     @ADBUFF,EDGE
      007052  005300                          DEC     R0
      007054  001370                          BNE     CONV
      007056  006237  001420                  ASR     EDGE
      007062  006237  001420                  ASR     EDGE
      007066  006237  001420                  ASR     EDGE
      007072  005537  001420                  ADC     EDGE
      007076  000207                          RTS     PC

                                      ;;SUBROUTINE TO TYPE EDGE VALUES;;
      007100  013703  001420          TYPEDG: MOV     EDGE,R3
      007104  010346                          MOV     R3,-(SP)          ;;SAVE R3 FOR TYPEOUT
 (1)                                                                    ;;TYPE OCTAL VALUE OF EDGE
 (1)  007106  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
 (1)  007110     004                          .BYTE   4                 ;;TYPE 4 DIGIT(S)
 (1)  007111     001                          .BYTE   1                 ;;TYPE LEADING ZEROS
      007112  023727  007142  000001          CMP     EDGFLG,#1
      007120  001407                          BEQ     RET
      007122  062703  000007                  ADD     #7,R3
      007126  104401  012103                  TYPE    ,MINUS            ;TYPE ASCIZ STRING
      007132  010346                          MOV     R3,-(SP)          ;;SAVE R3 FOR TYPEOUT
 (1)                                                                    ;;TYPE EDGE VALUE
 (1)  007134  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
 (1)  007136     004                          .BYTE   4                 ;;TYPE 4 DIGIT(S)
 (1)  007137     001                          .BYTE   1                 ;;TYPE LEADING ZEROS
      007140  000207                  RET:    RTS     PC
      007142  000000                  EDGFLG: 0
```

```
                                        ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
                                        ;CALL=JSR   R5,SARSUB
                                          :  XXX;XXX=PERCENT
                                        ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
007144  012537  001432                  SARSUB: MOV     (R5)+,PERCNT            ;GET PERCENT
007150  006337  001432                          ASL     PERCNT
007154  006337  001432                          ASL     PERCNT
007160  006337  001432                          ASL     PERCNT                  ;RESCALE PERCENT FOR 1600.
007164  006337  001432                          ASL     PERCNT                  ;POINTS PER BURST
007170  012737  000200  001422          SAR1:   MOV     #200,BITPNT             ;INITIALIZE BIT POINTER AT MSB
007176  005037  001414                          CLR     DAC                     ;INITIALIZE DAC VALUE
007202  005000                          TRY:    CLR     R0
007204  063737  001422  001414                  ADD     BITPNT,DAC              ;TRY BIT
007212  013777  001414  172102                  MOV     DAC,@ADBUFF
007220  012701  003100                          MOV     #1600.,R1               ;SET UP FOR 1600. CONVERSIONS
007224  113777  001370  172066          NXTCVT: MOVB    DUMMY,@ADST1            ;PRESET MUX TO DUMMY CHANNEL
007232  012777  001620  172064                  MOV     #RETURN,@VECTOR         ;RETURN ADDRESS
007240  052777  000101  172050                  BIS     #101,@STREG             ;CONVERSION ON DUMMY CHANNEL
007246  000001                                  WAIT                            ;WAIT FOR INTERRUPT
007250  017704  172046                          MOV     @ADBUFF,R4              ;DUMMY READ
007254  013704  001372                          MOV     CHANL,R4
007260  000304                                  SWAB    R4
007262  052704  000101                          BIS     #101,R4                 ;INTERRUPT ENABLE START
007266  010477  172024                          MOV     R4,@STREG               ;JUMP TO CHANNEL + START CONVERT
007272  000001                                  WAIT                            ;WAIT FOR INTERRUPT
007274  027737  172022  001420                  CMP     @ADBUFF,EDGE
007302  002001                                  BGE     2$
007304  005200                                  INC     R0                      ;COUNT RESULTS .LT. EDGE
007306  005301                          2$:     DEC     R1
007310  001345                                  BNE     NXTCVT
007312  020037  001432                          CMP     R0,PERCNT
007316  003003                                  BGT     SHIFT
007320  163737  001422  001414                  SUB     BITPNT,DAC              ;TAKE THE BIT OUT
007326  006237  001422                  SHIFT:  ASR     BITPNT
007332  001323                                  BNE     TRY
007334  000205                                  RTS     R5

                                        .*ROUTINE FOR PROCESSERS THAT CAN'T DO A SOB INSTRUCTION

007336  005300                          DELAY4: DEC     R0              ;DECREMENT R0, IS IT ZERO?
007340  001376                                  BNE     DELAY4          ;NO
007342  000002                                  RTI                     ;RETURN
```

```
                                         ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
007344  104401  013240                   DIFLIN: TYPE    ,MSG20
007350  013702  001376                           MOV     RNA,R2              ;SET UP RANDOM NUMBER GENERATOR
007354  013704  001400                           MOV     RNB,R4
007360  013705  001402                           MOV     RNC,R5
007364  012700  020034                           MOV     #BUFFER,R0
007370  012701  010000                           MOV     #4096.,R1           ;4096 WORDS FOR HISTOGRAM
007374  005020                           CLEAR1: CLR     (R0)+               ;CLEAR BUFFER AREA
007376  005301                                   DEC     R1
007400  001375                                   BNE     CLEAR1
007402  012700  017214                           MOV     #DIST,R0            ;DISTRIBUTION BUFFER POINTER
007406  012701  000310                           MOV     #200.,R1            ;200. WORDS FOR DISTRIBUTION
007412  005003                                   CLR     R3
007414  005037  001434                           CLR     OUT
007420  005037  001346                           CLR     WIDE
007424  005037  001350                           CLR     NARROW
007430  005037  001352                           CLR     FIRST
007434  005037  001354                           CLR     SKIPST
007440  005020                           CLEAR2: CLR     (R0)+               ;CLEAR DISTRIBUTION BUFFER AREA
007442  005301                                   DEC     R1
007444  001375                                   BNE     CLEAR2
007446  012700  000003                   CHANNL: MOV     #3,R0               ;CHANNEL 3
007452  063700  001342                           ADD     BASECH,R0
007456  000300                                   SWAB    R0                                      ;LOAD MUX BITS
007460  052700  000100                           BIS     #100,R0
007464  010077  171626                           MOV     R0,@STREG
007470  012737  001440  001416                   MOV     #800.,DELAY         ;NOMINAL STATE WIDTH - 1 LSB
007476  012777  001630  171620                   MOV     #RET1,@VECTOR
007504  012701  007776                   AGAIN:  MOV     #4094. R1
007510  060402                           NEXT:   ADD     R4,R2
007512  060502                                   ADD     R5,R2
007514  005502                                   ADC     R2
007516  060204                                   ADD     R2,R4
007520  060504                                   ADD     R5,R4
007522  005504                                   ADC     R4
007524  060205                                   ADD     R2,R5
007526  060405                                   ADD     R4,R5
007530  005505                                   ADC     R5
007532  010500                                   MOV     R5,R0               ;COPY INTO DELAY
007534  042700  177770                           BIC     #177770,R0          ;MASK IT TO 4 BITS ONLY
007540  001401                                   BEQ     CONVR
007542  077001                           DELAY3: SOB     R0,DELAY3           ;STALL TIME
007544  005277  171546                   CONVR:  INC     @STREG              ;START CONVERSION
007550  000001                                   WAIT
007552  000240                                   NOP
007554  017700  171542                           MOV     @ADBUFF,R0          ;GET CONVERTED VALUE
007560  001416                                   BEQ     DELAY1              ;IGNORE IF =0
007562  020027  007777                           CMP     R0,#7777            ;IGNORE IF =7777
007566  001416                                   BEQ     DELAY2
007570  006300                                   ASL     R0
007572  005260  020034                           INC     BUFFER(R0)          ;MAKE HISTOGRAM
007576  100016                                   BPL     OKAY
007600  012760  077777  020034                   MOV     #077777,BUFFER(R0)          ;PREVENT OVERFLOW
007606  000412                                   BR      OKAY
```

```
      007610  005037  001356          NOTOK:  CLR     TEMP
      007614  000407                          BR      OKAY
      007616  020027  007777          DELAY1: CMP     R0,#7777          ;EQUALIZE LOOP TIME
      007622  001400                          BEQ     DELAY2            ;WITH DUMMY INSTR.
      007624  005201                  DELAY2: INC     R1
      007626  005263  001356                  INC     TEMP(R3)
      007632  100766                          BMI     NOTOK
      007634  005301                  OKAY:   DEC     R1
      007636  001324                          BNE     NEXT
      007640  005337  001416          APOUND: DEC     DELAY
      007644  001317                          BNE     AGAIN
      007646  012700  007776                  MOV     #4094.,R0
      007652  012701  020036                  MOV     #BUFFER+2,R1
      007656  012102                  READ:   MOV     (R1)+,R2          ;GET STATE WIDTH
      007660  006202                          ASR     R2                ;1 LSB = 800.
      007662  006202                          ASR     R2
      007664  006202                          ASR     R2
      007666  005502                          ADC     R2                ;1 LSB = 100.
      007670  020227  000310                  CMP     R2,#200.          ;OUT OF RANGE?
      007674  002403                          BLT     INRNGE
      007676  005237  001434                  INC     OUT               ;YES - INCREMENT COUNTER
      007702  000423                          BR      TYPBAD
      007704  006302                  INRNGE: ASL     R2
      007706  005262  017214                  INC     DIST(R2)          ;MAKE STATE WIDTH DISTRIBUTION
      007712  006202                          ASR     R2
      007714  020227  000062                  CMP     R2,#50.           ;IS IT 1/2 LSB?
      007720  002007                          BGE     NOTNAR
      007722  005237  001350                  INC     NARROW
      007726  005702                          TST     R2                ;IS IT A SKIPPED STATE?
      007730  001002                          BNE     31$
      007732  005237  001354                  INC     SKIPST
      007736  000405                  31$:    BR      TYPBAD
      007740  020227  000226          NOTNAR: CMP     R2,#150.          ;IS IT 1.5 LSB?
      007744  003425                          BLE     LAST
      007746  005237  001346                  INC     WIDE
      007752  005737  001352          TYPBAD: TST     FIRST
      007756  001004                          BNE     60$
      007760  005237  001352                  INC     FIRST
      007764  104401  012147                  TYPE    ,STATE
      007770  010103                  60$:    MOV     R1,R3
      007772  162703  020036                  SUB     #BUFFER+2,R3
      007776  006203                          ASR     R3
      010000  010346                          MOV     R3,-(SP)          ;;SAVE R3 FOR TYPFOUT
 (1)                                                                    ;;TYPE STATE
 (1)  010002  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
 (1)  010004     004                          .BYTE   4                 ;;TYPE 4 DIGIT(S)
 (1)  010005     001                          .BYTE   1                 ;;TYPE LEADING ZEROS
      010006  104401  012143                  TYPE    ,DASH
      010012  104413                          TYPDC
      010014  104401  012134                  TYPE    ,LSBMSG
```

```
010020  005300                  LAST:   DEC     R0
010022  001315                          BNE     READ
010024  112737  000177  014620          MOVB    #177,DECPNT
010032  013702  001354                  MOV     SKIPST,R2                       ;GET NO. OF SKIPPED STATES
010036  104413                          TYPDC                                   ;TYPE IT
010040  104401  012642                  TYPE    ,SKPMSG         ;TYPE MESSAGE
010044  005737  001354                  TST     SKIPST
010050  001403                          BEQ     1$
010052  104401  012625                  TYPE    ,ERMSG          ;TYPE "ERROR"
010056  000402                          BR      NAR
010060  104401  012303          1$:     TYPE    ,OKMSG    .     ;TYPE #OK#
010064  013702  001350          NAR:    MOV     NARROW,R2                       ;GET NO. OF NARROW STATES
010070  104413                          TYPDC                                   ;TYPE IT
010072  104401  012664                  TYPE    ,NARMSG         ;TYPE MESSAGE
010076  013702  001346                  MOV     WIDE,R2
010102  063702  001434                  ADD     OUT,R2
010106  104413                          TYPDC                                   ;TYPE NO. OF WIDE STATES
010110  104401  012723                  TYPE    ,WIDMSG         ;TYPE MESSAGE
010114  013702  001434                  MOV     OUT,R2
010120  104413                          TYPDC                                   ;TYPE NO. OF STATES OUTSIDE 2 LSB
010122  104401  012762                  TYPE    ,OUTMSG         ;TYPE MESSAGE
010126  005737  001434                  TST     OUT
010132  001403                          BEQ     11$
010134  104401  012625                  TYPE    ,ERMSG          ;TYPE "ERROR"
010140  000402                          BR      HALF
010142  104401  012303          11$:    TYPE    ,OKMSG          ;TYPE "OK"
010146  013702  001350          HALF:   MOV     NARROW,R2
010152  063702  001346                  ADD     WIDE,R2
010156  063702  001434                  ADD     OUT,R2
010162  010200                          MOV     R2,R0
010164  104413                          TYPDC                                   ;TYPE NO. OF STATES OUTSIDE LIMITS
010166  112737  000056  014620          MOVB    #56,DECPNT
010174  104401  013015                  TYPE    ,HAFMSG
010200  020027  000051                  CMP     R0,#41.         ;COMPARE IT TO NOMINAL
010204  003403                          BLE     21$
010206  104401  012625                  TYPE    ,ERMSG          ;TYPE "ERROR"
010212  000402                          BR      SWDIST
010214  104401  012303          21$:    TYPE    ,OKMSG          ;TYPE "OK"
010220  005737  001410          SWDIST: TST     FLAG            ;VT55?
010224  001426                          BEQ     RELACC
010226  004737  010704                  JSR     PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
010232  104401  013272                  TYPE    ,MSG16
010236  104401  014064                  TYPE    ,BUFF1          ;TYPE BUFF1-PRINT GRID
010242  012700  017214                  MOV     #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
010246  012701  000310                  MOV     #200.,R1        ;GO 200. TIMES UP TO 2 LSB
010252  012002                  NXTY1:  MOV     (R0)+,R2
010254  004737  011400                  JSR     PC,LOADY
010260  005002                          CLR     R2
010262  004737  011400                  JSR     PC,LOADY
010266  005301                          DEC     R1
010270  001370                          BNE     NXTY1
010272  104401  014005                  TYPE    ,C2             ;TYPE ASCIZ STRING
010276  004737  010704                  JSR     PC,DELCLR
```

