MINC-11

MNCAD DIAGNOSTIC
CVMNAA0

AH-B086A-MC

COPYRIGHT © 1978

FICHE 1 OF 1

DEC 1978

digital

MADE IN USA

IDENTIFICATION

Product Code: AC-B085A-MC

Product Name: CVMNAA0 - MNCAD Performance Test

Date: August 1978

Maintainer: Diagnostic Group

1.0 ABSTRACT

This diagnostic has three starting addresses:

    200 Normal start
    204 Restart
    210 Starting address with tester

This diagnostic tests the MNCAD/MNCAM with or without the optional test module(s).

When starting the diagnostic, a set of tests are listed. The operator selects the test by the 'TEST CHARACTER" and then depresses the 'RETURN" key on the console. The following chart indicates which letter corresponds to which test loop or function to execute:

W: The entire wraparound test

        Analog subtests
        Noise test
        Interchannel Settling test
        Differential Linearity and Relative Accuracy test

C: Calibration loop for the mncad

P: Print converted analog values loop only

L: Logic subtests

A: Auto test

        Logic subtests
        Analog subtests
        Noise test
        Interchannel Settling test
        Differential Linearity and Relative Accuracy test

N: Noise tests on selected channels

V: Same as W for video bit map console terminal (I.E.   VT105, VT55)

B: Base or vector address change

G: Get new switch register value

H: Help the operator and re-type the test list

2.0 REQUIREMENTS

2.1 Equipment

     LSI-11 computer with 12K of memory
     I/O Terminal (LA36, VT100, etc.)
     MNCAD/MNCAM Module
     MNCAD-TA test module
     Bit map for graphic output (I.E. VT105, VT55) <optional>

2.2 Storage

     This program uses 12K of memory.

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   100000   Halt on error
     SW14=1   040000   Loop on test
     SW13=1   020000   Inhibit error typeouts
     SW12=1   010000   Inhibit sizing the number of MNCAD (A/D)'S
     SW11=1   004000   Inhibit iterations
     SW10=1   002000   Halt for video bit map display
     SW9 =1   001000   Loop on error
     SW8 =1   000400   Loop on test in SWR <7:0>

     200 is the starting address of the diagnostic for standard
     tolerances. 204 is the restart address. 210 is the starting
     address of the diagnostic when the tester is connected and
     tighter tolerances are used.

5.0 OPERATING PROCEDURE

Start the diagnostic at 200 or 210. The program heading, request for initial switch register value. The operator will normally depress the 'RETURN' key. The program now request if the MNCAD-TA test module is connected. The operator responds by typing a 'Y' or 'N' followed by depressing the 'RETURN' key. A list of tests, loops, or functions available will be printed out. The operator selects the character, according to the table listed, and depresses the 'RETURN' key.

A control character (^C) is set aside for interrupting a test and transferring control to the beginning of the diagnostic. During the logic tests, while a reset is being performed, control C will not be executed. Therefore, continue typing control C 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, depress 'CTRL' and 'G' together. 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 selected, the program will ask that the MNCAD (A/D)'S front panel switches be set to 'TEST'. If the test module is connected, the program will also ask that the test module(s) switches be set to single-ended. Set the switches and depress 'RETURN'. The program will then give a channel table for the MNCAD (A/D) under test. If the test module is connected, the program will then ask for channels to test. The channels under test must be contiguous. The program will run through the analog subtests, the noise test, the interchannel settling test, and the differential linearity and relative accuracy test.
If 'C' is typed, the program will run the calibration routine and loop on the test until it is calibrated and a 'RETURN' is typed. If a certain MNCAD (A/D) is to be calibrated, use the 'B' command to inform the program of its base and vector address.

If 'P' is typed, the program will run the print values routine and will loop on that test until the operator type 'CTRL C'. If a certain MNCAD is to be tested, use the 'B' command to inform the program of its base and vector address.

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 ask that the A/D switches
be set to "TEST". If the test module is connected, the program
will also ask that the test module(s) switches be set to
single-ended. Set switches and depress "RETURN". The program
will then give a channel table for the MNCAD (A/D) under test.
If the test module is connected, the program will then ask for
channels to test. The channels under test must be contiguous.
The program will run through the analog subtests, the noise test,
the interchannel settling test, and the differential linearity
and relative accuracy test.

If "L" is typed, the program will then size the number of MNCAD
(A/D)'S and report the number of units found. The program will
then execute the logic tests, printing "END PASS" when it has
completed an entire pass. If additional MNCAD (A/D)'S are
detected, the test will be run successively on each MNCAD. If
the test module is connected, the program will ask the operator
to depress the test module "EXTERNAL START" switch on the first
pass.

5.1 Inhibiting auto-size feature

Logic, auto and wraparound tests will automatically auto-size and report the number of MNCAD'S it detects on the system. To inhibit this feature, set switch register bit 12 to a one. Another way to inhibit this feature is to set bit 15 of location $ENVM (1214). Also, use the program 'B' command to modify the default base and vector addresses for other than the first MNCAD.

5.2 End of pass typeouts

At end of pass, the following typeout will occur:

'END PASS 12 ;TOTAL ERROR COUNT = 5 ;BAD UNITS 0000000000000100''

This indicates that:

    Twelve passes thru the program have been made.
    A total of 5 errors have been detected.
    Unit # 3 was the unit with errors.

6.0 ERRORS

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

UNIT:	Unit number
ERRPC:	Location at which an error was detected.
STREG:	Address of the status register.
ADBUFF:	Address of the buffer
CHANL:	Channel value
NOMINAL:	Experted 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; 22 minutes
                           for successive passes
Logic test: 30 seconds
Auto test: 8 minutes first pass, 23 minutes
                           for successive passes

7.2 Status register and vector addresses

When testing more than one MNCAD, the difference in addresses is
4 for bus address and 10 for vector address. These values are in
VADR (bus address) (1352) and VVCT (vector address) (1354).   The
first MNCAD'S status register address must be in $BASE (1244),
its vector address must be in the low byte of $VECT1 (1240). The
operator may use the 'B' program command to change the default
values.

7.3 Switch register

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

7.4 Bit map graphic output terminal available

The operator may inform the program that the console is a bit map
terminal (I.E.   VT105 or VT55) by using the 'V' command. the
program will then display the results of the differential
linearity test on the bit map terminal screen.

8.0 RESTRICTIONS

8.1 Testing

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 MNCAD
responds to the expected address. If the MNCAD does not respond,
a buss error occurs. Also bus errors can occur during the time
the program sizes to see how many MNCAD'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 an 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 24 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 depressing ''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 re-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 console terminal; if SWR bit 13 is up (1), it puts the converted value in the ''DISPLAY REGISTER''.   The operator may change the channel (using the switch register) 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 determines the width of each state to within 0.01 LSB.

9.5   Settling test

The purpose of this test is to verify that the time allowed for
settling to a new input value after switching channels does not
result in an error that exceeds the expected amount for such a
change.

9.6   Noise test

This test measures the short-term MINC-11 system noise.   RMS
noise equals 1 standard deviation of the Gaussian curve, PEAK
noise equals 3 standard deviation of the Gaussian curve.

9.7   Analog tests

These 6 subtests check the channels and their output.

```
        20                              .TITLE   CVMNA-A MNCAD/MNCAM        DIAGNOSTIC
        (1)                             ;*COPYRIGHT (C) 1978
        (1)                             ;*DIGITAL EQUIPMENT CORP.
        (1)                             ;*MAYNARD, MASS. 01754
        (1)                             ;*
        (1)                             ;*PROGRAM BY GEORGE STEVENS
        (1)                             ;*
        (1)                             ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
        (1)                             ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
        (1)                             ;*
        21                              .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             R2=     %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
 22                                     .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            INHIBIT SIZING # OF MNCAD'S
(1)                                     ;*        11            INHIBIT ITERATIONS
(1)                                     ;*        10            HALT FOR VIEWING BIT MAP TERMINAL DISPLAY
(1)                                     ;*         9            LOOP ON ERROR
(1)                                     ;*         8            LOOP ON TEST IN SWR<7:0>
 23             171000                  ABASE=  171000
 24             000400                  AVECT1= 400
 25
 26             000100                  .=100
 27  000100     000104  000200  000002  .WORD   104,200,2
 28
 29                                     .SBTTL  TRAP CATCHER
 30
 31             000000                  .=0
 32                                     ;*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ''.+2''
 33                                     ;*AND ''JSR PC,RO'' SEQUENCE TO CATCH ILLEGAL INTERRUPTS.
 34                                     ;*AND INTERRUPTS TO THE WRONG VECTOR.
 35                                     ;*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
 36                                     ;*VECTORS.
 46             000004                  .=4
 47  000004     016612  000200          .WORD   IOTRD,200       ;HANDLE BUSS ERROR.
 48             000174                  .=174
 49  000174     000000          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER.
 50  000176     000000          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER.
 51
 52  000200     000137  001556          JMP     BEGIN           ;START ADDRESS
 53  000204     000137  001620          JMP     @#RBEG2         ;RESTART ADDRESS
 54  000210     000137  001626          JMP     @#BEGIN2        ;START ADDRESS FOR OPTION TESTER
```

```
    56                                    .SBTTL   ACT11 HOOKS
   (1)
   (2)                        ;;********************************************************************
   (1)                        ;HOOKS REQUIRED BY ACT11
   (1)              000214            $SVPC=.                  ;SAVE PC
   (1)              000046            .=46
   (1)   000046     016500            $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            .=$SVPC                  ;; RESTORE PC
    57              001000     .=1000
    58                         .SBTTL   APT PARAMETER BLOCK
   (1)
   (2)                        ;;********************************************************************
   (1)                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
   (2)                        ;;********************************************************************
   (1)              001000            .$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     001170     $MBADR: .WORD    $MAIL       ;;ADDRESS OF APT MAILBOX (BITS 0-15)
   (1)   001004     002260     $TSTM:  .WORD    1200.       ;;RUN TIM OF LONGEST TEST
   (1)   001006     000764     $PASTM: .WORD    500.        ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
   (1)   001010     003244     $UNITM: .WORD    1700.       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
   (1)   001012     000031             .WORD    $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
   59                              .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                 ;;CONTAINS THE TEST NUMBER
  (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    0
  (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     077             $QUES:  .ASCII   /?/           ;;QUESTION MARK
  (1)  001165     015             $CRLF:  .ASCII   <15>          ;;CARRIAGE RETURN
  (1)  001166  000012             $LF:    .ASCIZ   <12>          ;;LINE FEED
  (2)                              ;;**************************************************************
  (2)                              .SBTTL   APT MAILBOX-ETABLE
  (2)
  (3)                              ;;**************************************************************
  (2)                              .EVEN
  (2)  001170                     $MAIL:                         ;;APT MAILBOX
  (2)  001170  000000             $MSGTY: .WORD    AMSGTY        ;;MESSAGE TYPE CODE
  (2)  001172  000000             $FATAL: .WORD    AFATAL        ;;FATAL ERROR NUMBER
  (2)  001174  000000             $TESTN: .WORD    ATESTN        ;;TEST NUMBER
  (2)  001176  000000             $PASS:  .WORD    APASS         ;;PASS COUNT
  (2)  001200  000000             $DEVCT: .WORD    ADEVCT        ;;DEVICE COUNT
  (2)  001202  000000             $UNIT:  .WORD    AUNIT         ;;I/O UNIT NUMBER
```

```
(2)   001204   000000          $MSGAD: .WORD    AMSGAD   ;;MESSAGE ADDRESS
(2)   001206   000000          $MSGLG: .WORD    AMSGLG   ;;MESSAGE LENGTH
(2)   001210                   $ETABLE:                  ;;APT ENVIRONMENT TABLE
(2)   001210      000          $ENV:   .BYTE    AENV     ;;ENVIRONMENT BYTE
(2)   001211      000          $ENVM:  .BYTE    AENVM    ;;ENVIRONMENT MODE BITS
(2)   001212   000000          $SWREG: .WORD    ASWREG   ;;APT SWITCH REGISTER
(2)   001214   000000          $USWR:  .WORD    AUSWR    ;;USER SWITCHES
(2)   001216   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)   001220      000          $MAMS1: .BYTE    AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
(2)   001221      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)   001222   000000          $MADR1: .WORD    AMADR1   ;;HIGH ADDRESS,BLK#1
(2)                            ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)   001224      000          $MAMS2: .BYTE    AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
(2)   001225      000          $MTYP2: .BYTE    AMTYP2   ;;MEM.TYPE,BLK#2
(2)   001226   000000          $MADR2: .WORD    AMADR2   ;;MEM.LAST ADDRESS,BLK#2
(2)   001230      000          $MAMS3: .BYTE    AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
(2)   001231      000          $MTYP3: .BYTE    AMTYP3   ;;MEM.TYPE,BLK#3
(2)   001232   000000          $MADR3: .WORD    AMADR3   ;;MEM.LAST ADDRESS,BLK#3
(2)   001234      000          $MAMS4: .BYTE    AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
(2)   001235      000          $MTYP4: .BYTE    AMTYP4   ;;MEM.TYPE,BLK#4
(2)   001236   000000          $MADR4: .WORD    AMADR4   ;;MEM.LAST ADDRESS,BLK#4
(2)   001240   000400          $VECT1: .WORD    AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)   001242   000000          $VECT2: .WORD    AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)   001244   171000          $BASE:  .WORD    ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)   001246   000000          $DEVM:  .WORD    ADEVM    ;;DEVICE MAP
(2)   001250   000000          $CDW1:  .WORD    ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
(2)   001252                   $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)   001252                      $ERRTB:
          61
          62
          63
          72                                ;ITEM    1
          73    001252  023733                      EM1                 ;MNCAD STATUS REG. ERROR
          74    001254  024312                      DH1                 ;ERRPC STREG EXPECTED ACTUAL
          75    001256  024552                      DT1                 ;$ERRPC, STREG, $GDDAT, $BDDAT
          76    001260  024642                      DF1
          77
          78
          79                                ;ITEM    2
          80    001262  023771                      EM2                 ;MNCAD FAILED TO INTERRUPT
          81    001264  024442                      DH3                 ;ERRPC STREG ACTUAL
          82    001266  024606                      DT3                 ;$ERRPC, STREG,  $BDDAT
          83    001270  024642                      DF1
          84
          85                                ;ITEM    3
          86    001272  024031                      EM3                 ;MNCAD UNEXPECTED INTERRUPT
          87    001274  024442                      DH3                 ;ERRPC STREG
          88    001276  024606                      DT3                 ;$ERRPC, STREG
          89    001300  024642                      DF1
          90
          91                                ;ITEM    4
          92    001302  024072                      EM4                 ;MNCAD ERROR ON A/D CHANNEL
          93    001304  024356                      DH2                 ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
          94    001306  024566                      DT2                 ;$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT
          95    001310  024642                      DF1
          96
          97                                ;ITEM    5
          98    001312  024133  024476  024620      EM5,DH5,DT5,DF1      ;EXISTING MNCAD NOW FAILS TO RESPOND
                001320  024642
          99
         100                                ;ITEM    6
         101    001322  024214  024522  024632      EM6,DH6,DT6,DF1      ;BUS ERROR ON SPECIFIED DEFAULT ADDRESS
                001330  024642
```

```
103
104   001332   171000        MNCAD0: ABASE                    ;ADDRESS OF MNCAD #0
105   001334   000400                AVECT1                   ;VECTOR OF MNCAD #0
106   001336   171004                ABASE+4                  ;              #1
107   001340   000410                AVECT1+10                ;              #1
108   001342   171010                ABASE+10                 ;              #2
109   001344   000460                AVECT1+60                ;              #2
110   001346   171014                ABASE+14                 ;              #3
111   001350   000470                AVECT1+70                ;              #3
112
113                                  .SBTTL        MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
114   001352   171000        STREG:  ABASE                    ;ADDRESS OF STATUS REGISTER
115   001354   171001        ADST1:  ABASE+1                  ;UPPER BYTE OF STATUS REG.
116   001356   171002        ADBUFF: ABASE+2                  ;ADDRESS OF A/D BUFFER
117   001360   000400        VECTOR: AVECT1                   ;VECTOR ADDRESS
118   001362   000402        VECTR1: AVECT1+2
119   001364   000404        VECTR2: AVECT1+4                 ;ERROR VECTOR ADDRESS
120   001366   000406        VECTR3: AVECT1+6
121   001370   000000        BASECH: 0                        ;BASE CHANNEL
122   001372   000000        BASEND: 0                        ;END CHANNEL
123   001374   000060        KBVECT: 60
124   001376   170400        GSTREG: 170400                   ;KNOWN GOOD A/D CSR
125   001400   170402        GADBUF: 170402                   ;KNOWN GOOD A/D DBR
126   001402   000410        GVECT:  410                      ;KNOWN GOOD A/D VECTOR
127   001404   170430        CLKCSR: 170430                   ;CLOCK CSR
128   001406   170432        CLKBPR: 170432                   ;CLOCK BPR
129   001410   167770        DRVCSR: 167770                   ;DRV11 CSR
130   001412   167772        DRVDOR: 167772                   ;DRV11 DOR
131   001414   167774        DRVDIR: 167774                   ;DRV11 DIR
132   001416   000000        WIDE:   0                        ;NO. OF WIDE STATES
133   001420   000000        NARROW: 0                        ;NO. OF NARROW STATES
134   001422   000000        FIRST:  0
135   001424   000000        SKIPST: 0                        ;NO. OF SKIPPED STATES
136   001426   000000        TEMP:   0                        ;WORK AREA
137   001430   000000        TEMP1:  0                        ;RESTART INDICATOR
138   001432   000000        CH1:    0                        ;FIRST CHANNEL
139   001434   000000        CH2:    0                        ;SECOND CHANNEL
140   001436   000000        NBEXT:  0                        ;NO. OF MNCAD'S TO BE TESTED
141   001440   000000        NMBEXT: 0                        ;NO. OF MNCAD'S TO BE TESTED
142   001442   000000        DUMMY:  0                        ;DUMMY CHANNEL
143   001444   000000        CHANL:  0                        ;CHANNEL VALUE
144   001446   000000        RNA:    0                        ;RANDOM
145   001450   000000        RNB:    0                        ;NUMBER
146   001452   000000        RNC:    0                        ;VALUES
147   001454   000000        RMS:    0                        ;RMS NOISE VALUE
148   001456   000000        PEAK:   0                        ;PEAK NOISE VALUE
149   001460   000000        FLAG:   0                        ;BIT MAP TERMINAL FLAG
150   001462   000000        SPREAD: 0                        ;DEVIATION FROM THE NOMINAL
151   001464   000000        DAC:    0                        ;SAR VALUE
152   001466   000000        DELAY:  0                        ;TIME DELAY COUNTER
153   001470   000000        EDGE:   0                        ;EDGE VALUE
154   001472   000000        BITPNT: 0
155   001474   000000        MIN:    0                        ;MIN VALUE
156   001476   000000        WFTEST: 0      ;0= NO DWARF ;BIT15 =1 TESTER; NON-ZERO = DWARF
```

```
157  001500  000000              MAX:    0                      ;MAX VALUE
158  001502  000000              PERCNT: 0                      ;PERCENT FOR SAR ROUTINE
159  001504  000000              OUT:    0
160  001506  000000              EVER:   0
161  001510  000000              BADUNT: 0                      ;BAD UNIT MAP
162  001512  000001              MASKNM: 1                      ;CURRENT UNIT MAP
163  001514  000000              UNITBD: 0
164
165  001516                      UNEXP:
(1)  001516  012737  001532  001162      MOV    #1$,$ESCAPE      ;;ESCAPE TO 1$ ON ERROR
166  001524  005237  001103            INC    $ERFLG
167  001530  104003                    ERROR  3
168  001532  005037  001162      1$:   CLR    $ESCAPE           ;RETURN ESCAPE TO NORMAL
169  001536  000002              RTI                            ;UNEXPECTED INTERRUPT
170  001540  022776  000001  000000  RETURN: CMP  #1,@0(SP)     ;DOES IT RETURN TO A WAIT?
171  001546  001002                    BNE    RET2              ;NO
172  001550  062716  000002      RET1: ADD    #2,(SP)           ;BUMP RETURN ADDRESS
173  001554  000002              RET2: RTI
1 .
175
176                              .SBTTL    INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
177  001556  005037  001476      BEGIN: CLR   WFTEST
178  001562  000424                    BR     RBEG
179  001564  012700  000200      BEG2:  MOV    #200,R0           ;LOAD STARTING ADDRESS POINTER
180  001570  012720  000137            MOV    #137,(R0)+         ;LOAD JUMP
181  001574  012720  001556            MOV    #BEGIN,(R0)+
182  001600  012720  000137            MOV    #137,(R0)+
183  001604  012720  001620            MOV    #RBEG2,(R0)+
184  001610  012720  000137            MOV    #137,(R0)+
185  001614  012720  001626            MOV    #BEGIN2,(R0)+
186  001620  005237  001430      RBEG2: INC   TEMP1              ;SET RESTART FLAG
187  001624  000405                    BR     RBEG1
188  001626  012737  100000  001476  BEGIN2: MOV  #BIT15,WFTEST  ;INDICATE TESTER IS CONNECTED
189  001634  005037  001430      RBEG:  CLR    TEMP1             ;CLEAR RESTAT FLAG
190  001640  000005              RBEG1: RESET
191                              .SBTTL  INITIALIZE THE COMMON TAGS
(1)                              ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1)  001642  012706  001100            MOV    #$CMTAG,R6         ;;FIRST LOCATION TO BE CLEARED
(1)  001646  005026                    CLR    (R6)+             ;;CLEAR MEMORY LOCATION
(1)  001650  022706  001140            CMP    #SWR,R6 ;;DONE?
(1)  001654  001374                    BNE    .-6               ;;LOOP BACK IF NO
(1)  001656  012706  001100            MOV    #STACK,SP          ;;SETUP THE STACK POINTER
(1)                              ;;INITIALIZE A FEW VECTORS
(1)  001662  012737  026304  000020    MOV    #$SCOPE,@#IOTVEC   ;;IOT VECTOR FOR SCOPE ROUTINE
(1)  001670  012737  000340  000022    MOV    #340,@#IOTVEC+2   ;;LEVEL 7
(1)  001676  012737  026626  000030    MOV    #$ERROR,@#EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
(1)  001704  012737  000340  000032    MOV    #340,@#EMTVEC+2  ;;LEVEL 7
(1)  001712  012737  030666  000034    MOV    #$TRAP,@#TRAPVEC  ;;TRAP VECTOR FOR TRAP CALLS
(1)  001720  012737  000340  000036    MOV    #340,@#TRAPVEC+2 ;;LEVEL 7
(1)  001726  012737  027172  000024    MOV    #$PWRDN,@#PWRVEC  ;;POWER FAILURE VECTOR
(1)  001734  012737  000340  000026    MOV    #340,@#PWRVEC+2  ;;LEVEL 7
(1)  001742  013737  016446  016440    MOV    $ENDCT,$EOPCT     ;;SETUP END-OF-PROGRAM COUNTER
(1)  001750  005037  001160            CLR    $TIMES            ;;INITIALIZE NUMBER OF ITERATIONS
(1)  001754  005037  001162            CLR    $ESCAPE           ;;CLEAR THE ESCAPE ON ERROR ADDRESS
```

```
 (1)   001760  112737  000001  001115         MOVB   #1,$SERMAX         ;;ALLOW ONE ERROR PER TEST
 (1)   001766  012737  001766  001106         MOV    #.,$LPADR          ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 (1)   001774  012737  00 774  001110         MOV    #.,$LPERR          ;;SETUP THE ERROR LOOP ADDRESS
 (2)                                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 (2)                                          ;;EQUAL TO A ''-1'', SETUP FOR A SOFTWARE SWITCH REGISTER.
 (2)   002002  013746  000004               MOV    @#ERRVEC,-(SP)     ;;SAVE ERROR VECTOR
 (2)   002006  012737  002042  000004         MOV    #64$,@#ERRVEC      ;;SET UP ERROR VECTOR
 (2)   002014  012737  177570  001140         MOV    #DSWR,SWR          ;;SETUP FOR A HARDWARE SWICH REGISTER
 (2)   002022  012737  177570  001142         MOV    #DDISP,DISPLAY     ;;AND A HARDWARE DISPLAY REGISTER
 (2)   002030  022777  177777  177102         CMP    #-1,@SWR           ;;TRY TO REFERENCE HARDWARE SWR
 (2)   002036  001012                         BNE    66$                ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 (2)                                                                    ;;AND   THE HARDWARE SWR IS NOT = -1
 (2)   002040  000403                         BR     65$                ;;BRANCH IF NO TIMEOUT
 (2)   002042  012716  002050          64$:   MOV    #65$,(SP)          ;;SET UP FOR TRAP RETURN
 (2)   002046  000002                         RTI
 (2)   002050  012737  000176  001140  65$:   MOV    #SWREG,SWR         ;;POINT TO SOFTWARE SWR
 (2)   002056  012737  000174  001142         MOV    #DISPREG,DISPLAY
 (2)   002064  012637  000004          66$:   MOV    (SP)+,@#ERRVEC     ;;RESTORE ERROR VECTOR
 (1)
 (2)   002070  005037  001176                 CLR    $PASS              ;;CLEAR PASS COUNT
 (2)   002074  132737  000200  001211         BITB   #APTSIZE,$ENVM     ;;TEST USER SIZE UNDER APT
 (2)   002102  001403                         BEQ    67$                ;;YES,USE NON-APT SWITCH
 (2)   002104  012737  001212  001140         MOV    #$SWREG,SWR        ;;NO,USE APT SWITCH REGISTER
 (2)   002112                          67$:
 192                                          ;ROUTINE TO OVERLAY THE ''$TYPE'' ROUTINE
 193   002112  012737  005046  027410         MOV    #5046,$TYPE        ;CLR -(SP)
 194   002120  012737  012746  027412         MOV    #12746,$TYPE+2     ;MOV #$TYPE+12,-(SP)
 195   002126  012737  027422  027414         MOV    #$TYPE+12,$TYPE+4
 196   002134  012737  000002  027416         MOV    #RTI,$TYPE+6       ;RTI
 197   002142  004737  024720                 JSR    PC,$TKINT          ;ENABLE TKB INTR.
 198   002146  005737  001430                 TST    TEMP1              ;TEST IF RESTART
 199   002152  001005                         BNE    20$                ;BR IF YES
 200   002154  005737  000042                 TST    @#42               ;TEST IF CHAIN MODE
 201   002160  001002                         BNE    20$                ;BR IF CHAIN MODE
 202   002162  104401  023075                 TYPE   .INITVT            ;INITILIZE THE TERMINAL
 203   002166                          20$:
 (1)                                          .SBTTL  TYPE PROGRAM NAME
 (1)                                          ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
 (1)   002166  005227  177777                 INC    #-1                ;;FIRST TIME?
 (1)   002172  001050                         BNE    68$                ;;BRANCH IF NO
 (1)   002174  022737  016500  000042         CMP    #$ENDAD,@#42       ;;ACT-11?
 (1)   002202  001444                         BEQ    68$                ;;BRANCH IF YES
 (1)   002204  104401  002252                 TYPE   .69$               ;;TYPE ASCIZ STRING
 (2)                                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 (2)   002210  005737  000042                 TST    @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
 (2)   002214  001012                         BNE    70$                ;;BRANCH IF YES
 (2)   002216  123727  001210  000001         CMPB   $ENV,#1            ;;ARE WE RUNNING UNDER APT?
 (2)   002224  001406                         BEQ    70$                ;;BRANCH IF YES
 (2)   002226  023727  001140  000176         CMP    SWR,#SWREG         ;;SOFTWARE SWITCH REG SELECTED?
 (2)   002234  001005                         BNE    71$                ;;BRANCH IF NO
 (2)   002236  104407                         GTSWR                     ;;GET SOFT-SWR SETTINGS
 (2)   002240  000403                         BR     71$
 (2)   002242  112737  000001  001134  70$:   MOVB   #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
 (2)   002250                          71$:
```