```
                                            ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

    010302  005001                  RELACC: CLR     R1              ;RUNNING ERROR = 0
    010304  005003                          CLR     R3              ;MAXIMUM ERROR = 0
    010306  104401  013655                  TYPE    ,MSG21
    010312  012700  020036                  MOV     #BUFFER+2,R0
    010316  011002                  NXTSTA: MOV     (R0),R2         ;STATE WIDTH = R2
    010320  162702  001440                  SUB     #800.,R2        ;STATE WIDTH ERROR IN R2
    010324  060201                          ADD     R2,R1           ;UPDATE RUNNING ERROR
    010326  010120                          MOV     R1,(R0)+        ;SAVE IN BUFFER
    010330  010104                          MOV     R1,R4           ;SAVE IN R4 ALSO
    010332  100001                          BPL     PLUS            ;IS IT POSITIVE?
    010334  005404                          NEG     R4              ;NO - MAKE IT POSITIVE
    010336  020403                  PLUS:   CMP     R4,R3           ;CHECK AGAINST PREVIOUS MAX. ERROR
    010340  003405                          BLE     NOTNEW          ;NOT A NEW MAXIMUM
    010342  010403                          MOV     R4,R3           ;UPDATE MAXIMUM IN R3
    010344  010005                          MOV     R0,R5
    010346  162705  020036                  SUB     #BUFFER+2,R5
    010352  006205                          ASR     R5              ;R5=EDGE VALUE AT MAX. RELACC
    010354  020027  040032          NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
    010360  001356                          BNE     NXTSTA          ;NO - REPEAT
    010362  006203                          ASR     R3              ;RESCALE FROM 1 LSB = 800. SCALING
    010364  006203                          ASR     R3              ;TO 1 LSB = 100. SCALING
    010366  006203                          ASR     R3
    010370  005503                          ADC     R3
    010372  010302                          MOV     R3,R2
    010374  104413                          TYPDC
    010376  104401  013702                  TYPE    ,LINEA
    010402  010546                          MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
(1)                                                                 ;;TYPE VALUE
(1) 010404  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
(1) 010406     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
(1) 010407     001                          .BYTE   1               ;;TYPE LEADING ZEROS
    010410  104401  012301                  TYPE    ,SLASH                  ;PRINT '/'
    010414  005205                          INC     R5
    010416  010546                          MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
(1)                                                                 ;;TYPE VALUE
(1) 010420  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
(1) 010422     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
(1) 010423     001                          .BYTE   1               ;;TYPE LEADING ZEROS
    010424  020337  011670                  CMP     R3,VLIN
    010430  003403                          BLE     41$
    010432  104401  012625                  TYPE    ,ERMSG
    010436  000402                          BR      42$
    010440  104401  012303          41$:    TYPE    ,OKMSG
    010444  005737  001410          42$:    TST     FLAG            ;VT55?
    010450  001503                          BEQ     L02
    010452  012700  020034                  MOV     #BUFFER,R0
    010456  012701  010000                  MOV     #4096.,R1
```

```
010462  011002                  GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
010464  006202                          ASR     R2              ;RESCALE IT TO 1 LSB = 100.
010466  006202                          ASR     R2
010470  006202                          ASR     R2
010472  005502                          ADC     R2
010474  062702  000166                  ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
010500  010220                          MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
010502  005301                          DEC     R1
010504  001366                          BNE     GETDAT
010506  012700  020034                  MOV     #BUFFER,R0
010512  012704  020034                  MOV     #BUFFER,R4
010516  012705  020036                  MOV     #BUFFER+2,R5
010522  012701  001000                  MOV     #512.,R1
010526  012702  000007          NXT8:   MOV     #7.,R2
010532  012003                          MOV     (R0)+,R3
010534  010337  001424                  MOV     R3,MIN          ;MINIMUM
010540  010337  001430                  MOV     R3,MAX          ;MAXIMUM
010544  012003                  NXTCMP: MOV     (R0)+,R3
010546  020337  001424                  CMP     R3,MIN
010552  002002                          BGE     MAXTST
010554  010337  001424                  MOV     R3,MIN          ;NEW MINIMUM
010560  020337  001430          MAXTST: CMP     R3,MAX
010564  003402                          BLE     TST8
010566  010337  001430                  MOV     R3,MAX          ;NEW MAXIMUM
010572  005302                  TST8:   DEC     R2
010574  001363                          BNE     NXTCMP
010576  013724  001424                  MOV     MIN,(R4)+
010602  013725  001430                  MOV     MAX,(R5)+
010606  022425                          CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
010610  005301                          DEC     R1
010612  001345                          BNE     NXT8
010614  104401  013200                  TYPE    ,MSG18
010620  104401  014112                  TYPE    ,BUFF2          ;TYPE BUFF2
010624  012700  020034                  MOV     #BUFFER,R0
010630  004737  010662                  JSR     PC,LOAD
010634  104401  014012                  TYPE    ,C3             ;TYPE ASCIZ STRING
010640  012700  020036                  MOV     #BUFFER+2,R0
010644  004737  010662                  JSR     PC,LOAD
010650  104401  014005                  TYPE    ,C2             ;TYPE ASCIZ STRING
010654  004737  010704                  JSR     PC,DELCLR
010660  000207                  LO2:    RTS     PC
010662  012701  001000          LOAD:   MOV     #512.,R1
010666  012002                  LOAD0:  MOV     (R0)+,R2
010670  005720                          TST     (R0)+
010672  004737  011400                  JSR     PC,LOADY
010676  005301                          DEC     R1
010700  001372                          BNE     LOAD0
010702  000207                          RTS     PC
```

```
     010704  032777  010000  170226  DELCLR: BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
     010712  001402                          BEQ     1$              ;;DON'T HALT FOR DISPLAY
     010714  000000                          HALT
     010716  000407                          BR      3$              ;;
     010720  005000                  1$:     CLR     R0
     010722  012701  000020                  MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
     010726  005300                  2$:     DEC     R0
     010730  001376                          BNE     2$
     010732  005301                          DEC     R1
     010734  001374                          BNE     2$
     010736  104401  014132          3$:     TYPE    ,VTINIT
     010742  000207                          RTS     PC
                                     ;;TYPE RMS AND PEAK VALUES;;
     010744  104401  012241          TYPRP:  TYPE    ,NOI
     010750  005737  001404                  TST     RMS
     010754  100002                          BPL     POSRMS
     010756  005037  001404                  CLR     RMS                     ;RMS<0,SET RMS=0
     010762  005737  001406          POSRMS: TST     PEAK
     010766  100002                          BPL     POSPEA
     010770  005037  001406                  CLR     PEAK                    ;PEAK<0,SET PEAK-0
     010774  013702  001404          POSPEA: MOV     RMS,R2
     011000  104413                          TYPDC
     011002  104401  013064                  TYPE    ,MESR           ;TYPE " LSB RMS, "
     011006  013702  001406                  MOV     PEAK,R2
     011012  104413                          TYPDC
     011014  104401  013077                  TYPE    ,MESP           ;TYPE " LSB PEAK AT "
     011020  004737  007100                  JSR     PC,TYPEDG
     011024  104401  012251                  TYPE    ,CHAN           ;TYPE " ON CHANNEL "
     011030  013746  001372                  MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
 (1)                                                                 ;;TYPE CHANL
 (1) 011034  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1) 011036     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1) 011037     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
     011040  023737  001404  011662          CMP     RMS,VNR         ;WITHIN LIMITS?
     011046  003007                          BGT     ER
     011050  023737  001406  011664          CMP     PEAK,VNP                ;WITHIN LIMITS?
     011056  003003                          BGT     ER
     011060  104401  012303                  TYPE    ,OKMSG
     011064  000207                          RTS     PC
     011066  104401  012625          ER:     TYPE    ,ERMSG
     011072  000207                          RTS     PC
```

```
                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
011074  012500                          CONVRT: MOV     (R5)+,R0                        ;GET CHANNEL VALUE
011076  063700  001342                          ADD     BASECH,R0
011102  010037  001372                          MOV     R0,CHANL
011106  000300                                  SWAB    R0
011110  005037  001356                          CLR     TEMP
011114  010077  170176                          MOV     R0,@STREG                       ;LOAD CHANNEL INTO MIX BITS
011120  012700  010000                          MOV     #10000,R0
011124  005300                          2$:     DEC     R0
011126  001376                                  BNE     2$
011130  012777  001620  170166                  MOV     #RETURN,@VECTOR                 ;LOAD VECTOR
011136  012700  000010                          MOV     #10,R0                  ;SET UP COUNTER
011142  152777  000101  170146          1$:     BISB    #101,@STREG                     ;SET INTRPT. EN., START CONV.
011150  000001                                  WAIT                            ;WAIT FOR CONVERSION
011152  067737  170144  001356                  ADD     @ADBUFF,TEMP                    ;READ BUFFER
011160  005300                                  DEC     R0
011162  001367                                  BNE     1$                      ;DO 8 TIMES
011164  006237  001356                          ASR     TEMP                    ;AVERAGE VALUE
011170  006237  001356                          ASR     TEMP
011174  006237  001356                          ASR     TEMP
011200  005537  001356                          ADC     TEMP
011204  000205                                  RTS     R5                      ;RETURN

                                        ;COMPARE $GDDAT AND $BDDAT;;
011206  012537  001124                  COMPAR: MOV     (R5)+,$GDDAT                    ;GET GOOD DATA
011212  013537  001412                          MOV     @(R5)+,SPREAD                   ;GET SPREAD
011216  013737  001356  001126                  MOV     TEMP,$BDDAT                     ;GET BAD(ACTUAL) DATA
011224  013701  001126                          MOV     $BDDAT,R1
011230  013700  001124                          MOV     $GDDAT,R0
011234  160100                                  SUB     R1,R0                   ;GET DIFFERENCE
011236  100001                                  BPL     7$
011240  005400                                  NEG     R0
011242  020037  001412                  7$:     CMP     R0,SPREAD                       ;COMPARE IT TO SPREAD
011246  003001                                  BGT     10$                     ;GO TO ERROR PRINTOUT
011250  005725                                  TST     (R5)+                   ;BUMP RETURN POINTER AROUND ERROR CALL
011252  000205                          10$:    RTS     R5
```

```
  1268                                          ;;SUBROUTINE TO TYPE INTRPT. TST MSG.;;
  1269  011254  005737  001202                  DUMW:   TST     $PASS
  1270  011260  001021                                  BNE     20$
  1271  011262  012737  011324  001110                  MOV     #20$,$LPERR
  1272  011270  012737  011324  001106                  MOV     #20$,$LPADR
  1273  011276  104401  014042                          TYPE    ,METST          ;TYPE ASCIZ STRING
  1274  011302  010046                                  MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
   (1)                                                                          ;;TYPE TEST NO.
   (1)  011304  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
   (1)  011306     002                                  .BYTE   2               ;;TYPE 2 DIGIT(S)
   (1)  011307     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
  1275  011310  104401  013141                          TYPE    ,ONAD
  1276  011314  013746  001316                          MOV     STREG,-(SP)     ;;SAVE STREG FOR TYPEOUT
   (1)                                                                          ;;TYPE BUS ADDRESS
   (1)  011320  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
   (1)  011322     006                                  .BYTE   6               ;;TYPE 6 DIGITS
   (1)  011323     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
  1277  011324  000207                          20$:    RTS     PC
  1278
  1279  011326  005737  001202                  DUMC:   TST     $PASS
  1280  011332  001010                                  BNE     30$
  1281  011334  012737  011354  001110                  MOV     #30$,$LPERR
  1282  011342  012737  011354  001106                  MOV     #30$,$LPADR
  1283  011350  104401  012266                          TYPE    ,DONE
  1284  011354  000207                          30$:    PTS     PC
```

```
                                           ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011356  000005                             RST:     RESET
011360  052777  000100  167556                      BIS     #100,@$TKS
011366  005046                                      CLR     -(SP)              ;CLEAR PSW
011370  012746  011376                              MOV     #1$,-(SP)
011374  000002                                      RTI
011376  000207                             1$:      RTS     PC

                                           ;SUBROUTINE LOADY;
011400  005702                             LOADY:   TST     R2                 ;ROUTINE TO LOAD VLAUE INTO R2
011402  100001                                      BPL     PLUSR2             ;AS A V155 Y-VALUE ,
011404  005002                                      CLR     R2
011406  020227  000353                     PLUSR2:  CMP     R2,#235.
011412  002402                                      BLT     LESS
011414  012702  000353                              MOV     #235.,R2
011420  010203                             LESS:    MOV     R2,R3
011422  042702  177740                              BIC     #177740,R2
011426  052702  000040                              BIS     #40,R2
011432  105777  167512                     B10:     TSTB    @$TPS              ;PRINT CHARACTER
011436  100375                                      BPL     B10
011440  110277  167506                              MOVB    R2,@$TPB
011444  006203                                      ASR     R3
011446  006203                                      ASR     R3
011450  006203                                      ASR     R3
011452  006203                                      ASR     R3
011454  006203                                      ASR     R3
011456  042703  177770                              BIC     #177770,R3
011462  052703  000040                              BIS     #40,R3
011466  105777  167456                     B11:     TSTB    @$TPS              ;PRINT CHARACTER
011472  100375                                      BPL     B11
011474  110377  167452                              MOVB    R3,@$TPB
011500  000207                                      RTS     PC
```