K 2

```
  (1)   002250  000421                          BR      68$               ;;GET OVER THE ASCIZ
  (1)                                  ;;69$:   .ASCIZ  <CRLF>#CVMNA-A  MNCAD (A/D) DIAGNOSTIC#<CRLF>
  (1)   002314                         68$:
  204   002314  013746  000010                  MOV     @#RESVEC,-(SP)    ;SAVE RESERVED VECTOR
  205   002320  012737  002360  000010           MOV     #1$,RESVEC        ;SET UP ILLEGAL INST. TRAP
  206   002326  012700  000001                  MOV     #1,R0             ;SET R0 TO ONE
  207   002332  077001                          SOB     R0,.              ;TRY SOB INSTRUCTION
  208   002334  012737  077001  013622          MOV     #77001,DELAY1     ;SET UP FOR SOB
  209   002342  012737  077001  013736          MOV     #77001,DELAY2     :
  210   002350  012737  077001  014052          MOV     #77001,DELAY3     :
  211   002356  000412                          BR      2$
  212   002360  022626                 1$:      CMP     (SP)+,(SP)+       ;POP TWO WORDS OFF STACK
  213   002362  012737  104420  013622          MOV     #DELY,DELAY1      ;INSTRUCTION FAILED
  214   002370  012737  104420  013736          MOV     #DELY,DELAY2      :
  215   002376  012737  104420  014052          MOV     #DELY,DELAY3      :
  216   002404  012637  000010         2$:      MOV     (SP)+,@#RESVEC    ;RESTORE ERROR VECTOR
  217   002410  004737  012510         3$:      JSR     PC,FIXONE         ;INITIALIZE ADDRESSES
  218   002414  012737  062341  001446          MOV     #62341,RNA        ;RANDOM NO, VARIABLES
  219   002422  012737  142315  001450          MOV     #142315,RNB       :
  220   002430  012737  127623  001452          MOV     #127623,RNC       :
  221   002436  004737  016266                  JSR     PC,WFADJ          ;SET UP TOLLERANCES
  222   002442  105737  001134                  TSTB    $AUTOB            ;TEST IF CHAIN/APT
  223   002446  001402                          BEQ     4$
  224   002450  000137  010776                  JMP     BEGL              ;GO TO LOGIC TESTS
  225   002454  005737  001430         4$:      TST     TEMP1             ;TEST IF RESTART
  226   002460  001026                          BNE     MTEST1
  227   002462  005737  001476                  TST     WFTEST            ;CHECH IF TESTER CONNECTED ?
  228   002466  100421                          BMI     MTEST0            ;BR IF YES
  229
  230                                           .SBTTL  OPERATOR INPUT DECODER
  231   002470  104401  001165         MTEST:   TYPE    ,$CRLF
  232   002474  104401  017745                  TYPE    ,YESNO            ;ASK FOR INPUT
  233   002500  104401  017267                  TYPE    ,DWRF             ;ABOUT DWARF MODULE
  234   002504  104412                          RDLIN
  235   002506  012600                          MOV     (SP)+,R0          ;GET INPUT
  236   002510  105037  001476                  CLRB    WFTEST            ;SET NO DWARF
  237   002514  042710  000040                  BIC     #40,(R0)          ;ENSURE UPPER CASE
  238   002520  122710  000131                  CMPB    #'Y,(R0)          ;TEST IF 1ST CHAR IS Y
  239   002524  001002                          BNE     MTEST0            ;BR IF NOT 'Y'
  240   002526  105237  001476                  INCB    WFTEST            ;SET DWARF CONNECTED FLAG
  241   002532  104401  023127         MTEST0:  TYPE    ,PRIME1           ;TELL THE OPERATOR THE STORY
  242   002536  000005         MTEST1:          RESET
  243   002540  052777  000100  176376          BIS     #BIT6,@$TKS       ;ENABLE TKB INTR.
  244   002546  005046                          CLR     -(SP)
  245   002550  012746  002556                  MOV     #1$,-(SP)
  246   002554  000002                          RTI                       ;LOWER PS
  247   002556  005037  001176         1$:      CLR     $PASS             ;INIT
  248   002562  005037  001112                  CLR     $ERTTL            :        THINGS
  249   002566  005037  001506                  CLR     EVER              :
  250   002572  104401  023643                  TYPE    ,DOT              ;TYPE THE 'DOT'
  251   002576  104412                          RDLIN
  252   002600  012600                          MOV     (SP)+,R0          ;READ ANSWER
  253   002602  142710  000040                  BICB    #40,(R0)
  254   002606  121027  000101                  CMPB    (R0),#'A          ;IS IT A?
```

```
255  002612  001002                          BNE     2$              ;;NO, TRY C
256  002614  000137  011040                  JMP     BEGINA          ;GO TO AUTO TEST
257  002620  121027  000103         2$:      CMPB    (R0),#'C        ;IS IT C?
258  002624  001002                          BNE     3$              ;;NO, TRY P
259  002626  000137  010122                  JMP     BEGINC          ;GO TO CALIBRATION LOOP
260  002632  121027  000120         3$:      CMPB    (R0),#'P        ;IS IT P?
261  002636  001002                          BNE     4$              ;;NO, TRY L
262  002640  000137  010614                  JMP     BEGINP          ;GO TO TYPE/DISPLAY CONVERSIONS TEST
263  002644  121027  000114         4$:      CMPB    (R0),#'L        ;IS IT L?
264  002650  001002                          BNE     5$              ;;NO, TRY M
265  002652  000137  010,76                  JMP     BEGL            ;GO TO LOGIC TESTS
266  002656  121027  000127         5$:      CMPB    (R0),#'W        ;IS IT W?
267  002662  001002                          BNE     6$              ;;NO,TRY AGAIN
268  002664  000137  011116                  JMP     BEGINW          ;GO TO WRAPAROUND TEST
269  002670  121027  000102         6$:      CMPB    (R0),#'B        ;IS IT B?
270  002674  001002                          BNE     7$              ;;NO TRY AGAIN
271  002676  000137  012306                  JMP     BASEXC          ;GO CHANGE BASE AND VECTOR ADDRESS
272  002702  121027  000110         7$:      CMPB    (R0),#'H        ;IF IT H?
273  002706  001670                          BEQ     MTEST           ;;YES, HELP THE OPERATOR
274  002710  121027  000107         10$:     CMPB    (R0),#'G        ;IS IT G
275  002714  001002                          BNE     11$             ;;NO, TRY AGAIN
276  002716  104407                          GTSWR
277  002722  000706                          BR      MTEST1
278  002722  121027  000126         11$:     CMPB    (R0),#'V        ;IS IT V?
279  002726  001004                          BNE     12$             ;;NO, TRY AGAIN
280  002730  005237  001460                  INC     FLAG            ;SET BIT MAP AVAILABLE FLAG
281  002734  000137  011116                  JMP     BEGINW          ;AND RUN WRAP TEST'S
282  002740  121027  000116         12$:     CMPB    (R0),#'N        ;IS IT N?
283  002744  001002                          BNE     13$             ;;NO, TRY AGAIN
284  002746  000137  011156                  JMP     BEGINN          ;RUN NOISE TESTS
285  002752  121027  000106         13$:     CMPB    (R0),#'F        ;IS IT F
286  002756  001002                          BNE     77$             ;;NO, TRY AGAIN
287  002760  000137  010376                  JMP     BEGINM          ;RUN SWITCH GAIN/PREAMP MANUAL TEST
288  002764  104401  017345         77$:     TYPE    ,QUEST
289  002770  000662                          BR      MTEST1          ;WAIT FOR CHARACTER
290                                           .SBTTL  DETERMINE THE NUMBER OF MNCAD'S ON THE SYSTEM
291  002772  013737  001244  001126 TESTAD:  MOV     $BASE,$BDDAT    ;GET BASE ADDRESS
292  003000  005037  001202                  CLR     $UNIT           ;CLR UNIT NUMBER
293  003004  012737  003060  000004          MOV     #2$,ERRVEC      ;LOAD RETURN ADDRESS
294  003012  005777  176110         1$:      TST     @$BDDAT         ;TEST IF ADDRESS EXISTS
295  003016  062737  000004  001126          ADD     #4,$BDDAT       ;UPDATE BUS ADDRESS
296  003024  005237  001202                  INC     $UNIT           ;UPDATE UNIT COUNT
297  003030  005737  001210                  TST     $ENV            ;TEST IF 'DO NOT SIZE''
298  003034  100424                          BMI     3$              ;BR IF NO SIZEING
299  003036  032777  010000  176074          BIT     #SW12,@SWR      ;TEST IF INHIBIT SIZING IS SET
300  003044  001020                          BNE     3$              ;BR IF SET
301  003046  022737  000004  001202          CMP     #4.,$UNIT       ;TEST IF MAX NUMBER
302  003054  001356                          BNE     1$              ;BR IF NOT
303  003056  000413                          BR      3$              ;BR IF MAX
304  003060  022626              2$:          CMP     (SP)+,(SP)+     ;RESTORE STACK
305  003062  005737  001202                  TST     $UNIT           ;TEST IF ANY EXIST
306  003066  001007                          BNE     3$              ;BR IF ANY ARE THERE
307  003070  005737  000042                  TST     @#42            ;TEST IF XXDP CHAIN MODE
308  003074  001004                          BNE     3$              ;BR IF YES
```