```
                                    ;;SUBROUTINE TO TYPE DECIMAL VALUE;.
                                    ;;IN R2 AS X.XX;;
011502  005702                      DECTYP: TST     R2              ;TEST VALUE TO BE TYPED
011504  100003                              BPL     POS
011506  104401  012103                      TYPE    ,MINUS                  ;TYPE MINUS SIGN
011512  005402                              NEG     R2
011514  020227  001747              POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
011520  003402                              BLE     OKAYD
011522  012702  001747                      MOV     #999.,R2
011526  105037  014622              OKAYD:  CLRB    ONES            ;CLEAR ONES
011532  105037  014621                      CLRB    TENS            ;CLEAR TENS
011536  105037  014617                      CLRB    HUNS            ;CLEAR HUNS
011542  005702                      TESTR2: TST     R2                      ;CONVERT VALUE TO A DECIMAL VALUE
011544  001424                              BEQ     TYPOUT
011546  005302                              DEC     R2
011550  105237  014622                      INCB    ONES
011554  123727  014622  000012              CMPB    ONES,#10.
011562  001367                              BNE     TESTR2
011564  105037  014622                      CLRB    ONES
011570  105237  014621                      INCB    TENS
011574  123727  014621  000012              CMPB    TENS,#10.
011602  001357                              BNE     TESTR2
011604  105037  014621                      CLRB    TENS
011610  105237  014617                      INCB    HUNS
011614  000752                              BR      TESTR2
011616  152737  000060  014617      TYPOUT: BISB    #60,HUNS                ;PREPARE FOR TYPOUT
011624  152737  000060  014621              BISB    #60,TENS
011632  152737  000060  014622              BISB    #60,ONES
011640  104401  014617                      TYPE    ,HUNS           ;TYPE VALUE
011644  000002                              RTI
011646  000000                      V0:     0                       ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
011650  000002                      V2:     2
011652  000004                      V4:     4
011654  000012                      V12:    12
011656  000062                      V50D:   50.
011660  000326                      V326:   326

011662  000050                      VNR:    40.             ;.4 LSB,NORMAL LIMITS FOR SYSTEM
011664  000310                      VNP:    200.            ;2 LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
011666  000144                      VSET:   100.            ;1 LSB
011670  000175                      VLIN:   125.            ;1.25 LSB
011672  100000                              BIT15

011674  052777  000100  167242      AGATST: BIS     #100,@$TKS
011702  000137                              JMP     @(PC)+
011704  001644                      AGTST:  BEGIN
```

```
                                        .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                                   ;;************************************************************
  (1)                                   ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                   ;*IF THERES A MONITOR GO TO IT
  (1)                                   ;*IF THERE ISN'T JUMP TO AGATST
  (1)
  (1)  011706                           $EOP:
  (2)  011706  000240                           NOP
  (1)  011710  005037  001102                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  011714  005037  001160                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  011720  005237  001202                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  011724  042737  100000  001202           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  011732  005327                           DEC     (PC)+           ;;LOOP?
  (1)  011734  000001                   $EOPCT: .WORD   1
  (1)  011736  003035                           BGT     $DOAGN          ;;YES
  (1)  011740  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  011742  000001                   $ENDCT: .WORD   1
  (1)  011744  011734                           $EOPCT
  (3)  011746  104401  011754                   TYPE    ,65$            ;;TYPE ASCIZ STRING
  (3)  011752  000414                           BR      64$             ;;GET OVER THE ASCIZ
  (3)                                   ;;65$:  .ASCIZ  <15><12>/ENDPASS  GOOD UNITS  /
  (3)  012004                           64$:
  (3)  012004  013746  001436                   MOV     GUNITS,-(SP)    ;;SAVE GUNITS FOR TYPEOUT
  (3)  012010  104405                           TYPBN                   ;;GO TYPE--BINARY ASCII
  (1)  012012  013700  000042           $GET42: MOV     @#42,R0         ;;GET 'ONITOR ADDRESS
  (1)  012016  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  012020  000005                           RESET                   ;;CLEAR THE WORLD
  (1)  012022  004710                   $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  012024  000240                           NOP                     ;;SAVE ROOM
  (1)  012026  000240                           NOP                     ;;FOR
  (1)  012030  000240                           NOP                     ;;ACT11
  (1)  012032                           $DOAGN:
  (1)  012032  000137                           JMP     @(PC)+          ;;RETURN
  (1)  012034  011674                   $RTNAD: .WORD   AGATST
  (1)  012036     377     377     000   $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)          012042                           .EVEN
```

```
                                            .SBTTL          ASCII MESSAGES
012042  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST/<15><12>
012050  020105  042524  052123
012056  005015     000
012061     015  051412  052105  SETMSG: .ASCIZ  <15><12>/SETTLING TEST/<15><12>
012066  046124  047111  020107
012074  042524  052123  005015
012102     000
012103     055     000          MINUS:  .BYTE   55,0
012105     077     000          QUEST:  .BYTE   77,0
012107     136     101     040  AMSG:   .BYTE   136,101,40,40,0
012112     040     000
012114     136     103     040  CMSG:   .BYTE   136,103,40,40,0
012117     040     000
012121     136     107     015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012124     012     123     127
012127     122     105     107
012132     072     000
012134  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
012142     000
012143     055  020055     000  DASH:   .ASCIZ  /-- /
012147     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012154  026455  053440  042111
012162  044124  005015     000
012167     103  000110          CH:     .ASCIZ  /CH/
012172  020040  020040     000  SPACE:  .ASCIZ  /    /
012177     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
012204  047117  041440  000110
012212  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
012220  047111  020107  051106
012226  046517  041440  000110
012234  040440  020124     000  ATMSG:  .ASCIZ  / AT /
012241     116  044517  042523  NOI:    .ASCIZ  /NOISE: /
012246  020072     000
012251     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
012256  040510  047116  046105
012264  000040
012266  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
012274  042516  005015     000
012301     057     000          SLASH:  .ASCIZ  #/#
012303     040  020040  047440  OKMSG:  .ASCIZ  /    OK/<15><12>
012310  006513  000012
012314  005015  054524  042520  CCHAN:  .ASCIZ  <15><12>/TYPE CHANNEL & CR: /
012322  041440  040510  047116
012330  046105  023040  041440
012336  035122  000040
012342  005015  054524  042520  SEL:    .ASCIZ  <15><12>/TYPE "O" FOR OFFSET, "G" FOR GAIN & CR: /
012350  021040  021117  043040
012356  051117  047440  043106
012364  042523  026124  021040
012372  021107  043040  051117
012400  043440  044501  020116
012406  020046  051103  020072
012414     000
```

```
 1392 012415       015  040412  045104  XADJ:   .ASCII  <15><12>/ADJUST R15/
      012422    051525  020124  030522
      012430       065
 1393 012431       040  047506  020122  MOLSB:  .ASCIZ  / FOR 0.00 LSB ERROR/
      012436    027060  030060  046040
      012444    041123  042440  051122
      012452    051117     000
 1394 012455       015  044412  050116  IGND:   .ASCII  <15><12>/INPUT A GROUND ON THE CHANNEL/
      012462    052125  040440  043440
      012470    047522  047125  020104
      012476    047117  052040  042510
      012504    041440  040510  047116
      012512    046105
 1395 012514    005015  054524  042520  CRWR:   .ASCIZ  <15><12>/TYPE CR WHEN READY/<15><12>
      012522    041440  020122  044127
      012530    047105  051040  040505
      012536    054504  005015     000
 1396 012543       015  044412  050116  IVOLT:  .ASCIZ  <15><12>/INPUT +5.115 VOLTS ON THE CHANNEL/
      012550    052125  025440  027065
      012556    030461  020065  047526
      012564    052114  020123  047117
      012572    052040  042510  041440
      012600    040510  047116  046105
      012606       000
 1397 012607       015  040412  045104  YADJ:   .ASCIZ  <15><12>/ADJUST R3/
      012614    051525  020124  031522
      012622       000
 1398 012623       053     000          POSITV: .ASCIZ  /+/
 1399 012625       040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
      012632    047522  025122  006452
      012640    000012
 1400 012642    051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
      012650    042105  051440  040524
      012656    042524  051450  000051
 1401 012664    047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
      012672    020127  036050  030440
      012700    031057  046040  041123
      012706    020051  052123  052101
      012714    024105  024523  005015
      012722       000
 1402 012723       040  044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
      012730    024040  020076  020061
      012736    027461  020062  051514
      012744    024502  051440  040524
      012752    042524  051450  006451
      012760    000012
 1403 012762    051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
      012770    051450  020051  044527
      012776    042504  020122  044124
      013004    047101  031040  046040
      013012    041123     000
 1405  013015     040  052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
       013022  026505  044527  052104
       013030  024110  024523  047440
       013036  052125  044523  042504
       013044  025440  047440  020122
       013052  020055  027461  020062
       013060  051514  000102
 1406  013064  046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
       013072  051515  020054     000
 1407  013077     040  051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
       013104  042520  045501  040440
       013112  020124     000
 1408  013115     015  042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
       013122  047440  020106  047514
       013130  044507  020103  042524
       013136  052123     123
 1409  013141     040  047117  040440  ONAD:   .ASCIZ  / ON ADV11 AT /
       013146  053104  030461  040440
       013154  020124     000
 1410  013157     040  042101  030526  MSG50:  .ASCIZ  / ADV11'S FOUND/<15><12>
       013164  023461  020123  047506
       013172  047125  006504  000012
 1411  013200  005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
       013206  020062  051514  006502
       013214  005012  005012  005012
       013222  005012  005012  005012
 1412  013230  030455  031057  051514          .ASCIZ  \-1/2LSB\
       013236  000102
 1413
 1414                                          .EVEN
 1415  013240  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
       013246  047105  044524  046101
       013254  046040  047111  040505
       013262  044522  054524  006472
       013270  000012
```

```
  1417 013272  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
       013300  020040  020040  020040
       013306  020040  020040  020040
       013314  020040  052123  052101
       013322  026505  044527  052104
       013330  020110  044504  052123
       013336  044522  052502  044524
       013344  047117  005015  005012
  1418 013352  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
       013360  051440  040524  042524
       013366  005123  005012  005012
       013374  005012  005012  005012
       013402  005012  005012  005012
       013410  005012
  1419 013412  020040  020040  020040          .ASCII  /                                                 STATE WIDTH (LSB)/<15>
       013420  020040  020040  020040
       013426  020040  020040  020040
       013434  020040  020040  020040
       013442  020040  020040  020040
       013450  020040  020040  020040
       013456  020040  020040  020040
       013464  020040  020040  020040
       013472  051440  040524  042524
       013500  053440  042111  044124
       013506  024040  051514  024502
       013514  005015
  1420 013516  030040  020040  020040          .ASCIZ  # 0              1/2             1            1 1/2             2#
       013524  020040  020040  020040
       013532  020040  020040  027461
       013540  020062  020040  020040
       013546  020040  020040  020040
       013554  020040  020061  020040
       013562  020040  020040  020040
       013570  020040  030440  030440
       013576  031057  020040  020040
       013604  020040  020040  020040
       013612  020040  031040     000
  1421 013617     015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR TEST: /
       013624  020105  042514  052124
       013632  051105  023040  041440
       013640  020122  047506  020122
       013646  042524  052123  020072
       013654     000
  1422 013655     122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
       013662  053111  020105  041501
       013670  052503  040522  054503
       013676  006472  000012
  1423 013702  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
       013710  054101  046511  046525
       013716  040440  020124     000
  1424 013723     015  050012  044522  HEAD5:  .ASCII  <15><12>/PRINT VALUES--/
       013730  052116  053040  046101
       013736  042525  026523     055
  1426  013743     040  042523  020124  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
        013750  044103  047101  042516
        013756  020114  047111  051440
        013764  051127  046040  053517
        013772  041040  052131  006505
        014000  000012
  1427  014002  055033     000          C0:     .ASCIZ  <33><132>
  1428  014005     033  015462  000110  C2:     .ASCIZ  <33><62><33><110>        ;CLEAR GRAPH MODE AND HOME
  1429  014012  000112                  C3:     .ASCIZ  <112>
  1430  014014  005015  043117  051506  MOFSET: .ASCIZ  <15><12>/OFFSET =/
        014022  052105  036440     000
  1431  014027     040  051514  020102  MLSB:   .ASCIZ  / LSB /
        014034     000
  1432  014035     040  052101  000040  MAT:    .ASCIZ  / AT /
  1433  014042  005015  042440  052116  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
        014050  051105  047111  020107
        014056  042524  052123  000040
  1434  014064     033     061     101  BUFF1:  .BYTE   33,61,101,61,111,62,114,41,60,45,63,51,66,55,71,61,74,110,41,40,112,0
        014067     061     111     062
        014072     114     041     060
        014075     045     063     051
        014100     066     055     071
        014103     061     074     110
        014106     041     040     112
        014111     000
  1435  014112     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
        014115     047     111     061
        014120     104     050     065
        014123     044     062     110
        014126     040     040     102
        014131     000
  1436  014132     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0      ;HOME & ERASE SCREEN & CLEAR GRA
        014135     112     033     061
        014140     101     040     033
        014143     062     000
```