M 2
CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 3-5
CVMNAA.P11   DETERMINE THE NUMBER OF MNCAD'S ON THE SYSTEM

SEQ 0025

```
309  003076  104006                        ERROR   6              ;BASE ADDRESS CAUSED A BUS TRAP
310  003100  005726                        TST     (SP)+          ;POP 1 ARG.
311  003102  000137  016412                JMP     $EOP
312  003106  012737  016612  000004  3$:   MOV     #IOTRD,ERRVEC
313  003114  012737  000200  000006        MOV     #200,ERRVEC+2
314  003122  005737  001506                TST     EVER           ;TEST IF # HAS BEEN REPORTED
315  003126  100427                         BMI     4$             ;IF YES BRANCH
316  003130  005737  001476                TST     WFTEST         ;TEST IF IN TESTER MODE
317  003134  100415                         BMI     7$             ;BR IF TESTER
318  003136  104401  022074                TYPE    ,FOUND1        ;TELL OPERATOR # OF MNCAD'S FOUND
319  003142  013746  001202                MOV     $UNIT,-(SP)    ;PUT # TO BE TYPED ON STACK
320  003146  104405                        TYPDS
321  003150  104401  022117                TYPE    ,FOUND2        ;FINISH MESSAGE
322  003154  005737  001202                TST     $UNIT          ;TEST IF ANY UNITS
323  003160  001003                         BNE     7$             ;ANY UNIT
324  003162  005726                         TST     (SP)+          ;POP 1 ARG. OFF STACK
325  003164  000137  016412                JMP     $EOP           ;REPORT EOP
326  003170  013737  001202  001506  7$:   MOV     $UNIT,EVER     ;SAVE THE # OF MNCAD'S FOR LATER
327  003176  052737  100000  001506        BIS     #BIT15,EVER    ;SET 'REPORTED # FLAG''
328  003204  000410                         BR      5$
329  003206  123737  001506  001202  4$:   CMPB    EVER,$UNIT     ;TEST IF ANY HAVE GONE AWAY
330  003214  001404                         BEQ     5$             ;BR IF ALL ARE STILL THERE
331  003216  113737  001506  001426        MOVB    EVER,TEMP      ;SAVE FOR ERROR REPORT
332  003224  104005                        ERROR   5              ;EXISTING DEVICE FAILED TO RESPOND
333  003226  005037  001202          5$:   CLR     $UNIT          ;RESET UNIT POINTER
334  003232  113737  001506  001440        MOVB    EVER,NMBEXT    ;GET # OF UNITS
335  003240  005337  001440                DEC     NMBEXT         ;ADJUST IT
336  003244  004737  012510                JSR     PC,FIXONE      ;FIX BUS AND VECTOR ADDRESSES
337  003250  005037  001510                CLR     BADUNT         ;RESET BAD UNIT INDICATOR
338  003254  005046                         CLR     -(SP)          ;LOWER PRIORITY LEVEL 0
339  003256  012746  003264                MOV     #6$,-(SP)
340  003262  000002                        RTI
341  003264  000207                  6$:   RTS     PC             ;EXIT
```