```
 1438 014145      015  005012  042115  HEAD1:  .ASCII  <15><12><12>/ 'D-11-CVADA-C    ADV11 DIAGNOSTIC/<15><12>
      014152   030455  026461  053103
      014160   042101  026501  020103
      014166   020040  040440  053104
      014174   030461  042040  040511
      014202   047107  051517  044524
      014210   006503     012
 1439 014213      012  035101  040440          .ASCII  <12>/A: AUTO TEST/
      014220   052125  020117  042524
      014226   052123
 1440 014230   005015  035103  041440          .ASCII  <15><12>/C: CALIBRATION/
      014236   046101  041111  040522
      014244   044524  047117
 1441 014250   005015  035120  050040          .ASCII  <15><12>/P: PRINT VALUES/
      014256   044522  052116  053040
      014264   046101  042525     123
 1442 014271      015  046012  020072          .ASCII  <15><12>/L: LOGIC/
      014276   047514  044507     103
 1443 014303      015  053412  020072          .ASCIZ  <15><12>/W: WRAPAROUND/<15><12>
      014310   051127  050101  051101
      014316   052517  042116  005015
      014324      000
 1444 014325      123  040524  052524  EM1:    .ASCIZ  /STATUS REG. ERROR/
      014332   020123  042522  027107
      014340   042440  051122  051117
      014346      000
 1445 014347      106  044501  042514  EM2:    .ASCIZ  /FAILED TO INTERRUPT/
      014354   020104  047524  044440
      014362   052116  051105  052522
      014370   052120     000
 1446 014373      125  042516  050130  EM3:    .ASCIZ  /UNEXPECTED INTERRUPT/
      014400   041505  042524  020104
      014406   047111  042524  051122
      014414   050125  000124
 1447 014420   051105  047522  020122  EM4:    .ASCIZ  #ERROR ON A/D CHANNEL#
      014426   047117  040440  042057
      014434   041440  040510  047116
      014442   046105     000
 1449 014445     105  051122  041520  DH1:     .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/
      014452  051440  051124  043505
      014460  042440  050130  041505
      014466  042524  020104  041501
      014474  052524  046101     000
 1450 014501     105  051122  041520  DH2:     .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
      014506  020040  052123  042522
      014514  020107  020040  044103
      014522  047101  042516  020114
      014530  047040  046517  047111
      014536  046101  020040  047524
      014544  042514  040522  041516
      014552  020105  040440  052103
      014560  040525  000114
 1451 014564  051105  050122  020103  DH3:     .ASCIZ  /ERRPC      STREG    ACTUAL/
      014572  020040  020040  051440
      014600  051124  043505  020040
      014606  020040  041501  052524
      014614  046101     000
 1452 014617     000                  HUNS:    .BYTE   0
 1453 014620     056                  DECPNT:  .BYTE   56
 1454 014621     000                  TENS:    .BYTE   0
 1455 014622     000     000          ONES:    .BYTE   0,0
 1456                                 .EVEN
 1457
 1458 014624  001116  001316  001124  DT1:     $ERRPC, STREG, $GDDAT, $BDDAT,0
      014632  001126  000000
 1459 014636  001116  001316  001372  DT2:     $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
      014644  001124  001412  001126
      014652  000000
 1460 014654  001116  001316  001126  DT3:     $ERRPC,STREG,$BDDAT,0
      014662  000000
 1461 014664  000000                  DF1:     0
```

```
                                        .SBTTL  TTY INPUT ROUTINE
 (1)
 (2)                                    ;;*************************************************************
 (1)                                    .ENABL  LSB
 (1)
 (1)                                    .DSABL  LSB
 (1)
 (1)
 (2)                                    ;;*************************************************************
 (1)                                    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 (1)                                    ;*CALL:
 (1)                                    ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 (1)                                    ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 (1)                                    ;*                              ;;WITH PARITY BIT STRIPPED OFF
 (1)                                    ;
 (1)
 (1) 014666 011646                      $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 (1) 014670 016666 000004 000002                MOV     4(SP),2(SP)     ;;SAVE THE PS
 (1) 014676 105777 164242               1$:     TSTB    @$TKS           ;;WAIT FOR
 (1) 014702 100375                              BPL     1$              ;;A CHARACTER
 (1) 014704 117766 164236 000004                MOVB    @$TKB,4(SP)     ;;READ THE TTY
 (1) 014712 042766 177600 000004                BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 (1) 014720 026627 000004 000023                CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 (1) 014726 001013                              BNE     3$              ;;BRANCH IF NO
 (1) 014730 105777 164210               2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 (1) 014734 100375                              BPL     2$              ;;LOOP UNTIL ITS THERE
 (1) 014736 117746 164204                       MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 (1) 014742 042716 177600                       BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 (1) 014746 022627 000021                       CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 (1) 014752 001366                              BNE     2$              ;;IF NOT DISCARD IT
 (1) 014754 000750                              BR      1$              ;;YES, RESUME
 (1) 014756 026627 000004 000140        3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 (1) 014764 002407                              BLT     4$              ;;BRANCH IF YES
 (1) 014766 026627 000004 000175                CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 (1) 014774 003003                              BGT     4$              ;;BRANCH IF YES
 (1) 014776 042766 000040 000004                BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 (1) 015004 000002                      4$:     RTI                     ;;GO BACK TO USER
 (2)                                    ;;*************************************************************
 (1)                                    ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 (1)                                    ;*CALL:
 (1)                                    ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 (1)                                    ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 (1)                                    ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 (1)
 (1) 015006 010346                      $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 (1) 015010 012703 015114               1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 (1) 015014 022703 015124               2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 (1) 015020 101405                              BLOS    4$              ;;BR IF YES
 (1) 015022 104406                              RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 (1) 015024 112613                              MOVB    (SP)+,(R3)      ;;GET CHARACTER
 (1) 015026 122713 000177               10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 (1) 015032 001003                              BNE     3$              ;;SKIP IF NOT
 (1) 015034 104401 001170               4$:     TYPE    ,$QUES          ;;TYPE A '?'
 (1) 015040 000763                              BR      1$              ;;CLEAR THE BUFFER AND LOOP
 (1) 015042 111337 015112               3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 (1) 015046 104401 015112                       TYPE    ,9$
```

```
   (1)  015052  122723  000015                    CMPB    #15,(R3)+           ;;CHECK FOR RETURN
   (1)  015056  001356                            BNE     2$                  ;;LOOP IF NOT RETURN
   (1)  015060  105063  177777                    CLRB    -1(R3)              ;;CLEAR RETURN (THE 15)
   (1)  015064  104401  001172                    TYPE    ,$LF                ;;TYPE A LINE FEED
   (1)  015070  012603                            MOV     (SP)+,R3            ;;RESTORE R3
   (1)  015072  011646                            MOV     (SP),-(SP)          ;;ADJUST THE STACK AND PUT ADDRESS OF THE
   (1)  015074  016666  000004  000002            MOV     4(SP),2(SP)         ;;       FIRST ASCII CHARACTER ON IT
   (1)  015102  012766  015114  ^00004            MOV     #$TTYIN,4(SP)
   (1)  015110  000002                            RTI                         ;;RETURN
   (1)  015112     000                    9$:     .BYTE   0                   ;;STORAGE FOR ASCII CHAR. TO TYPE
   (1)  015113     000                            .BYTE   0                   ;;TERMINATOR
   (1)  015114  000010                    $TTYIN: .BLKB   8.                  ;;RESERVE 8 BYTES FOR TTY INPUT
   (1)  015124  052536  005015     000    $CNTLU: .ASCIZ  /^U/<15><12>        ;;CONTROL ''U''
   (1)  015131     136  006507  000012    $CNTLG: .ASCIZ  /^G/<15><12>        ;;CONTROL ''G''
   (1)  015136  005015  053523  020122    $MSWR:  .ASCIZ  <15><12>/SWR = /
   (1)  015144  020075     000
   (1)  015147     040  047040  053505    $MNEW:  .ASCIZ  /  NEW = /
   (1)  015154  036440  000040
```

```
 1465                                   .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
  (1)                                   ;*CHANGE IT TO BINARY.
  (1)                                   ;*CALL:
  (1)                                   ;*      RDOCT                           ;;READ AN OCTAL NUMBER
  (1)                                   ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
  (1)                                   ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
  (1)
  (1)   015160  011646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
  (1)   015162  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
  (3)   015170  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
  (3)   015172  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
  (3)   015174  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
  (1)   015176  104407                  1$:     RDLIN                           ;;READ AN ASCIZ LINE
  (1)   015200  012600                          MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
  (1)   015202  005001                          CLR     R1                      ;;CLEAR DATA WORD
  (1)   015204  005002                          CLR     R2
  (1)   015206  112046                  2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
  (1)   015210  001412                          BEQ     3$                      ;;IF ZERO GET OUT
  (1)   015212  006301                          ASL     R1                      ;;*2
  (1)   015214  006102                          ROL     R2
  (1)   015216  006301                          ASL     R1                      ;;*4
  (1)   015220  006102                          ROL     R2
  (1)   015222  006301                          ASL     R1                      ;;*8
  (1)   015224  006102                          ROL     R2
  (1)   015226  042716  177770                  BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
  (1)   015232  062601                          ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
  (1)   015234  000764                          BR      2$                      ;;LOOP
  (1)   015236  005726                  3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
  (1)   015240  010166  000012                  MOV     R1,12(SP)               ;;SAVE THE RESULT
  (1)   015244  010237  015260                  MOV     R2,$HIOCT
  (3)   015250  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
  (3)   015252  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
  (3)   015254  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
  (1)   015256  000002                          RTI                             ;;RETURN
  (1)   015260  000000                  $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
 1467                                        .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                                        ;;*************************************************************
  (1)                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
  (1)                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
  (1)                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                        ;*SW14=1        LOOP ON TEST
  (1)                                        ;*SW11=1        INHIBIT ITERATIONS
  (1)                                        ;*SW09=1        LOOP ON ERROR
  (1)                                        ;*SW08=1        LOOP ON TEST IN SWR<7:0>
  (1)                                        ;*CALL
  (1)                                        ;*      SCOPE           ;;SCOPE=IOT
  (1)
  (1)  015262                                $SCOPE:
  (1)  015262  032777  040000  163650        1$:     BIT     #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
  (1)  015270  001114                                BNE     $OVER            ;;YES IF SW14=1
  (1)                                        ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  015272  000416                        $XTSTR: BR      6$               ;;IF RUNNING ON THE "XOR" TESTER CHANGE
  (1)                                                                         ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
  (1)  015274  013746  000004                        MOV     @#ERRVEC,-(SP)   ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  015300  012737  015320  000004                MOV     #5$,@#ERRVEC     ;;SET FOR TIMEOUT
  (1)  015306  005737  177060                        TST     @#177060         ;;TIME OUT ON XOR?
  (1)  015312  012637  000004                        MOV     (SP)+,@#ERRVEC   ;;RESTORE THE ERROR VECTOR
  (1)  015316  000463                                BR      $SVLAD           ;;GO TO THE NEXT TEST
  (1)  015320  022626                        5$:     CMP     (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  015322  012637  000004                        MOV     (SP)+,@#ERRVEC   ;;RESTORE THE ERROR VECTOR
  (1)  015326  000423                                BR      7$               ;;LOOP ON THE PRESENT TEST
  (1)  015330                                6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  015330  032777  000400  163602                BIT     #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
  (1)  015336  001404                                BEQ     2$               ;;BR IF NO
  (1)  015340  127737  163574  001102                CMPB    @SWR,$TSTNM      ;;ON THE RIGHT TEST?   SWR<7:0>
  (1)  015346  001465                                BEQ     $OVER            ;;BR IF YES
  (1)  015350  105737  001103                2$:     TSTB    $ERFLG           ;;HAS AN ERROR OCCURRED?
  (1)  015354  001421                                BEQ     3$               ;;BR IF NO
  (1)  015356  123737  001115  001103                CMPB    $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  015364  101015                                BHI     3$               ;;BR IF NO
  (1)  015366  032777  001000  163544                BIT     #BIT09,@SWR      ;;LOOP ON ERROR?
  (1)  015374  001404                                BEQ     4$               ;;BR IF NO
  (1)  015376  013737  001110  001106        7$:     MOV     $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
  (1)  015404  000446                                BR      $OVER
  (1)  015406  105037  001103                4$:     CLRB    $ERFLG           ;;ZERO THE ERROR FLAG
  (1)  015412  005037  001160                        CLR     $TIMES           ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  (1)  015416  000415                                BR      1$               ;;ESCAPE TO THE NEXT TEST
  (1)  015420  032777  004000  163512        3$:     BIT     #BIT11,@SWR      ;;INHIBIT ITERATIONS?
  (1)  015426  001011                                BNE     1$               ;;BR IF YES
  (1)  015430  005737  001202                        TST     $PASS            ;;IF FIRST PASS OF PROGRAM
  (1)  015434  001406                                BEQ     1$               ;;      INHIBIT ITERATIONS
  (1)  015436  005237  001104                        INC     $ICNT            ;;INCREMENT ITERATION COUNT
  (1)  015442  023737  001160  001104                CMP     $TIMES,$ICNT     ;;CHECK THE NUMBER OF ITERATIONS MADE
  (1)  015450  002024                                BGE     $OVER            ;;BR IF MORE ITERATION REQUIRED
  (1)  015452  012737  000001  001104        1$:     MOV     #1,$ICNT         ;;REINITIALIZE THE ITERATION COUNTER
  (1)  015460  013737  015536  001160                MOV     $MXCNT,$TIMES    ;;SET NUMBER OF ITERATIONS TO DO
  (1)  015466  105237  001102                $SVLAD: INCB    $TSTNM           ;;COUNT TEST NUMBERS
  (1)  015472  113737  001102  001200                MOVB    $TSTNM,$TESTN    ;;SET TEST NUMBER IN APT MAILBOX
  (1)  015500  011637  001106                        MOV     (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
```

```
   (1)  015504  011637  001110                  MOV     (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
   (1)  015510  005037  001162                  CLR     $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
   (1)  015514  112737  000001  001115          MOVB    #1,$ERMAX        ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
   (1)  015522  013777  001102  163412  $OVER:  MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER
   (1)  015530  013716  001106                  MOV     $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
   (1)  015534  000002                          RTI                      ;;FIXES PS
   (1)  015536  003720                  $MXCNT: 2000.                    ;;MAX. NUMBER OF ITERATIONS
  1468                                  .SBTTL  ERROR HANDLER ROUTINE
   (1)
   (2)                                  ;;*************************************************************
   (1)                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
   (1)                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
   (1)                                  ;*AND GO TO $ERRTYP ON ERROR
   (1)                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
   (1)                                  ;*SW15=1        HALT ON ERROR
   (1)                                  ;*SW13=1        INHIBIT ERROR TYPEOUTS
   (1)                                  ;*SW10=1        BELL ON ERROR
   (1)                                  ;*SW09=1        LOOP ON ERROR
   (1)                                  ;*CALL
   (1)                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
   (1)
   (1)  015540                          $ERROR:
   (3)  015540  043737  001440  001436          BIC     TSTBIT,GUNITS
   (1)  015546  105237  001103          7$:     INCB    $ERFLG           ;;SET THE ERROR FLAG
   (1)  015552  001775                          BEQ     7$               ;;DON'T LET THE FLAG GO TO ZERO
   (1)  015554  013777  001102  163360          MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
   (1)  015562  032777  002000  163350          BIT     #BIT10,@SWR      ;;BELL ON ERROR?
   (1)  015570  001402                          BEQ     1$               ;;NO - SKIP
   (1)  015572  104401  001164                  TYPE    ,$BELL           ;;RING BELL
   (1)  015576  005237  001112          1$:     INC     $ERTTL           ;;COUNT THE NUMBER OF ERRORS
   (1)  015602  011637  001116                  MOV     (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
   (1)  015606  162737  000002  001116          SUB     #2,$ERRPC
   (1)  015614  117737  163276  001114          MOVB    @$ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
   (1)  015622  032777  020000  163310          BIT     #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
   (1)  015630  001004                          BNE     20$              ;;SKIP TYPEOUTS
   (1)  015632  004737  015742                  JSR     PC,$ERRTYP       ;;GO TO USER ERROR ROUTINE
   (1)  015636  104401  001171                  TYPE    ,$CRLF
   (1)  015642                          20$:
   (1)  015642  122737  000001  001214          CMPB    #APTENV,$ENV     ;;RUNNING IN APT MODE
   (1)  015650  001007                          BNE     2$               ;;NO,SKIP APT ERROR REPORT
   (1)  015652  113737  001114  015664          MOVB    $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
   (1)  015660  004737  016376                  JSR     PC,$ATY4         ;;REPORT FATAL ERROR TO APT
   (1)  015664     000                  21$:    .BYTE   0
   (1)  015665     000                          .BYTE   0
   (1)  015666  000777                  22$:    BR      22$              ;;APT ERROR LOOP
   (1)  015670  005777  163244          2$:     TST     @SWR             ;;HALT ON ERROR
   (1)  015674  100001                          BPL     3$               ;;SKIP IF CONTINUE
   (1)  015676  000000                          HALT                     ;;HALT ON ERROR!
   (1)  015700  032777  001000  163232  3$:     BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
   (1)  015706  001402                          BEQ     4$               ;;BR IF NO
   (1)  015710  013716  001110                  MOV     $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
   (1)  015714  005737  001162          4$:     TST     $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
   (1)  015720  001402                          BEQ     5$               ;;BR IF NONE
   (1)  015722  013716  001162                  MOV     $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
   (1)  015726                          5$:
   (1)  015726  022737  012022  000042          CMP     #$ENDAD,@#42     ;;ACT-11 AUTO-ACCEPT?
```

```
 (1)  015734  001001                            BNE     6$              ;;BRANCH IF NO
 (1)  015736  000000                            HALT                    ;;YES
 (1)  015740                            6$:
 (1)  015740  000002                            RTI                     ;;RETURN
                                        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 (1)
 (2)                                    ;;*******************************************************************
 (1)                                    ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 (1)                                    ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 (1)                                    ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 (1)
 (1)  015742                            $ERRTYP:
 (1)  015742  104401  001171                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  015746  010046                            MOV     R0,-(SP)        ;;SAVE R0
 (1)  015750  005000                            CLR     R0              ;;PICKUP THE ITEM INDEX
 (1)  015752  153700  001114                    BISB    @#$ITEMB,R0
 (1)  015756  001004                            BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 (1)                                                                    ;;TYPE THE PC OF THE ERROR
 (2)  015760  013746  001116                    MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 (2)                                                                    ;;ERROR ADDRESS
 (2)  015764  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  015766  000426                            BR      6$              ;;GET OUT
 (1)  015770  005300                    1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 (1)  015772  006300                            ASL     R0              ;;       WORK FOR THE ERROR TABLE
 (1)  015774  006300                            ASL     R0
 (1)  015776  006300                            ASL     R0
 (1)  016000  062700  001256                    ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 (1)  016004  012037  016014                    MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 (1)  016010  001404                            BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 (1)  016012  104401                            TYPE                    ;;TYPE THE "ERROR MESSAGE"
 (1)  016014  000000                    2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
 (1)  016016  104401  001171                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  016022  012037  016032            3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
 (1)  016026  001404                            BEQ     5$              ;;SKIP TYPEOUT IF 0
 (1)  016030  104401                            TYPE                    ;;TYPE THE "DATA HEADER"
 (1)  016032  000000                    4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
 (1)  016034  104401  001171                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  016040  011000                    5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
 (1)  016042  001004                            BNE     7$              ;;GO TYPE THE DATA
 (1)  016044  012600                    6$:     MOV     (SP)+,R0        ;;RESTORE R0
 (1)  016046  104401  001171                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  016052  000207                            RTS     PC              ;;RETURN
 (1)  016054                            7$:
 (2)  016054  013046                            MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
 (2)  016056  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  016060  005710                            TST     (R0)            ;;IS THERE ANOTHER NUMBER?
 (1)  016062  001770                            BEQ     6$              ;;BR IF NO
 (1)  016064  104401  016072                    TYPE    ,8$             ;;TYPE TWO(2) SPACES
 (1)  016070  000771                            BR      7$              ;;LOOP
 (1)  016072  020040     000            8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
 (1)          016076                            .EVEN
```

```
                                          .SBTTL  TYPE ROUTINE
 (1)
 (2)                                      ;;*******************************************************************
 (1)                                      ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 (1)                                      ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 (1)                                      ;;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 (1)                                      ;;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 (1)                                      ;;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 (1)                                      ;;*
 (1)                                      ;;*CALL:
 (1)                                      ;;*1) USING A TRAP INSTRUCTION
 (1)                                      ;;*       TYPE     ,MESADR           ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 (1)                                      ;;*OR
 (1)                                      ;;*       TYPE
 (1)                                      ;;*       MESADR
 (1)                                      ;;*
 (1)
 (1)  016076  105737  001157              $TYPE:  TSTB     $TPFLG            ;;IS THERE A TERMINAL?
 (1)  016102  100002                              BPL      1$                ;;BR IF YES
 (1)  016104  000000                              HALT                       ;;HALT HERE IF NO TERMINAL
 (1)  016106  000430                              BR       3$                ;;LEAVE
 (1)  016110  010046                      1$:     MOV      R0,-(SP)          ;;SAVE R0
 (1)  016112  017600  000002                      MOV      @2(SP),R0         ;;GET ADDRESS OF ASCIZ STRING
 (1)  016116  122737  000001  001214              CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
 (1)  016124  001011                              BNE      62$               ;;NO,GO CHECK FOR APT CONSOLE
 (1)  016126  132737  000100  001215              BITB     #APTSPOOL,$ENVM   ;;SPOOL MESSAGE TO APT
 (1)  016134  001405                              BEQ      62$               ;;NO,GO CHECK FOR CONSOLE
 (1)  016136  010037  016146                      MOV      R0,61$            ;;SETUP MESSAGE ADDRESS FOR APT
 (1)  016142  004737  016366                      JSR      PC,$ATY3          ;;SPOOL MESSAGE TO APT
 (1)  016146  000000                      61$:    .WORD    0                 ;;MESSAGE ADDRESS
 (1)  016150  132737  000040  001215      62$:    BITB     #APTCSUP,$ENVM    ;;APT CONSOLE SUPPRESSED
 (1)  016156  001003                              BNE      60$               ;;YES,SKIP TYPE OUT
 (1)  016160  112046                      2$:     MOVB     (R0)+,-(SP)       ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 (1)  016162  001005                              BNE      4$                ;;BR IF IT ISN'T THE TERMINATOR
 (1)  016164  005726                              TST      (SP)+             ;;IF TERMINATOR POP IT OFF THE STACK
 (1)  016166  012600                      60$:    MOV      (SP)+,R0          ;;RESTORE R0
 (1)  016170  062716  000002              3$:     ADD      #2,(SP)           ;;ADJUST RETURN PC
 (1)  016174  000002                              RTI                        ;;RETURN
 (1)  016176  122716  000011              4$:     CMPB     #HT,(SP)          ;;BRANCH IF <HT>
 (1)  016202  001430                              BEQ      8$
 (1)  016204  122716  000200                      CMPB     #CRLF,(SP)        ;;BRANCH IF NOT <CRLF>
 (1)  016210  001006                              BNE      5$
 (1)  016212  005726                              TST      (SP)+             ;;POP  <CR><LF> EQUIV
 (1)  016214  104401                              TYPE                       ;;TYPE A CR AND LF
 (1)  016216  001171                              $CRLF
 (1)  016220  105037  016354                      CLRB     $CHARCNT          ;;CLEAR CHARACTER COUNT
 (1)  016224  000755                              BR       2$                ;;GET NEXT CHARACTER
 (1)  016226  004737  016310              5$:     JSR      PC,$TYPEC         ;;GO TYPE THIS CHARACTER
 (1)  016232  123726  001156              6$:     CMPB     $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
 (1)  016236  001350                              BNE      2$                ;;IF NO GO GET NEXT CHAR.
 (1)  016240  013746  001154                      MOV      $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
 (1)                                                                         ;;AND THE NULL CHAR.
 (1)  016244  105366  000001              7$:     DECB     1(SP)             ;;DOES A NULL NEED TO BE TYPED?
 (1)  016250  002770                              BLT      6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
 (1)  016252  004737  016310                      JSR      PC,$TYPEC         ;;GO TYPE A NULL
 (1)  016256  105337  016354                      DECB     $CHARCNT          ;;DO NOT COUNT AS A COUNT
```

```
   (1)  016262  000770                                 BR      7$                 ;;LOOP
   (1)
   (1)                                         ;HORIZONTAL TAB PROCESSOR
   (1)
   (1)  016264  112716  000040                 8$:     MOVB    #' ,(SP)           ;;REPLACE TAB WITH SPACE
   (1)  016270  004737  016310                 9$:     JSR     PC,$TYPEC          ;;TYPE A SPACE
   (1)  016274  132737  000007  016354                 BITB    #7,$CHARCNT        ;;BRANCH IF NOT AT
   (1)  016302  001372                                 BNE     9$                 ;;TAB STOP
   (1)  016304  005726                                 TST     (SP)+              ;;POP SPACE OFF STACK
   (1)  016306  000724                                 BR      2$                 ;;GET NEXT CHARACTER
   (1)  016310  105777  162634                 $TYPEC: TSTB    @$TPS              ;;WAIT UNTIL PRINTER IS READY
   (1)  016314  100375                                 BPL     $TYPEC
   (1)  016316  116677  000002  162626                 MOVB    2(SP),@$TPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
   (1)  016324  122766  000015  000002                 CMPB    #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
   (1)  016332  001003                                 BNE     1$                 ;;BRANCH IF NO
   (1)  016334  105037  016354                         CLRB    $CHARCNT           ;;YES--CLEAR CHARACTER COUNT
   (1)  016340  000406                                 BR      $TYPEX             ;;EXIT
   (1)  016342  122766  000012  000002  1$:    CMPB    #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
   (1)  016350  001402                                 BEQ     $TYPEX             ;;BRANCH IF YES
   (1)  016352  105227                                 INCB    (PC)+              ;;COUNT THE CHARACTER
   (1)  016354  000000                         $CHARCNT:.WORD  0                  ;;CHARACTER COUNT STORAGE
   (1)  016356  000207                         $TYPEX: RTS     PC
   (1)
  1472                                         .SBTTL  APT COMMUNICATIONS ROUTINE
   (1)
   (2)                                         ;;*****************************************************************
   (1)  016360  112737  000001  016624  $ATY1:  MOVB    #1,$FFLG           ;;TO REPORT FATAL ERROR
   (1)  016366  112737  000001  016622  $ATY3:  MOVB    #1,$MFLG           ;;TO TYPE A MESSAGE
   (1)  016374  000403                                 BR      $ATYC
   (1)  016376  112737  000001  016624  $ATY4:  MOVB    #1,$FFLG           ;;TO ONLY REPORT FATAL ERROR
   (1)  016404                                 $ATYC:
   (3)  016404  010046                                 MOV     R0,-(SP)           ;;PUSH R0 ON STACK
   (3)  016406  010146                                 MOV     R1,-(SP)           ;;PUSH R1 ON STACK
   (1)  016410  105737  016622                         TSTB    $MFLG              ;;SHOULD TYPE A MESSAGE?
   (1)  016414  001450                                 BEQ     5$                 ;;IF NOT:  BR
   (1)  016416  122737  000001  001214                 CMPB    #APTENV,$ENV       ;;OPERATING UNDER APT?
   (1)  016424  001031                                 BNE     3$                 ;;IF NOT:  BR
   (1)  016426  132737  000100  001215                 BITB    #APTSPOOL,$ENVM    ;;SHOULD SPOOL MESSAGES?
   (1)  016434  001425                                 BEQ     3$                 ;;IF NOT:  BR
   (1)  016436  017600  000004                         MOV     @4(SP),R0          ;;GET MESSAGE ADDR.
   (1)  016442  062766  000002  000004                 ADD     #2,4(SP)                   ;;BUMP RETURN ADDR.
   (1)  016450  005737  001174                 1$:     TST     $MSGTYPE           ;;SEE IF DONE W/ LAST XMISSION?
   (1)  016454  001375                                 BNE     1$                 ;;IF NOT:  WAIT
   (1)  016456  010037  001210                         MOV     R0,$MSGAD          ;;PUT ADDR IN MAILBOX
   (1)  016462  105720                         2$:     TSTB    (R0)+              ;;FIND END OF MESSAGE
   (1)  016464  001376                                 BNE     2$
   (1)  016466  163700  001210                         SUB     $MSGAD,R0          ;;SUB START OF MESSAGE
   (1)  016472  006200                                 ASR     R0                 ;;GET MESSAGE LNGTH IN WORDS
   (1)  016474  010037  001212                         MOV     R0,$MSGLGT         ;;PUT LENGTH IN MAILBOX
   (1)  016500  012737  000004  001174                 MOV     #4,$MSGTYPE        ;;TELL APT TO TAKE MSG.
   (1)  016506  000413                                 BR      5$
   (1)  016510  017637  000004  016534  3$:     MOV     @4(SP),4$          ;;PUT MSG ADDR IN JSR LINKAGE
   (1)  016516  062766  000002  000004                 ADD     #2,4(SP)                   ;;BUMP RETURN ADDRESS.
   (3)  016524  013746  177776                         MOV     177776,-(SP)       ;;PUSH 177776 ON STACK
   (1)  016530  004737  016076                         JSR     PC,$TYPE           ;;CALL TYPE MACRO
   (1)  016534  000000                         4$:     .WORD   0
```