N 2

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 4
CVMNAA.P11   DETERMINE THE NUMBER OF MNCAD'S ON THE SYSTEM   SEQ 0026

```
 343   003266                       BEGINL:
 344                                 ;;***********************************************************
 (3)                                 ;*TEST 1           +15 VOLT TEST   (IN-HOUSE TESTER ONLY)
 (3)                                 ;;***********************************************************
 (2)   003266  012737  003266  001106  TST1:   MOV     #TST1,$LPADR
 (1)   003274  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
 345   003302  012737  000001  001102          MOV     #$TN-1,$TSTNM   ;SET UP TEST NUMBER
 346   003310  012737  003266  001110          MOV     #TST1,$LPERR
 347   003316  005737  001476                  TST     WFTEST          ;IS PROGRAM RUNNING IN WESTFIELD MODE?
 348   003322  100075                          BPL     TST3            ;;NO, SKIP FIRST 2 TESTS
 349   003324  005737  001176                  TST     $PASS           ;DO FIRST 2 TESTS ON 1ST PASS ONLY
 350   003330  001072                          BNE     TST3
 351   003332  005046                          CLR     -(SP)           ;RESET PRIORITY
 352   003334  012746  003342                  MOV     #1$,-(SP)
 353   003340  000002                          RTI
 354   003342  104401  020424          1$:     TYPE    ,TP15           ;TYPE ''+15 = ''
 355   003346  004537  015550                  JSR     R5,GCONVT       ;CONVERT CHANNEL 12
 356   003352  000012                          12
 357   003354  013703  001426                  MOV     TEMP,R3         ;GET TEMP
 358   003360  004737  015664                  JSR     PC,CONV15       ;TYPE VOLTAGE
 359   003364  104401  021014                  TYPE    ,SPACE          ;TYPE 4 SPACES
 360   003370  004537  015502                  JSR     R5,COMPAR       ;TEST RESULTS
 361   003374  006020                          6020
 362   003376  016334                          V100D
 363   003400  000403                          BR      2$              ;;ERROR
 364   003402  104401  021132                  TYPE    ,OKMSG          ;TYPE ''OK''
 365   003406  000406                          BR      TST2            ;;GOTO NEXT TEST
 366   003410  104401  021606          2$:     TYPE    ,ERMSG          ;TYPE ''**ERROR**''
 367   003414  004737  026564                  JSR     PC,WHICHV       ;INDICATE ERROR UNIT
 368   003420  005237  001112                  INC     $ERTTL          ;UPDATE ERROR COUNT
 369
 370                                 ;;***********************************************************
 (3)                                 ;*TEST 2           -15 VOLT TEST   (IN-HOUSE TESTER ONLY)
 (3)                                 ;;***********************************************************
 (2)   003424  000004                  TST2:   SCOPE
 (1)   003426  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
 371   003434  104401  020433                  TYPE    ,TM15           ;TYPE ''-15 = ''
 372   003440  004537  015550                  JSR     R5,GCONVT       ;CONVERT CHANNEL 11
 373   003444  000011                          11
 374   003446  013703  001426                  MOV     TEMP,R3         ;GET TEMP
 375   003452  004737  015664                  JSR     PC,CONV15       ;TYPE VOLTAGE
 376   003456  104401  021014                  TYPE    ,SPACE          ;TYPE 4 SPACES
 377   003462  004537  015502                  JSR     R5,COMPAR       ;TEST RESULTS
 378   003466  001760                          1760
 379   003470  016334                          V100D
 380   003472  000403                          BR      1$              ;;ERROR
 381   003474  104401  021132                  TYPE    ,OKMSG          ;TYPE ''OK''
 382   003500  000406                          BR      TST3            ;;GOTO NEXT TEST
 383   003502  104401  021606          1$:     TYPE    ,ERMSG          ;TYPE ''**ERROR**''
 384   003506  004737  026564                  JSR     PC,WHICHV          ;INDICATE BAD UNIT
 385   003512  005237  001112                  INC     $ERTTL             ;UPDATE ERROR COUNT
 386
```

```
388                              ;;********************************************************************
(3)                              ;*TEST 3              FLOAT A ONE THRU MULTIPLEXER BITS
(3)                              ;;********************************************************************
(2)  003516  000004      TST3:   SCOPE
389  003520  012737  000003  001102    MOV     #STN-1,$TSTNM            ;ENSURE PROPER TEST NUMBER
390  003526  012737  000400  001124    MOV     #BIT8,$GDDAT            ;LOAD FIRST BIT
391  003534  013777  001124  175610  2$: MOV   $GDDAT,@STREG          ;LOAD EXPECTED VALUE
392  003542  017737  175604  001126    MOV     @STREG,$BDDAT          ;READ STATUS REGISTER
393  003550  042737  000002  001126    BIC     #BIT1,$BDDAT           ;CLEAR NXC BIT
394  003556  023737  001124  001126    CMP     $GDDAT,$BDDAT          ;COMPARE RESULTS
395  003564  001401                     BEQ     1$
396  003566  104001                     ERROR   1                      ;FAILED TO LOAD + READ BIT
397  003570  006337  001124          1$: ASL    $GDDAT                 ;GET NEXT BIT
398  003574  023727  001124  040000    CMP     $GDDAT,#BIT14          ;FINISHED?
399  003602  001354                     BNE     2$              ;;NO,GO TO NEXT TEST
400                              ;;********************************************************************
(3)                              ;*TEST 4              LOAD AND READ BACK ERROR I.E. BIT14
(3)                              ;;********************************************************************
(2)  003604  000004      TST4:   SCOPE
401  003606  012737  040000  001124    MOV     #BIT14,$GDDAT
402  003614  104415                     CHKIT
403  003616  104001                     ERROR   1                       ;FAILED TO LOAD + READ ERROR I.E.
404                              ;;********************************************************************
(3)                              ;*TEST 5              LOAD AND READ BACK INTERRUPT ENABLE BIT6
(3)                              ;;********************************************************************
(2)  003620  000004      TST5:   SCOPE
405  003622  012777  001516  175530    MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
406  003630  012777  000200  175524    MOV     #200,@VECTR1           ;LOAD BR LEVEL
407  003636  012737  000100  001124    MOV     #BIT6,$GDDAT           ;LOAD EXPECTED DATA
408  003644  104415                     CHKIT
409  003646  104001                     ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE
410                              ;;********************************************************************
(3)                              ;*TEST 6              LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
(3)                              ;;********************************************************************
(2)  003650  000004      TST6:   SCOPE
411  003652  012737  000040  001124    MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
412  003660  104415                     CHKIT
413  003662  104001                     ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB
414                              ;;********************************************************************
(3)                              ;*TEST 7              LOAD AND READ BACK EXTERNAL START ENABLE BIT4
(3)                              ;;********************************************************************
(2)  003664  000004      TST7:   SCOPE
415  003666  012737  000020  001124    MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
416  003674  104415                     CHKIT
417  003676  104001                     ERROR   1                  ;FAILED TO LOAD + READ EXT. START ENABLE
418                              ;;********************************************************************
(3)                              ;*TEST 10             LOAD AND READ BACK MAINT. TST BIT2
(3)                              ;;********************************************************************
(2)  003700  000004      TST10:  SCOPE
419  003702  012737  000004  001124    MOV     #BIT2,$GDDAT
420  003710  104415                     CHKIT
421  003712  104001                     ERROR   1                      ;FAILED TO LOAD + READ BACK MAINT. TST
```

```
 423                            ;;*********************************************************************
 (3)                           ;*TEST 11        LOAD AND READ BACK ENABLE I.D. BIT3
 (3)                           ;;*********************************************************************
 (2)  003714  000004    TST11: SCOPE
 424  003716  012737  000010  001124   MOV    #BIT3,$GDDAT
 425  003724  104415           CHKIT
 426  003726  104001           ERROR   1                               ;FAILED TO LOAD + READ ENABLE I.D. BIT
 427
 428                           ;;*********************************************************************
 (3)                           ;*TEST 12        LOAD AND READ BACK ERROR FLAG BIT15
 (3)                           ;;*********************************************************************
 (2)  003730  000004    TST12: SCOPE
 429  003732  012737  100000  001124   MOV    #BIT15,$GDDAT  ;LOAD EXPECTED DATA
 430  003740  104415           CHKIT
 431  003742  104001           ERROR   1                     ;FAILED TO LOAD + READ ERROR FLAG
 432                           ;;*********************************************************************
 (3)                           ;*TEST 13        TEST INIT CLEARS BITS 2-6,8-14
 (3)                           ;;*********************************************************************
 (2)  003744  000004    TST13: SCOPE
 (1)  003746  012737  000300  001160   MOV    #300,$TIMES    ;;DO 300 ITERATIONS
 433  003754  005037  001124           CLR    $GDDAT         ;LOAD EXPECTED DATA
 434  003760  012777  077574  175364 2$: MOV   #77574,@STREG ;SET STATUS REGISTER
 435  003766  000005           RESET                         ;INITIALIZE
 436  003770  052777  000100  175146   BIS    #100,@STKS     ;SET INTRPT. ENABLE
 437  003776  104414           CHECK                         ;GO CHECK RESULTS
 438  004000  104001           ERROR   1                     ;RESET FAILED TO CLEAR AD ST. REG. BITS
```

D 3

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 7
CVMNAA.P11   T14   TEST INIT CLEARS ERROR FLAG   SEQ 0029

```
440                         ;;*****************************************************************
(3)                         ;*TEST 14          TEST INIT CLEARS ERROR FLAG
(3)                         ;;*****************************************************************
(2)  004002  000004     TST14:  SCOPE
441  004004  012737  000300 001160      MOV    #300,$TIMES             ;DO 300 ITERATIONS
442  004012  012777  100000 175332      MOV    #BIT15,@STREG           ;SET BIT 15
443  004020  000005              RESET                                 ;ISSUE INIT
444  004022  052777  000100 175114      BIS    #100,@$TKS              ;SET INTRPT. EN. FOR KEYBOARD
445  004030  104414              CHECK
446  004032  104001              ERROR  1
447                         ;;*****************************************************************
(3)                         ;*TEST 15          TEST DONE FLAG SETS AND BIT0 CLEARS ON END OF CONV.
(3)                         ;;*****************************************************************
(2)  004034  000004     TST15:  SCOPE
(1)  004036  012737  000100 001160      MOV    #100,$TIMES             ;;DO 100 ITERATIONS
448  004044  012700  001000          MOV    #BIT9,R0                ;STALL TIME COUNTER
449  004050  005277  175276          INC    @STREG                  ;START CONVERSION
450  004054  012737  000200 001124      MOV    #BIT7,$GDDAT            ;LOAD EXPECTED
451  004062  005300          1$:  DEC    R0                      ;STALL
452  004064  001376              BNE    1$                      ;TIME
453  004066  042777  100000 175256      BIC    #BIT15,@STREG           ;MASK OUT ERROR BIT
454  004074  104414              CHECK
455  004076  104001              ERROR  1                       ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
456  004100  017700  175252          MOV    @ADBUFF,R0             ;CLEAR DONE FLAG FOR ITERATIONS
457                         ;;*****************************************************************
(3)                         ;*TEST 16          TEST INIT CLEARS DONE FLAG
(3)                         ;;*****************************************************************
(2)  004104  000004     TST16:  SCOPE
(1)  004106  012737  000300 001160      MOV    #300,$TIMES             ;;DO 300 ITERATIONS
458  004114  005037  001124          CLR    $GDDAT                  ;CLEAR EXPECTED
459  004120  005277  175226          INC    @STREG                  ;START CONVERSION
460  004124  105777  175222      2$:  TSTB   @STREG
461  004130  100375              BPL    2$
462  004132  000005              RESET
463  004134  052777  000100 175002      BIS    #BIT6,@$TKS             ;ENABLE INTR.
464  004142  104414              CHECK
465  004144  104001              ERROR  1                       ;DONE FLAG FAILED TO CLEAR
466
467                         ;;*****************************************************************
(3)                         ;*TEST 17          TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
(3)                         ;;*****************************************************************
(2)  004146  000004     TST17:  SCOPE
468  004150  005037  001124          CLR    $GDDAT                  ;CLEAR EXPECTED
469  004154  005277  175172          INC    @STREG                  ;SET A/D START CONVERSION BIT
470  004160  105777  175166      1$:  TSTB   @STREG                  ;WAIT FOR FLAG
471  004164  100375              BPL    1$
472  004166  017700  175164          MOV    @ADBUFF,R0             ;READ CONVERTED VALUE
473  004172  104414              CHECK
474  004174  104001              ERROR  1                       ;DONE FLAG FAILED TO CLEAR
```