```
    (1)  016536                          5$:
    (1)  016536  105737  016624          10$:     TSTB     $FFLG            ;;SHOULD REPORT FATAL ERROR?
    (1)  016542  001416                           BEQ      12$              ;;IF NOT:  BR
    (1)  016544  005737  001214                   TST      $ENV             ;;RUNNING UNDER APT?
    (1)  016550  001413                           BEQ      12$              ;;IF NOT:  BR
    (1)  016552  005737  001174          11$:     TST      $MSGTYPE         ;;FINISHED LAST MESSAGE?
    (1)  016556  001375                           BNE      11$              ;;IF NOT:  WAIT
    (1)  016560  017637  000004  001176           MOV      @4(SP),$FATAL    ;;GET ERROR #
    (1)  016566  062766  000002  000004           ADD      #2,4(SP)                 ;;BUMP RETURN ADDR.
    (1)  016574  005237  001174                   INC      $MSGTYPE         ;;TELL APT TO TAKE ERROR
    (1)  016600  105037  016624          12$:     CLRB     $FFLG            ;;CLEAR FATAL FLAG
    (1)  016604  105037  016623                   CLRB     $LFLG            ;;CLEAR LOG FLAG
    (1)  016610  105037  016622                   CLRB     $MFLG            ;;CLEAR MESSAGE FLAG
    (3)  016614  012601                           MOV      (SP)+,R1         ;;POP STACK INTO R1
    (3)  016616  012600                           MOV      (SP)+,R0         ;;POP STACK INTO R0
    (1)  016620  000207                           RTS      PC               ;;RETURN
    (1)  016622     000                  $MFLG:   .BYTE    0                ;;MESSG. FLAG
    (1)  016623     000                  $LFLG:   .BYTE    0                ;;LOG FLAG
    (1)  016624     000                  $FFLG:   .BYTE    0                ;;FATAL FLAG
    (1)          016626                           .EVEN
    (1)          000200                  APTSIZE=200
    (1)          000001                  APTENV=001
    (1)          000100                  APTSPOOL=100
    (1)          000040                  APTCSUP=040
```

```
                                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 (1)
 (2)                                            ;;*************************************************************
 (1)                                            ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 (1)                                            ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 (1)                                            ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 (1)                                            ;*CALL:
 (1)                                            ;*       MOV      NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                            ;*       TYPOS                    ;;CALL FOR TYPEOUT
 (1)                                            ;*       .BYTE    N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 (1)                                            ;*       .BYTE    M               ;;M=1 OR 0
 (1)                                            ;*                                        ;;1=TYPE LEADING ZEROS
 (1)                                            ;*                                        ;;0=SUPPRESS LEADING ZEROS
 (1)                                            ;*
 (1)                                            ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 (1)                                            ;*$TYPOS OR $TYPOC
 (1)                                            ;*CALL:
 (1)                                            ;*       MOV      NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                            ;*       TYPON                    ;;CALL FOR TYPEOUT
 (1)                                            ;*
 (1)                                            ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 (1)                                            ;*CALL:
 (1)                                            ;*       MOV      NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                            ;*       TYPOC                    ;;CALL FOR TYPEOUT
 (1)
 (1)  016626  017646  000000                    $TYPOS: MOV      @(SP),-(SP)     ;;PICKUP THE MODE
 (1)  016632  116637  000001  017051                    MOVB     1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 (1)  016640  112637  017053                            MOVB     (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 (1)  016644  062716  000002                            ADD      #2,(SP)         ;;ADJUST RETURN ADDRESS
 (1)  016650  000406                                    BR       $TYPON
 (1)  016652  112737  000001  017051        $TYPOC: MOVB     #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 (1)  016660  112737  000006  017053                    MOVB     #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 (1)  016666  112737  000005  017050        $TYPON: MOVB     #5,$OCNT        ;;SET THE ITERATION COUNT
 (1)  016674  010346                                    MOV      R3,-(SP)        ;;SAVE R3
 (1)  016676  010446                                    MOV      R4,-(SP)        ;;SAVE R4
 (1)  016700  010546                                    MOV      R5,-(SP)        ;;SAVE R5
 (1)  016702  113704  017053                            MOVB     $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 (1)  016706  005404                                    NEG      R4
 (1)  016710  062704  000006                            ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 (1)  016714  110437  017052                            MOVB     R4,$OMODE       ;;SAVE IT FOR USE
 (1)  016720  113704  017051                            MOVB     $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 (1)  016724  016605  000012                            MOV      12(SP),R5       ;;PICKUP THE INPUT NUMBER
 (1)  016730  005003                                    CLR      R3              ;;CLEAR THE OUTPUT WORD
 (1)  016732  006105                            1$:     ROL      R5              ;;ROTATE MSB INTO "C"
 (1)  016734  000404                                    BR       3$              ;;GO DO MSB
 (1)  016736  006105                            2$:     ROL      R5              ;;FORM THIS DIGIT
 (1)  016740  006105                                    ROL      R5
 (1)  016742  006105                                    ROL      R5
 (1)  016744  010503                                    MOV      R5,R3
 (1)  016746  006103                            3$:     ROL      R3              ;;GET LSB OF THIS DIGIT
 (1)  016750  105337  017052                            DECB     $OMODE          ;;TYPE THIS DIGIT?
 (1)  016754  100016                                    BPL      7$              ;;BR IF NO
 (1)  016756  042703  177770                            BIC      #177770,R3      ;;GET RID OF JUNK
 (1)  016762  001002                                    BNE      4$              ;;TEST FOR 0
 (1)  016764  005704                                    TST      R4              ;;SUPPRESS THIS 0?
 (1)  016766  001403                                    BEQ      5$              ;;BR IF YES
```

```
   (1)  016770  005204                   4$:     INC     R4                ;;DON'T SUPPRESS ANYMORE 0'S
   (1)  016772  052703  000060                   BIS     #'0,R3            ;;MAKE THIS DIGIT ASCII
   (1)  016776  052703  000040           5$:     BIS     #' ,R3            ;;MAKE ASCII IF NOT ALREADY
   (1)  017002  110337  017046                   MOVB    R3,8$             ;;SAVE FOR TYPING
   (1)  017006  104401  017046                   TYPE    ,8$               ;;GO TYPE THIS DIGIT
   (1)  017012  105337  017050           7$:     DECB    $OCNT             ;;COUNT BY 1
   (1)  017016  003347                           BGT     2$                ;;BR IF MORE TO DO
   (1)  017020  002402                           BLT     6$                ;;BR IF DONE
   (1)  017022  005204                           INC     R4                ;;INSURE LAST DIGIT ISN'T A BLANK
   (1)  017024  000744                           BR      2$                ;;GO DO THE LAST DIGIT
   (1)  017026  012605                   6$:     MOV     (SP)+,R5          ;;RESTORE R5
   (1)  017030  012604                           MOV     (SP)+,R4          ;;RESTORE R4
   (1)  017032  012603                           MOV     (SP)+,R3          ;;RESTORE R3
   (1)  017034  016666  000002  000004           MOV     2(SP),4(SP)       ;;SET THE STACK FOR RETURNING
   (1)  017042  012616                           MOV     (SP)+,(SP)
   (1)  017044  000002                           RTI                       ;;RETURN
   (1)  017046     000                   8$:     .BYTE   0                 ;;STORAGE FOR ASCII DIGIT
   (1)  017047     000                           .BYTE   0                 ;;TERMINATOR FOR TYPE ROUTINE
   (1)  017050     000                   $OCNT:  .BYTE   0                 ;;OCTAL DIGIT COUNTER
   (1)  017051     000                   $OFILL: .BYTE   0                 ;;ZERO FILL SWITCH
   (1)  017052  000000                   $OMODE: .WORD   0                 ;;NUMBER OF DIGITS TO TYPE
                                         .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
   (1)
   (2)                                   ;;*************************************************************
   (1)                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
   (1)                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
   (1)                                   ;*CALL:
   (1)                                   ;*      MOV     NUMBER,-(SP)      ;;NUMBER TO BE TYPED
   (1)                                   ;*      TYPBN                     ;;TYPE IT
   (1)
   (1)  017054  010146                   $TYPBN: MOV     R1,-(SP)          ;;SAVE R1 ON THE STACK
   (1)  017056  016601  000006                   MOV     6(SP),R1          ;;GET THE INPUT NUMBER
   (1)  017062  000261                           SEC                       ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
   (1)  017064  112737  000060  017126   1$:     MOVB    #'0,$BIN          ;;SET CHARACTER TO AN ASCII "0".
   (1)  017072  006101                           ROL     R1                ;;GET THIS BIT
   (1)  017074  001406                           BEQ     2$                ;;DONE?
   (1)  017076  105537  017126                   ADCB    $BIN              ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
   (1)  017102  104401  017126                   TYPE    ,$BIN             ;;GO TYPE THIS BIT
   (1)  017106  000241                           CLC                       ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
   (1)  017110  000765                           BR      1$                ;;GO DO THE NEXT BIT
   (1)  017112  012601                   2$:     MOV     (SP)+,R1          ;;POP THE STACK INTO R1
   (1)  017114  016666  000002  000004           MOV     2(SP),4(SP)       ;;ADJUST THE STACK
   (1)  017122  012616                           MOV     (SP)+,(SP)
   (1)  017124  000002                           RTI                       ;;RETURN TO USER
   (1)  017126     000     000           $BIN:   .BYTE   0,0               ;;STORAGE FOR ASCII CHAP. AND TERMINATOR
```

```
 1477                                    .SBTTL  TRAP DECODER
  (1)
  (2)                                    ;;*******************************************************************
  (1)                                    ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
  (1)                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
  (1)                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
  (1)                                    ;*GO TO THAT ROUTINE.
  (1)
  (1)   017130  010046                   $TRAP:  MOV     R0,-(SP)          ;;SAVE R0
  (1)   017132  016600  000002                   MOV     2(SP),R0          ;;GET TRAP ADDRESS
  (1)   017136  005740                           TST     -(R0)             ;;BACKUP BY 2
  (1)   017140  111000                           MOVB    (R0),R0           ;;GET RIGHT BYTE OF TRAP
  (1)   017142  006300                           ASL     R0                ;;POSITION FOR INDEXING
  (1)   017144  016000  017164                   MOV     $TRPAD(R0),R0     ;;INDEX TO TABLE
  (1)   017150  000200                           RTS     R0                ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                    ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
  (1)
  (1)   017152  011646                   $TRAP2: MOV     (SP),-(SP)        ;;MOVE THE PC DOWN
  (1)   017154  016666  000004  000002           MOV     4(SP),2(SP)       ;;MOVE THE PSW DOWN
  (1)   017162  000002                           RTI                       ;;RESTORE THE PSW
  (1)
  (3)                                    .SBTTL  TRAP TABLE
  (3)
  (3)                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
  (3)                                    ;*BY THE "TRAP" INSTRUCTION.
  (3)
  (3)                                    ;       ROUTINE
  (3)                                    ;       -------
  (3)   017164  017152                   $TRPAD: .WORD   $TRAP2
  (3)   017166  016076                           $TYPE   ;;CALL=TYPE       TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)   017170  016652                           $TYPOC  ;;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)   017172  016626                           $TYPOS  ;;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)   017174  016666                           $TYPON  ;;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)   017176  017054                           $TYPBN  ;;CALL=TYPBN      TRAP+5(104405)  TYPE BINARY (ASCII) NUMBER
  (1)
  (1)
  (3)   017200  014666                           $RDCHR  ;;CALL=RDCHR      TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
  (3)   017202  015006                           $RDLIN  ;;CALL=RDLIN      TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
  (3)   017204  015160                           $RDOCT  ;;CALL=RDOCT      TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
 1478   017206  004136                           TEST    ;;CALL=CHECK      TRAP+11(104411)
 1479   017210  004130                           TESTIT  ;;CALL=CHKIT      TRAP+12(104412)
 1480   017212  011502                           DECTYP  ;;CALL=TYPDC      TRAP+13(104413)
 1481                                    .EVEN
 1482   017214  000310                   DIST:   .BLKW   200.              ;STATE-WIDTH DISTRIBUTION
 1483   020034  010000                   BUFFER: .BLKW   4096.             ;BUFFER AREA
 1484           000001                   .END
```