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 8
CVMNAA.P11   T20   TEST ALL 'O'S RESULTS USING MAINT. ADTST. BIT

E 3

SEQ U030

```
476                                 ;:**********************************************************
(3)                                 ;*TEST 20        TEST ALL 'O'S RESULTS USING MAINT. ADTST. BIT
(3)                                 ;:**********************************************************
(2)  004176  000004                 TST20:  SCOPE
477  004200  005037  001124                 CLR     $GDDAT          ;CLEAR EXPECTED VALUE
478  004204  005037  001444                 CLR     CHANL           ;SET CHANL = 0
479  004210  005037  001462                 CLR     SPREAD          ;SET SPREAD = 0
480  004214  012777  000005  175130          MOV     #5,aSTREG       ;CONVERT EVEN CHANNEL WITH MAINT. BIT SET
481  004222  105777  175124         1$:     TSTB    aSTREG          ;WAIT FOR DONE
482  004226  100375                          BPL     1$
483  004230  017737  175122  001126          MOV     aADBUFF,$BDDAT  ;RESULTS TO BDDAT FOR CHECKING
484  004236  001401                          BEQ     TST21           ;;GOTO NEXT TEST
485  004240  104004                          ERROR   4               ;DID NOT GET ALL 'O'S RSULT WITH MAINT. ADTST
486                                 ;:**********************************************************
(3)                                 ;*TEST 21        TEST ALL '1'S RESULT USING MAINT. ADTST. BIT
(3)                                 ;:**********************************************************
(2)  004242  000004                 TST21:  SCOPE
487  004244  012737  007777  001124          MOV     #7777,$GDDAT    ;EXPECT ALL '1'S RESULT
488  004252  012737  000001  001444          MOV     #1,CHANL        ;SET CHANL = 1
489  004260  005037  001462                  CLR     SPREAD          ;SET SPREAD = 0
490  004264  012777  000405  175060          MOV     #405,aSTREG     ;CONVERT ODD CHANNEL WITH MAINT. BIT SET
491  004272  105777  175054         1$:     TSTB    aSTREG          ;WAIT FOR DONE
492  004276  100375                          BPL     1$
493  004300  017737  175052  001126          MOV     aADBUFF,$BDDAT  ;RESULTS TO BDDAT FOR CHECKING
494  004306  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;EQUAL?
495  004314  001401                          BEQ     TST22           ;;GOTO NEXT TEST
496  004316  104004                          ERROR   4               ;DID NOT GET ALL '1'S RESULT WITH MAINT. ADTST
497                                 ;:**********************************************************
(3)                                 ;*TEST 22        GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
(3)                                 ;:**********************************************************
(2)  004320  000004                 TST22:  SCOPE
(1)  004322  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
498  004330  012737  004336  001106          MOV     #10$,$LPADR     ;LOAD RETURN ADDRESS
499  004336  042777  000100  174600  10$:    BIC     #BIT6,a$*KS     ;REMOVE TKB INTERRUPT
500  004344  005046                          CLR     -(SP)           ;RESET PRIORITY
501  004346  012746  004354                  MOV     #1$,-(SP)
502  004352  000002                          RTI
503  004354  004737  013166         1$:     JSR     PC,SETINT       ;LOAD VECTOR AREA WITH TRAP CATCHER
504  004360  012777  004442  174772          MOV     #3$,aVECTOR     ;INTERRUPT VECTOR ADDRESS
505  004366  012777  000200  174766          MOV     #200,aVECTR1    ;SET UP NEW PSW
506  004374  012777  000101  174750          MOV     #BIT6!BIT0,aSTREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
507  004402  105777  174744         2$:     TSTB    aSTREG          ;WAIT FOR DONE
508  004406  100375                          BPL     2$              ;FLAG TO SET
509  004410  017737  174736  001126          MOV     aSTREG,$BDDAT   ;READ STATUS REGISTER
510  004416  005077  174730                  CLR     aSTREG          ;ENSURE INTR. ENABLE IS CLEARED
511  004422  017737  174730  001124          MOV     aADBUFF,$GDDAT  ;READ TO CLEAR DONE FLAG
512  004430  012737  000300  001124          MOV     #BIT7!BIT6,$GDDAT ;LOAD EXPECTED GOOD DATA
513  004436  104002                          ERROR   2               ;FAILED TO INTERRUPT ON DONE
514  004440  000401                          BR      4$              ;;BRANCH TO NEXT TEST
515  004442  022626                 3$:     CMP     (SP)+,(SP)+     ;RESET STACK POINTER
516  004444  013777  001362  174706  4$:     MOV     VECTR1,aVECTOR  ;SET UP FOR UNEXPECTED INTERRUPT
517  004452  012777  004700  174702          MOV     #4700,aVECTR1
518  004460  005046                          CLR     -(SP)           ;CLEAR PSW
519  004462  012746  004470                  MOV     #5$,-(SP)
```

```
520   004466  000002                          RTI
521   004470  005077  174656         5$:      CLR     @STREG
522   004474  005777  174656                  TST     @ADBUFF          ;CLEAR DONE BIT
523
524                               ;;****************************************************************
(3)                              ;*TEST 23         TEST INTERRUPT OCCURS WHEN ERROR AND I.E.E. IS SET
(3)                              ;;****************************************************************
(2)   004500  000004             TST23:  SCOPE
(1)   004502  012737  000100  001160         MOV     #100,$TIMES      ;;DO 100 ITERATIONS
525   004510  012737  004516  001106         MOV     #10$,$LPADR      ;LOAD RETURN ADDRESS
526   004516  042777  000100  174420  10$:   BIC     #BIT6,@$TKS      ;REMOVE TKB INTERRUPT
527   004524  005046                          CLR     -(SP)            ;LOWER PRIORITY
528   004526  012746  004534                  MOV     #1$,-(SP)
529   004532  000002                          RTI
530   004534  004737  013166         1$:      JSR     PC,SETINT        ;LOAD VECTOR AREA WITH TRAP CATCHER
531   004540  012777  004612  174616          MOV     #2$,@VECTR2      ;SETUP VECTOR ADDRESS
532   004546  012777  000200  174612          MOV     #200,@VECTR3     ;SET UP NEW PSW
533   004554  012777  140000  174570          MOV     #BIT15!BIT14,@STREG    ;CAUSE AN INTERRUPT
534   004562  017737  174564  001126          MOV     @STREG,$BDDAT    ;BAD DATA
535   004570  012737  140000  001124          MOV     #BIT15!BIT14,$GDDAT    ;GOOD DATA
536   004576  005077  174550                  CLR     @STREG           ;CLEAR STATUS
537   004604  005777  174550                  TST     @ADBUFF          ;AND CLEAR DONE
538   004606  104002                          ERROR   2
539   004610  000401                          BR      3$
540   004612  022626                 2$:      CMP     (SP)+,(SP)+      ;POP STACK
541   004614  005077  174532         3$:      CLR     @STREG           ;CLEAR STATUS REG.
542   004620  005777  174532                  TST     @ADBUFF          ;FALSE READ TO CLEAR DONE
543   004624  013777  001366  174532          MOV     VECTR3,@VECTR2   ;RESET VECTOR
544   004632  012777  004700  174526          MOV     #4700,@VECTR3    ;
545   004640  005046                          CLR     -(SP)            ;RESET PRIORITY
546   004642  012746  004650                  MOV     #4$,-(SP)
547   004646  000002                          RTI
548   004650  005077  174476         4$:      CLR     @STREG
549                               ;;****************************************************************
(3)                              ;*TEST 24         TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
(3)                              ;;****************************************************************
(2)   004654  000004             TST24:  SCOPE
550   004656  012777  000001  174466          MOV     #BIT0,@STREG     ;START CONVERSION
551   004664  052777  000100  174252          BIS     #BIT6,@$TKS      ;ENABLE TKB INTERRUPT
552   004672  105777  174454         1$:      TSTB    @STREG           ;WAIT FOR
553   004676  100375                          BPL     1$
554   004700  012737  100200  001124  2$:     MOV     #BIT15!BIT7,$GDDAT  ;LOAD EXPECTED VALUE
555   004706  012777  000001  174436          MOV     #BIT0,@STREG     ;START 2ND CONVERSION
556   004714  012700  001000                  MOV     #BIT9,R0         ;WAIT FOR 2ND
557   004720  005300                 3$:      DEC     R0               ;CONVERSION TO END
558   004722  001376                          BNE     3$
559   004724  104414                 4$:      CHECK
560   004726  104001                          ERROR   1                ;ERROR FLAG NOT SET WHEN 2ND
561                                                                     ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
562   004730  017700  174422                  MOV     @ADBUFF,R0       ;CLEAR DONE FLAG
```

G 3

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 9
CVMNAA.P11   T25   TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS

SEQ 0032

```
  564                                    ;;**********************************************************
  (3)                                    ;*TEST 25        TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
  (3)                                    ;;**********************************************************
  (2)   004734   000004         TST25:   SCOPE
  565   004736   012737  100000  001124            MOV     #BIT15,$GDDAT    ;LOAD EXPECTED DATA
  566   004744   012777  000001  174400            MOV     #BIT0,@STREG     ;START CONVERSION
  567   004752   112777  000001  174372            MOVB    #BIT0,@STREG     ;START NEXT CONVERSION
  568   004760   112777  000001  174364            MOVB    #BIT0,@STREG     ;ONCE AGAIN IN CASE REFRESH INTERVENED
  569   004766   017737  174360  001126            MOV     @STREG,$BDDAT    ;READ STATUS REGISTER
  570   004774   042737  077777  001126            BIC     #77777,$BDDAT    ;MASK OUT BIT 15
  571   005002   023737  001124  001126            CMP     $GDDAT,$BDDAT    ;COMPARE RESULTS
  572   005010   001401                            BEQ     1$               ;;BRANCH OVER ERROR
  573   005012   104001                            ERROR   1                ;ERROR FLAG NOT SET WHEN 2ND
  574                                                                       ;CONVERT BEGINS BEFORE FIRST DONE
  575   005014   017700  174336          1$:       MOV     @ADBUFF,R0
  576   005020   005077  174326                    CLR     @STREG                           ;CLEAR STATUS REGISTER
  577                                    ;;**********************************************************
  (3)                                    ;*TEST 26        TEST CHANNELS 0-7 FOR SINGLE ENDED
  (3)                                    ;;**********************************************************
  (2)   005024   000004         TST26:   SCOPE
  578   005026   005037  001124          CLR     $GDDAT
  579   005032   013777  000010  174312            MOV     BIT3,@STREG      ;ENABLE PREAMP STATUS
  580   005040   005277  174306          1$:       INC     @STREG           ;START A CONVERSION
  581   005044   105777  174302          2$:       TSTB    @STREG           ;IS CONVERSION DONE?
  582   005050   100375                            BPL     2$               ;NO, WAIT TILL IT IS DONE
  583   005052   017737  174300  001126            MOV     @ADBUFF,$BDDAT   ;GET PREAMP STATUS
  584   005060   042737  007777  001126            BIC     #7777,$BDDAT     ;MASK OUT CONVERTED VALUE
  585   005066   001401                            BEQ     3$               ;;SKIP OVER ERROR IF ZERO
  586   005070   104001                            ERROR   1
  587   005072   062777  000400  174252  3$:       ADD     #BIT8,@STREG     ;INCREMENT CHANNEL TO BE TESTED
  588   005100   032777  004000  174244            BIT     #BIT11,@STREG    ;IS IT DONE?
  589   005106   001754                            BEQ     1$               ;;NO
  590                                    ;;**********************************************************
  (3)                                    ;*TEST 27        TEST CLOCK OVERFLOW STARTS A/D   (TESTER ONLY)
  (3)                                    ;;**********************************************************
  (2)   005110   000004         TST27:   SCOPE
  591   005112   005737  001476          TST     WFTEST           ;RUNNING IN NORMAL MODE?
  592   005116   100022                            BPL     2$               ;;YES, GO TO NEXT TEST
  593   005120   012737  000240  001124            MOV     #BIT7!BIT5,$GDDAT        ;SET UP EXPECTED RESULT
  594   005126   013777  001124  174216            MOV     $GDDAT,@STREG    ;ENABLE CLOCK OVERFLOW START
  595   005134   012700  001000                    MOV     #BIT9,R0         ;STALL TIME COUNTER
  596   005140   012777  177777  174240            MOV     #177777,@CLKBPR  ;SET CLOCK NEAR OVERFLOW
  597   005146   012777  000011  174230            MOV     #11,@CLKCSR      ;START CLOCK AT LINE RATE
  598   005154   005300                  1$:       DEC     R0               ;STALL
  599   005156   001376                            BNE     1$               ;TIME
  600   005160   104414                            CHECK                    ;CHECK RESULT
  601   005162   104001                            ERROR   1                ;DONE FLAG FAILED TO SET
  602   005164   005777  174166          2$:       TST     @ADBUFF          ;CLEAR DONE FLAG
  603   005170   005077  174156                    CLR     @STREG           ;INHIBIT CLOCK OVERFLOW START
```

```
605                                      ;;*********************************************************************
(3)                                      ;*TEST 30          TEST EXTERNAL START STARTS A/D   (TEST MODULE OR TESTER)
(3)                                      ;;*********************************************************************
(2)   005174  000004        TST30:  SCOPE
(1)   005176  012737  000001 001160        MOV    #1,$TIMES          ;;DO 1 ITERATION
606   005204  005737  001476               TST    WFTEST             ;RUNNING IN DWARF OR TESTER MODE?
607   005210  001453                       BEQ    4$                 ;;NO, GO TO NEXT TEST
608   005212  012737  000220 001124        MOV    #BIT7!BIT4,$GDDAT       ;SET UP EXPECTED RESULT
609   005220  013777  001124 174124        MOV    $GDDAT,@STREG      ;ENABLE EXTERNAL START
610   005226  005737  001476               TST    WFTEST             ;RUNNING IN TESTER MODE?
611   005232  100013                       BPL    2$                 ;;NO
612   005234  012700  001000               MOV    #BIT9,R0           ;STALL TIME COUNTER
613   005240  052777  000400 174144        BIS    #BIT8,@DRVDOR      ;GENERATE EXTERNAL START
614   005246  042777  000400 174136        BIC    #BIT8,@DRVDOR      ;RESET BIT
615   005254  005300        1$:    DEC    R0                 ;STALL
616   005256  001376                       BNE    1$                 ;TIME
617   005260  000425                       BR     3$                 ;;TEST RESULTS
618   005262  105737  001134        2$:    TSTB   $AUTOB             ;IS IT UNDER A MONITOR ?
619   005266  001024                       BNE    4$                 ;;YES
620   005270  005737  001176               TST    $PASS              ;IS IT THE FIRST PASS?
621   005274  001021                       BNE    4$                 ;;NO, DON'T RUN TEST
622   005276  104401  020330               TYPE   ,EXTST             ;TYPE MESSAGE ABOUT EXT. START
623   005302  004737  026572               JSR    PC,WHICHU          ;DETERMINE UNIT #
624   005306  013746  001514               MOV    UNITBD,-(SP)       ;;SAVE UNITBD FOR TYPEOUT
(1)   005312  104403                       TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)   005314  001                          .BYTE  1                  ;;TYPE 1 DIGIT(S)
(1)   005315  000                          .BYTE  0                  ;;SUPPRESS LEADING ZEROS
625   005316  104401  021463               TYPE   ,CRWR              ;TYPE "TYPE CR WHEN READY"
626   005322  104412                       RDLIN                     ;WAIT FOR CR
627   005324  005726                       TST    (SP)+              ;POP WORD OFF STACK
628   005326  042777  100000 174016        BIC    #BIT15,@STREG      ;CLEAR A/D ERROR
629   005334  104414        3$:    CHECK                     ;CHECK RESULT
630   005336  104001                       ERROR  1                  ;DONE FLAG FAILED TO SET
631   005340  005777  174012        4$:    TST    @ADBUF             ;CLEAR DONE FLAG
632   005344  005077  174002               CLR    @STREG             ;INHIBIT EXTERNAL START
633
634
635                                      ;****************************
636   005350  000004               SCOPE
637   005352  000207               RTS    PC                 ;RETURN TO TEST SECTION
638                                      ;****************************
639
640
641                                      ;;SUBROUTINE FOR LOGIC TESTS;;
642   005354  013777  001124 173770 TESTIT: MOV    $GDDAT,@STREG            ;LOAD EXPECTED VALUE
643   005362  017737  173764 001126 TEST:   MOV    @STREG,$BDDAT           ;READ ST. REG.
644   005370  023737  001124 001126         CMP    $GDDAT,$BDDAT           ;COMPARE RESULTS
645   005376  001002                        BNE    RETERR                  ;;ERROR RETURN
646   005400  062716  000002                ADD    #2,(SP)                 ;BUMP RETURN ADDRESS TO GET AROUND ERROR
647   005404  000002        RETERR: RTI
```

```
649                                  .SBTTL          WRAPAROUND TEST SECTION
650   005406                     WRAP:
651                              ;;**********************************************************
(3)                             ;*TEST 31        TEST CH0 GROUND
(3)                             ;;**********************************************************
(2)   005406  012737  000031  001102  TST31:  MOV     #STN,STSTNM
(1)   005414  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
652   005422  012737  005406  001106          MOV     #TST31,$LPADR   ;*SET UP LOOP ADDRESS
(2)   005430  012737  005406  001110          MOV     #TST31,$LPERR   ;*SET UP ERROR LOOP ADDRESS
653   005436  005737  001370                  TST     BASECH          ;TESTING CHANNEL 0-7?
654   005442  001111                          BNE     WRAPX           ;NO, DON'T TEST
655   005444  004537  015352                  JSR     R5,CONVRT       ;CONVERT 8 TIMES
656   005450  000000                          0
657   005452  004537  015502                  JSR     R5,COMPAR       ;COMPARE RESULTS
658   005456  004000                          4000            ;NOMINAL
659   005460  016330                          V12             ;TOLERANCE
660   005462  104004                          ERROR   4       ;ERROR ON A/D CHANNEL
661                              ;;**********************************************************
(3)                             ;*TEST 32        TEST CH1 +4.5 VOLT
(3)                             ;;**********************************************************
(2)   005464  000004                  TST32:  SCOPE
(1)   005466  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
662   005474  004537  015352                  JSR     R5,CONVRT       ;CONVERT 8 TIMES
663   005500  000001                          1               ;CHANNEL 1
664   005502  004537  015502                  JSR     R5,COMPAR       ;COMPARE RESULTS
665   005506  007344                          7344            ;NOMINAL
666   005510  016336                          V326            ;TOLERANCE
667   005512  104004                          ERROR   4       ;ERROR ON A/D CHANNEL
668
669                              ;;**********************************************************
(3)                             ;*TEST 33        TEST CH2 -4.5 VOLT
(3)                             ;;**********************************************************
(2)   005514  000004                  TST33:  SCOPE
(1)   005516  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
670   005524  004537  015352                  JSR     R5,CONVRT       ;CONVERT 8 TIMES
671   005530  000002                          2               ;CHANNEL 2
672   005532  004537  015502                  JSR     R5,COMPAR       ;COMPARE RESULTS
673   005536  000434                          434             ;NOMINAL
674   005540  016336                          V326            ;TOLERANCE
675   005542  104004                          ERROR   4       ;ERROR ON A/D CHANNEL
676                              ;;**********************************************************
(3)                             ;*TEST 34        TEST CH5 GROUND (DWARF OR TESTER)
(3)                             ;;**********************************************************
(2)   005544  000004                  TST34:  SCOPE
(1)   005546  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
677   005554  005737  001476                  TST     WFTEST          ;RUNNING WITHOUT TESTER OR DWARF
678   005560  001002                          BNE     1$              ;;NO
679   005562  000137  006240                  JMP     STD             ;YES, BYPASS DWARF/TESTER CHECKS
680   005566  004537  015352          1$:     JSR     R5,CONVRT       ;CONVERT 8 TIMES
681   005572  000005                          5               ;CHANNEL 5
682   005574  004537  015502                  JSR     R5,COMPAR       ;COMPARE RESULTS
683   005600  004000                          4000            ;NOMINAL
684   005602  016330                          V12             ;TOLLERANCE
685   005604  104004                          ERROR   4       ;ERROR ON A/D CHANNEL
```