```
ABASE = 170400         24#      38      77      78      79
ACDW1 = 000000         38
ACDW2 = 000000         38
ACPUOP= 000000         38
ADBUFF  001322         79#     312     322     358     374     384     395     421     447     461     514*    518*    550
                      552      553     555     734     789*    803*    805*    844*    870*    883     916*    922     928
                      991     1245
ADDW0 = 000000         38
ADDW1 = 000000         38
ADDW10= 000000         38
ADDW11= 000000         38
ADDW12= 000000         38
ADDW13= 000000         38
ADDW14= 000000         38
ADDW15= 000000         38
ADDW2 = 000000         38
ADDW3 = 000000         38
ADDW4 = 000000         38
ADDW5 = 000000         38
ADDW6 = 000000         38
ADDW7 = 000000         38
ADDW8 = 000000         38
ADDW9 = 000000         38
ADEVCT= 000000         38
ADEVM = 000000         38
ADST1   001320         78#     788*    802*    804*    918*
AENV  = 000000         38
AENVM = 000000         38
AFATAL= 000000         38
AGAIN   007504        974#    1011
AGATST  011674       1362#    1366
AGTST   011704        755*     771*    780*   1364#
AJGAIN  005544        673      690#
AJOFF   005472        671      677#
AMADR1= 000000         38
AMADR2= 000000         38
AMADR3= 000000         38
AMADR4= 000000         38
AMAMS1= 000000         38
AMAMS2= 000000         38
AMAMS3= 000000         38
AMAMS4= 000000         38
AMSG    012107        138     1373#
AMSGAD= 000000         38
AMSGLG= 000000         38
AMSGTY= 000000         38
AMTYP1= 000000         38
AMTYP2= 000000         38
AMTYP3= 000000         38
AMTYP4= 000000         38
APASS = 000000         38
APRIOR= 000200         26#      38      81
APTCSU= 000040       1471     1472#
APTENV= 000001       1468     1471    1472#
APTSIZ= 000200        164     1472#
APTSPO= 000100       1471     1472#
```

```
AROUND   007640        1010#
ASKCH    013743        1426#
ASWREG= 000000          38
ATESTN= 000000          38
ATMSG    012234         841    1383#
AUNIT  = 000000          38
AUSWR  = 000000          38
AVECT1= 100400          25#     38      80      82      83      84
AVECT2= 000000          38
BASEBR   001326          81#    808*    809*
BASECH   001342          87#    527     528     843     968    1233
BEGIN    001644          31     160#   1364
BEGINA   006122         212     759#
BEGINC   005420         215     663#
BEGINL   002670         270#    752     761
BEGINP   005674         218     715#
BEGINW   006204         224     775#
BEGIN2   001652          33     162#
BEGL     006064         199     221     750#
BEG2     002262          32     135     195#    689     712
BITPNT   001422         111#    912*    915     935     936*
BIT0   = 000001          22#    319     407     436     440     450     451     452
BIT00  = 000001          22#
BIT01  = 000002          22#
BIT02  = 000004          22#
BIT03  = 000010          22#
BIT04  = 000020          22#
BIT05  = 000040          22#
BIT06  = 000100          22#
BIT07  = 000200          22#
BIT08  = 000400          22#   1467
BIT09  = 001000          22#   1467    1468
BIT1   = 000002          22#
BIT10  = 002000          22#   1468
BIT11  = 004000          22#   1467
BIT12  = 010000          22#    277     314     323     324    1193
BIT13  = 020000          22#    723     735    1468
BIT14  = 040000          22#    281     425     427    1467
BIT15  = 100000          22#    329     344     355     425     427     439     449    1360
BIT2   = 000004          22#    300
BIT3   = 000010          22#    305     319
BIT4   = 000020          22#    296
BIT5   = 000040          22#    291
BIT6   = 000100          22#    286     407     411     729
BIT7   = 000200          22#    352     411     439
BIT8   = 000400          22#    273
BIT9   = 001000          22#    350     441
BPTVEC= 000014          22#
BUFFER   020034         951     996*    998*   1013    1041    1107    1119    1121    1141    1153    1154    1155    1177
                       1180    1483#
BUFF1    014064        1090    1434#
BUFF2    014112        1176    1435#
BUMPAD   006242         753     769     778     784#
BYPASS   006526         796     819#
B10      011432        1304#   1305
B11      011466        1314#   1315
```

```
CCHAN   012314        663     1389#
CH      012167        725     1379#
CHAN    012251       1220     1385#
CHANL   001372         99#     379*     390*     527*     571*     572      590*     591      593*     594      601*     665*     666
                      871      923     1221     1234*    1459
CHANNL  007446        967#
CHECK = 104411        347      356      366      375      444     1478#
CHKIT = 104412        274      282      287      292      297      301      306      330     1479#
CH1     001360         94#     599*     611      613      614*     840      842      860
CH2     001362         95#     600*     601      614      615*     836      856
CLEAR1  007374        953#     955
CLEAR2  007440        964#     965
CMSG    012114        132     1374#
COMPAR  011206        482      489      497      505      522      532      559      564      656     1255#
CONV    007036        881#     885
CONVR   007544        986      988#
CONVRT  011074        480      487      495      503      515      519      845     1232#
CR    = 000015         22#    1471
CRLF  = 000200         22#    1471
CRWR    012514        681      691      696     1395#
C0      014002        171     1427#
C2      014005       1099     1182     1428#
C3      014012       1179     1429#
DAC     001414        108#     578      581      584      587      627      631      701      705      859      863      913*     915*
                      916      935*
DASH    012143       1044     1377#
DDISP = 177570         22#      38      164
DECPNT  014620       1050*    1079*    1453#
DECTYP  011502       1321#    1480
DELAY   001416        109#     647*     651*     972*    1010*
DELAY1  007616        992     1003#
DELAY2  007624        994     1004     1005#
DELAY3  007542        987#
DELAY4  007336        799      942#     943
DELCLR  010704       1088     1100     1183     1193#
DF1     014664         55       62       68       74     1461#
DH1     014445         53     1449#
DH2     014501         72     1450#
DH3     014564         60       66     1451#
DIFLIN  007344        621      947#
DISPLA  001142         38#     164*     737*    1467*    1468*
DISPRE  000174         31#     164
DIST    017214        956     1024*    1091     1482#
DONE    012266       1283     1386#
DSWR  = 177570         22#      38      164
DT1     014624         54     1458#
DT2     014636         73     1459#
DT3     014654         61       67     1460#
DUMC    011326        413      420      429      432     1279#
DUMMY   001370         98#     528*     572*     666*     856*     860*     918
DUMW    011254        401      423     1269#
EDGE    001420        110#     624*     628*     698*     702*     878*     883*     886*     887*     888*     889*     893      928
EDGFLG  007142        588*     609*     895      901#
EMTVEC= 000030         22#     164*
EM1     014325         52     1444#
EM2     014347         59     1445#
```

```
EM3      014373            65     1446#
EM4      014420            71     1447#
ER       011066          1223     1225     1228#
ERMSG    012625           537      852     1056     1071     1083     1136     1228     1399#
ERR      006714           849      852#
ERRVEC=  000004            22#     164*     250      251*     266*     798*    1467*
FIRST    001352            91#     962*    1036     1038*
FIXADR   006340           785      797#
FIXONE   006344           188      798#
FLAG     001410           106#     166*     183*    1086     1139
GETDAT   010462          1144#    1152
GETEDG   006764           573      602      870#
GMSG     012121           146     1375#
GNS    = ****** U          31     1366     1477     1478     1479     1480
GUNITS   001436           117#     240*     255*    1366     1468*
HAFMSG   013015          1080     1405#
HALF     010146          1072     1074#
HEAD1    014145           186     1438#
HEAD5    013723           717     1424#
HT     = 000011            22#    1471
HUNS     014617          1330*    1342*    1344*    1347     1452#
IGND     012455           677     1394#
INRNGE   007704          1020     1023#
IOTVEC=  000020            22#     164*
ISERV    001464           127#     190
IVOLT    012543           690     1396#
KBVECT   001344            88#     189      821
LAST     010020          1034     1048#
LEND     005212           620      622#
LESS     011420          1299     1301#
LF     = 000012            22#    1471
LINEA    013702          1129     1423#
LOAD     010662          1178     1181     1185#
LOADY    011400          1094     1096     1188     1295#
LOAD0    010666          1186#    1190
LO2      010660          1140     1184#
LSB      012177           835     1381#
LSBMSG   012134          1046     1376#
MAT      014035           837     1432#
MAX      001430           114#    1160*    1165     1167*    1171
MAXTST   010560          1163     1165#
MEND     013115           762     1408#
MESP     013077          1218     1407#
MESR     013064          1215     1406#
METST    014042          1273     1433#
MIN      001424           112#    1159*    1162     1164*    1170
MINUS    012103           898     1323     1371#
MLSB     014027           639     1431#
MOFSET   014014           530     1430#
MSG16    013272          1089     1417#
MSG18    013200          1175     1411#
MSG20    013240           947     1415#
MSG21    013655          1106     1422#
MSG50    013157           263     1410#
MSG71    013617           202     1421#
MOLSB    012431           695     1393#
```

```
NAR      010064         1057    1059#
NARMSG   012664         1061    1401#
NARROW   001350           90#    961*   1028*   1059    1074
NBEXT    001364           96#    252*    254*    262     264*    265     784     795*    817*
NEXT     007510          975#   1009
NMBEXT   001366           97#    265*    817
NOI      012241         1206    1384#
NOIMSG   012042          570    1369#
NONE     001636          143     145     156#
NOTNAR   007740         1C27    1033#
NOTNEW   010354         1116    1121#
NOTOK    007610         1001#   1007
NOVT55   002212          174     177     182     186#    234
NXTCMP   010544         1161#   1169
NXTCVT   007224          918#    932
NXTSTA   010316         1108#   1122
NXTY1    010252         1093#   1098
NXT8     010526         1157#   1174
OFFERR   004520          535     537#
OFFOK    004526          536     539#
OFFSET   005214          529     624#    683
OKAY     007634          997     999    1002    1008#
OKAYD    011526         1326    1328#
OKMSG    012303          539     850    1058    1073    1085    1138    1226    1388#
ONAD     013141         1275    1409#
ONES     014622         1328*   1334*   1335    1337*   1346*   1455#
OUT      001434          116#    959*   1021*   1063    1066    1069    1076
OUTMSG   012762         1068    1403#
PEAK     001406          105#    575*    584*    587*   1210    1212*   1216    1224
PERCNT   001432          115#    907*    908*    909*    910*    911*    933
PIRQ   = 177772           22#
PIRQVE= 000240            22#
PLUS     010336         1113    1115#
PLUSR2   011406         1296    1298#
POPRO    001616          151#    157
POS      011514         1322    1325#
POSITV   012623          637    1398#
POSPEA   010774         1211    1213#
POSRMS   010762         1208    1210#
PRO    = 000000           22#
PR1    = 000040           22#
PR2    = 000100           22#
PR3    = 000140           22#
PR4    = 000200           22#
PR5    = 000240           22#
PR6    = 000300           22#
PR7    = 000340           22#
PS     = 177776           22#
PSW    = 177776           22#
PWRVEC= 000024            22#
QUEST    012105          156     225    1372#
RBEG     001660          161     163#
RDCHR  = 104406         1463    1477#
RDLIN  = 104407          203     668     678     692    1465    1477#
RDOCT  = 104410          149     664    1477#
READ     007656         1014#   1049
```

```
RELACC  010302          1087    1104#
REST1   002216           168     187#
RESVEC= 000010            22#    799*
RET     007140           896     900#
RETERR  004160           471     473#
RETURN  001620           152#    727     880     919    1241
RET1    001630           154#    973
RET2    001634           153     155#
RMS     001404           104#    574*    578*    581*   1207    1209*   1213    1222
RNA     001376           101#    192*    948
RNB     001400           102#    193*    949
RNC     001402           103#    194*    950
RST     011356           134     140    1287#
SARSUB  007144           576     579     582     585     625     629     699     703     857     861     907#
SAR1    007170           912#
SEL     012342           667    1390#
SETCH   012212           839    1382#
SETMSG  012061           598    1370#
SET1A   006722           604     605     856#
SHIFT   007326           934     936#
SKIPST  001354            92#    963*   1031*   1051    1054
SKPMSG  012642          1053    1400#
SLASH   012301           148    1131    1387#
SPACE   012172           646     739    1380#
SPREAD  001412           107#    380*    391*   1256*   1263    1459
STACK = 001100            22#    133     139     164     195
STATE   012147          1039    1378#
STKLMT= 177774            22#
STREG   001316            77#    309*    310     319*    320     334*    337     344*    351*    355*    362*    363     371*
                         372     381*    382     392*    393     407*    408     410     425*    426     433*    436*    437
                         440*    450*    451*    452*    453     459     462*    468*    469     545*    547*    548     716*
                         730*    732*    763     787*    801*    874*    881*    920*    926*    971*    988*   1237*   1243*
                        1276    1458    1459    1460
SWDIST  010220          1084    1086#
SWR     001140            38#    144     147     150*    164*    721     723     735    1193    1467    1468
SWREG   000176            31#    164
SW0   = 000001            22#
SW00  = 000001            22#
SW01  = 000002            22#
SW02  = 000004            22#
SW03  = 000010            22#
SW04  = 000020            22#
SW05  = 000040            22#
SW06  = 000100            22#
SW07  = 000200            22#
SW08  = 000400            22#
SW09  = 001000            22#
SW1   = 000002            22#
SW10  = 002000            22#
SW11  = 004000            22#
SW12  = 010000            22#
SW13  = 020000            22#
SW14  = 040000            22#
SW15  = 100000            22#
SW2   = 000004            22#
SW3   = 000010            22#
```

```
SW4   = 000020          22#
SW5   = 000040          22#
SW6   = 000100          22#
SW7   = 000200          22#
SW8   = 000400          22#
SW9   = 001000          22#
TADDR   001374         100#     141      715*     750*     759*     775*
TBITVE= 000014          22#
TCHK    005312         641#     685      708
TEMP    001356          93#     242*     245*     248*     257*     517      521*     558*     563*     627*     631*     632*     635
                       701*     705*     706*     847     1001*    1006*    1236*    1245*    1248*    1249*    1250*    1251*    1257
TENS    014621        1329*    1338*    1339     1341*    1345*    1454#
TEST    004136         469#    1478
TESTAD  002464         239#     751      760      776
TESTIT  004130         468#    1479
TESTR2  011542        1331#    1336     1340     1343
TKVEC = 000060          22#
TOFF    005270         531      635#     684      707
TPVEC = 000064          22#
TRAPVE= 000034          22#     164*
TRTVEC= 000014          22#
TRY     007202         914#     937
TRYAG   002314         203#     226
TSTBIT  001440         118#     241*     255      256*     267*     786*     800*    1468
TST1    002670         271#     272
TST10   003050         308#
TST11   003112         315      318#
TST12   003156         325      328#
TST13   003172         332#
TST14   003236         338      342#
TST15   003270         349#
TST16   003332         360#
TST1/   003374         370#
TST2    002732         280#
TST20   003420         377#     430
TST21   003464         385      388#
TST22   003542         397      400#
TST23   003670         414      422#
TST24   003754         435#
TST25   004026         448#
TST26   004162         477#     478      479
TST27   004232         486#
TST3    002746         284#
TST30   004262         494#
TST31   004312         501#
TST32   004364         513#
TST33   004444         526#
TST34   004532         538      541#
TST35   004660         569#
TST36   005044         597#
TST37   005170         612      618#
TST4    002770         290#
TST5    003004         295#
TST6    003020         299#
TST7    003034         304#
TST8    010572        1166     1168#
```

```
TYPBAD  007752           1022    1032    1036#
TYPBN = 104405           1366    1477#
TYPDC = 104413            638     834    1045    1052    1060    1064    1067    1078    1128    1214    1217    1480#
TYPE  = 104401            132     138     146     148     156     171     186     202     225     263     530     537     539
                          570     598     637     639     644     646     663     667     674     677     680     681     690
                          691     694     695     696     717     725     739     746     762     767     835     837     839
                          841     850     852     898     947    1039    1044    1046    1053    1056    1058    1061    1065
                         1068    1071    1073    1080    1083    1085    1089    1090    1099    1106    1129    1131    1136
                         1138    1175    1176    1179    1182    1203    1206    1215    1218    1220    1226    1228    1273
                         1275    1283    1323    1347    1366    1463    1468    1469    1471    1474    1475    1477#
TYPEDG  007100            838     893#   1219
TYPOC = 104402           1469    1477#
TYPON = 104404           1477#
TYPOS = 104403            147     262     726     740     764     836     840     847     894     899    1043    1130    1133
                         1221    1274    1276    1477#
TYPOUT  011616           1332    1344#
TYPRP   010744            589    1206#
TYPSET  006572            610     834#
UNEXP   001442            120#    285     416
VADR    001336             85#    259     787     788     789
VECTOR  001324             80#    285*    405*    416*    727*    790*    806*    807*    810     812     814     880*    919*
                          973*   1241*
VECTR1  001330             82#    406*    791*    794*    810*    811*    816*
VECTR2  001332             83#    424*    792*    812*    813*
VECTR3  001334             84#    793*    814*    815*
VLIN    011670           1134    1359#
VNP     011664           1224    1357#
VNR     011662           1222    1356#
VSET    011666            848    1358#
VTFLG   002432            172     175     178     228#
VTINIT  014132           1203    1436#
VT55    002206            180     183#
VVCT    001340             86#    790     791     792     793
V0      011646            561     566    1349#
V12     011654            484     507    1352#
V2      011650            524    1350#
V326    011660            491     499    1354#
V4      011652            658    1351#
V50D    011656            534    1353#
WFTEST  001426            113#    160*    162*    200     246
WIDE    001346             89#    960*   1035*   1062    1075
WIDMSG  012723           1065    1402#
WRAP    004162            476#    768     777
XADJ    012415            680    1392#
YADJ    012607            694    1397#
$APTHD  001000             37#
$ASTAT= ****** U         1472
$ATYC   016404           1472#
$ATY1   016360           1472#
$ATY3   016366           1471    1472#
$ATY4   016376           1468    1472#
$AUTOB  001134             38#
$BASE   001250             38#    249     801     802     803
$BDADR  001122             38#
$BDDAT  001126             38#    249*    253     259*    312*    314     322*    324     337*    384*    395*    396     410*
                          426*    453*    454*    455     469*    470    1257*   1258    1458    1459    1460
```

```
$BELL   001164          38#    1468
$BIN    017126        1475#*
$CDW1   001254          38#
$CHARC  016354        1471#*
$CKSWR= ****** U      1477
$CMTAG  001100          38#     164
$CM3  = 000000          38#
$CNTLG  015131        1463#
$CNTLU  015124        1463#
$CPUOP  001222          38#
$CRLF   001171          38#     644     746     767    1463    1468    1469    1471
$DEVCT  001204          38#
$DEVM   001252          38#
$DOAGN  012032        1366#
$ENDAD  012022          35     1366#   1468
$ENDCT  011742         164     1366#
$ENULL  012036        1366#
$ENV    001214          38#    1468    1471    1472
$ENVM   001215          38#     164     243    1471    1472
$EOP    011706         756      772     781    1366#
$EOPCT  011734         164*    1366#
$ERFLG  001103          38#     121*   1467*   1468*
$ERMAX  001115          38#     164*   1467*
$ERROR  015540         164     1468#
$ERRPC  001116          38#    1458    1459    1460    1468*   1469
$ERRTB  001256          38#    1469
$ERRTY  015742        1468     1469#
$ERTTL  001112          38#    1468*
$ESCAP  001162          38#     120*    123*    164*   1467*   1468
$ETABL  001214          38#
$ETEND  001256          37       38#
$FATAL  001176          38#    1472*
$FFLG   016624        1472#*
$FILLC  001156          38#    1471
$FILLS  001155          38#    1471
$GDADR  001120          38#
$GDDAT  001124          38#     273*    276*    277     281*    286*    291*    296*    300*    305*    313*    323*    329*
                       333*     352*    361*    378*    389*    396     411*    427*    439*    449*    455     468     470
                      1255*    1259    1458    1459
$GET42  012012        1366#
$GTSWR= ****** U      1477
$HD   = 000000          21
$HIBTS  001000          37#
$HIOCT  015260        1465#*
$ICNT   001104          38#    1467*
$INTAG  001135          38#
$ITEMB  001114          38#    1468*   1469
$LF     001172          38#    1463    1468    1471
$LFLG   016623        1472#*
$LPADR  001106          38#     164*    271*    479*   1272*   1282*   1467*
$LPERR  001110          38#     164*    272*    478*   1271*   1281*   1467*   1468
$MADR1  001226          38#
$MADR2  001232          38#
$MADR3  001236          38#
$MADR4  001242          38#
$MAIL   001174          37       38#    164    1467    1468    1471
```

```
$MAMS1  001224               38#
$MAMS2  001230               38#
$MAMS3  001234               38#
$MAMS4  001240               38#
$MBADR  001002               37#
$MFLG   016622             1472#*
$MNEW   015147             1463#
$MSGAD  001210               38#   1472*
$MSGLG  001212               38#   1472*
$MSGTY  001174               38#   1472*
$MSWR   015136             1463#
$MTYP1  001225               38#
$MTYP2  001231               38#
$MTYP3  001235               38#
$MTYP4  001241               38#
$MXCNT  015536             1467#
$NULL   001154               38#   1471
$NWTST= 000001              271#    280#    284#    290#    295#    299#    304#    308#    318#    328#    332#    342#    349#
                            360#    370#    377#    388#    400#    422#    435#    448#    477#    486#    494#    501#    513#
                            526#    541#    569#    597#    618#
$OCNT   017050             1474#*
$OMODE  017052             1474#*
$OVER   015522             1467#
$PASS   001202               38#    164*    239*    619    1269    1279    1366*   1467
$PASTM  001006               37#
$QUES   001170               38#    674    1463    1468    1471
$RDCHR  014666             1463#   1477
$RDDEC= ****** U           1477
$RDLIN  015006             1463#   1477
$RDOCT  015160             1465#   1477
$RDSZ = 000010             1463#
$RTNAD  012034             1366#
$R2A  = ****** U           1477
$SAVRE= ****** U           1477
$SCOPE  015262              164    1467#
$SETUP= 000027               39#    164    1366    1463    1467    1468
$STUP = 177777               39#
$SVLAD  015466             1467#
$SVPC = 000214               35#
$SWR  = 167400                6#     21      23      38     164     271     280     284     290     295     299     304     308
                            318     328     332     342     349     360     370     377     388     400     422     435     448
                            477     486     494     501     513     526     541     569     597     618    1366    1467    1468
$SWREG  001216               38#    164
$SWRMK= 000000               23    1467
$TESTN  001200               38#   1467*
$TIMES  001160               38#    164*    332*    343*    360*    477*    486*    494*    501*    513*    526*    542*    569*
                            597*    618*   1366*   1467*
$TKB    001146               38#    128     235     654    1463
$TKS    001144               38#    170*    204*    229     336*    346*    368*    648*    649     655*   1288*   1362*   1463
$TN   = 000040                7#     21     271#    280#    284#    290#    295#    299#    304#    308#    315     318#    325
                            328#    332#    338     342#    349#    360#    370#    377#    385     388#    397     400#    401
                            414     422#    423     435#    448#    477#    486#    494#    501#    513#    526#    538     541#
                            569#    597#    612     618#
$TPB    001152               38#   1306*   1316*   1471*
$TPFLG  001157               38#   1471
$TPS    001150               38#   1304    1314    1471
```

```
$TRAP    017130           164     1477#
$TRAP2   017152          1477#
$TRP   = 000014          1477#    1478#    1479#    1480#
$TRPAD   017164          1477#
$TSTM    001004            37#
$TSTNM   001102            38#     477*    1366*    1467*    1468
$TTYIN   015114          1463#
$TYPBN   017054          1475#    1477
$TYPDS=  ****** U        1477
$TYPE    016076          1471#    1472     1477
$TYPEC   016310          1471#
$TYPEX   016356          1471#
$TYPOC   016652          1474#    1477
$TYPON   016666          1474#    1477
$TYPOS   016626          1474#    1477
$UNIT    001206            38#
$UNITM   001010            37#
$USWR    001220            38#
$VECT1   001244            38#     806      808
$VECT2   001246            38#
$XTSTR   015272          1467#
$$GET4=  000000          1366#
$OFILL   017051          1474#*
$40CAT=  ****** U        1467     1468
.      = 040034            28#      31#      35#      36#      37#      38#     164     1366#    1463#    1467     1468     1469#    1471
                         1472#    1482#    1483#
.$ASTA=  ****** U        1472
.$X    = 001000            37#
```

```
BERROR      17#    1468
COMMEN      22#
DUMWRN      43#     401     423
ENDCOM      22#
ENDPAS      12#    1366
ERROR       22#     122     275     283     288     293     298     302     307     316     326     331     339     348     357
           367      376     386     398     412     428     445     457     485     492     500     508     525     562     567
ESCAPE      22#     120
GETPRI      22#
GETSWR      22#
MULT        22#
NEWTST      22#     271     280     284     290     295     299     304     308     318     328     332     342     349     360
           370      377     388     400     422     435     448     477     486     494     501     513     526     541     569
           597      618
POP         22#    1465    1472
PUSH        22#    1465    1472
REPORT      22#
SCOPE       22#     280     284     290     295     299     304     308     318     328     332     342     349     360     370
           377      388     400     422     435     448     463     486     494     501     513     526     541     569     597
           618
SETPRI      22#
SETTRA    1477#    1478    1479    1480
SETUP       22#     164
SKIP        22#     201     211     214     217     220     223     230     232     234     244     247     258     260     278
           311      315     321     325     338     385     397     414     456     471     511     538     592     595     612
           616      636     642     645     650     652     653     659     671     673     675     686     687     709     710
           724     1194    1196
SLASH       22#
SPACE       22#
STARS       22#      35      37      38     271     280     284     290     295     299     304     308     318     328     332
           342      349     360     370     377     388     400     422     435     448     477     486     494     501     513
           526      541     569     597     618    1366    1463    1465    1467    1468    1469    1471    1472    1474    1475
          1477
SWRSU       22#     164#
TRMTRP    1477#
TYPBIN      22#    1366
TYPDEC      22#
TYPNAM      22#
TYPNUM      22#
TYPOCS      22#     147     262     726     740     836     840     847     894     899    1043    1130    1133    1221    1274
          1276
TYPOCT      22#    1469
TYPTXT      22#    1366
$$CMRE      38#
$$CMTM      38#
$$ESCA      22#
$$NEWT      22#     271     280     284     290     295     299     304     308     318     328     332     342     349     360
           370      377     388     400     422     435     448     477     486     494     501     513     526     541     569
           597      618
$$SET     1477#    1478    1479    1480
$$SETM     164#
$$SKIP      22#     315     325     338     385     397     414     538     612
.EQUAT       8#      22
.HEADE       8#      21
.SETUP      10#      39
.SWRHI      10#      23
```

```
.SWRLO     23#
.$ACT1     11#      35
.$APTB     11#      38#
.$APTH     11#      37
.$APTY     11#    1472
.$CATC      8#      31
.$CMTA      8#      38
.$EOP       8#    1366
.$ERRO      8#    1468
.$ERRT     10#    1469
.$PARM      9#
.$POWE      9#
.$RAND     11#
.$RDOC     11#    1465
.$READ      9#    1463
.$SAVE      9#
.$SCOP      9#    1467
.$SPAC     10#
.$SWDO     10#
.$TRAP     10#    1477
.$TYPB      9#    1475
.$TYPD     11#
.$TYPE     10#    1471
.$TYPO      9#    1474


. ABS.  040034      000      OVR    RO     ABS    LCL    I


 ERRORS DETECTED:  0

 CVADAC,CVADAC/CRF=CVADAC
 RUN-TIME: 20 9 1 SECONDS
 RUN-TIME RATIO: 99/31=3.1
 CORE USED:  26K  (51 PAGES)
```