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 11-1
CVMNAA.P11   T34   TEST CH5 GROUND (DWARF OR TESTER)   SEQ 0035

                                                          J 3

```
686
687                           ;:***************** **************************************
(3)                           ;*TEST 35        TEST CH4 +2.6 VOLTS      (DWARF OR TESTER)
(3)                           ;:***********************************************************
(2)   005606  000004          TST35:  SCOPE
(1)   005610  012737  000010  001160          MOV     #10,$TIMES     ;;DO 10 ITERATIONS
688   005616  004537  015352                  JSR     R5,CONVRT      ;CONVERT 8 TIMES
689   005622  000004                          4                      ;CHANNEL 4
690   005624  004537  015502                  JSR     R5,COMPAR      ;COMPARE RESULTS
691   005630  006020                          6020                   ;NOMINAL
692   005632  016336                          V326                   ;TOLLERANCE
693   005634  104004                          ERROR   4              ;ERROR ON A/D CHANNEL
694
695                           ;:***************** **************************************
(3)                           ;*TEST 36        TEST CH6 -2.2 VOLTS      (DWARF OR TESTER)
(3)                           ;:***********************************************************
(2)   005636  000004          TST36:  SCOPE
(1)   005640  012737  000010  001160          MOV     #10,$TIMES     ;;DO 10 ITERATIONS
696   005646  004537  015352                  JSR     R5,CONVRT      ;CONVERT 8 TIMES
697   005652  000006                          6                      ;CHANNEL 6
698   005654  004537  015502                  JSR     R5,COMPAR      ;COMPARE RESULTS
699   005660  001760                          1760                   ;NOMINAL
700   005662  016336                          V326                   ;TOLLERANCE
701   005664  104004                          ERROR   4              ;ERROR ON A/D CHANNEL
702
703   005666                  WRAPX:
704                           ;:***************** **************************************
(3)                           ;*TEST 37        TEST VOLTAGE ON CHANNELS        (DWARF OR TESTER)
(3)                           ;:***********************************************************
(2)   005666  000004          TST37:  SCOPE
(1)   005670  012737  000010  001160          MOV     #10,$TIMES     ;;DO 10 ITERATIONS
705   005676  012737  000037  001102          MOV     #$TN-1,$TSTNM  ;SET UP TEST NUMBER
706   005704  013737  001370  001444          MOV     BASECH,CHANL   ;SET UP CHANNEL
707   005712  001003                          BNE     1$             ;;CHANNEL OK
708   005714  012737  000010  001444          MOV     #10,CHANL      ;ON CHANNEL 10
709   005722  012705  016340          1$:     MOV     #VTABLE,R5     ;POINT TO VOLTAGE TABLE
710   005726  012537  005752          2$:     MOV     (R5)+,4$       ;SET UP EXPECTED VALUE
711   005732  023737  001372  001444          CMP     BASEND,CHANL   ;DONE?
712   005740  103415                          BLO     TST40          ;;YES, GO TO NEXT TEST
713   005742  004537  015360          3$:     JSR     R5,CONVTC      ;CONVERT 8 TIMES
714   005746  004537  015502                  JSR     R5,COMPAR      ;COMPARE RESULTS
715   005752  005560          4$:             5560                   ;VOLTAGE
716   005754  016336                          V326                   ;TOLLERANCE
717   005756  104004                          ERROR   4              ;ERROR ON A/D CHANNEL
718   005760  005237  001444                  INC     CHANL          ;GET NEXT CHANNEL
719   005764  020527  016360                  CMP     R5,#VTABLE+20  ;IS VOLTAGE TABLE POINTER AT END OF TABLE?
720   005770  001356                          BNE     2$             ;;NO, GET NEXT EXPECTED VALUE
721   005772  000753                          BR      1$             ;;
```

K 3

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 12   SEQ 0036
CVMNAA.P11   T40   TEST CHANNEL FOR +- 2.2 VOLTS IN DIFFERENTIAL MODE

CV
CV

```
 723                                    ;;****************************************************************
  (3)                                   ;*TEST 40        TEST CHANNEL FOR +- 2.2 VOLTS IN DIFFERENTIAL MODE
  (3)                                   ;;****************************************************************
  (2)  005774  000004          TST40:   SCOPE
  (1)  005776  012737  000001  001160            MOV    #1,$TIMES       ;;DO 1 ITERATION
 724  006004  005737  001476            TST    WFTEST          ;RUNNING IN TESTER MODE?
 725  006010  100007                    BPL    1$              ;;BR IF NO
 726  006012  012737  000010  001160            MOV    #10,$TIMES      ;DO 10 ITERATIONS
 727  006020  052777  000200  173364            BIS    #BIT7,@DRVDOR   ;SET A/D AND MUX TO DIFFERENTIAL
 728  006026  000423                    BR     3$              ;;
 729  006030  005737  001176    1$:     TST    $PASS           ;IS THIS THE FIRST PASS?
 730  006034  001101                    BNE    TST41           ;;NO, DON'T RUN TEST
 731  006036  023727  001372  000010            CMP    BASEND,#10      ;TESTING ANY DIFFERENTIAL CHANNELS?
 732  006044  103475                    BLO    TST41           ;;NO
 733  006046  104401  020235            TYPE   .DIFM           ;TYPE SET DWARF TO DIFFERENTIAL MESSAGE
 734  006052  104401  021463            TYPE   .CRWR           ;TYPE "TYPE CR WHEN READY"
 735  006056  104412                    RDLIN                  ;WAIT FOR CARRIAGE RETURN
 736  006060  005726                    TST    (SP)+           ;POP WORD OFF STACK
 737  006062  012737  006076  001106    2$:     MOV    #3$,$LPADR      ;SET UP LOOP ADDRESS
 738  006070  012737  006076  001110            MOV    #3$,$LPERR      ;SET UP ERROR LOOP ADDRESS
 739  006076  012737  002220  006156    3$:     MOV    #2220,6$        ;SET UP INITIAL EXPECTED VALUE -2.2 V
 740  006104  013700  001370            MOV    BASECH,R0       ;GET FIRST CHANNEL TO TEST
 741  006110  020027  000010            CMP    R0,#10          ;IS R0 >= 10
 742  006114  103002                    BHIS   4$              ;;YES
 743  006116  012700  000010            MOV    #10,R0          ;SET R0 = 10
 744  006122  013705  001372    4$:     MOV    BASEND,R5       ;GET LAST CHANNEL TO TEST
 745  006126  160005                    SUB    R0,R5           ;GET DIFFERENCE BETWEEN FIRST AND LAST
 746  006130  006205                    ASR    R5              ;DIVIDE IT IN HALF
 747  006132  060005                    ADD    R0,R5           ;ADD FIRST CHANNEL GIVING LAST CHANNEL
 748                                                           ;TO TEST
 749  006134  010037  001444            MOV    R0,CHANL        ;SET UP FIRST CHANNEL TO TEST
 750  006140  020537  001444    5$:     CMP    R5,CHANL        ;DONE?
 751  006144  103417                    BLO    7$              ;;YES, GO TO NEXT TEST
 752  006146  004537  015360            JSR    R5,CONVTC       ;CONVERT 8 TIMES
 753  006152  004537  015502            JSR    R5,COMPAR       ;TEST RESULTS
 754  006156  002220            6$:     2220                   ;NOMINAL
 755  006160  016336                    V326                   ;TOLLERANCE
 756  006162  104004                    ERROR  4               ;ERROR ON A/D CHANNEL
 757  006164  005237  001444            INC    CHANL           ;POINT TO NEXT CHANNEL TO TEST
 758  006170  005437  006156            NEG    6$              ;REVERSE SIGN OF EXPECTED VALUE
 759  006174  042737  170000  006156            BIC    #170000,6$      ;CLEAR EXTRA BITS
 760  006202  000756                    BR     5$
 761  006204  005737  001476    7$:     TST    WFTEST          ;RUNNING IN TESTER MODE?
 762  006210  100005                    BPL    8$              ;;BR IF DWARF
 763  006212  000005                    RESET
 764  006214  052777  000100  172722            BIS    #100,@$TKS      ;ENABLE INTERRUPTS
 765  006222  000406                    BR     TST41           ;;GO TO NEXT TEST
 766  006224  104401  020125    8$:     TYPE   .SDSE           ;TYPE SET DWARF TO SINGLE ENDED MESSAGE
 767  006230  104401  021463            TYPE   .CRWR           ;TYPE "TYPE CR WHEN READY"
 768  006234  104412                    RDLIN                  ;WAIT FOR CR
 769  006236  005726                    TST    (SP)+           ;POP WORD OFF STACK
```

```
 771   006240                           STD:
 772                                     ;;************* ***************************************
 (3)                                     ;*TEST 41        TEST VERNIER OFFSET DAC ON CH0
 (3)                                     ;;************* ***************************************
 (2)   006240  000004                    TST41:  SCOPE
 (1)   006242  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
 773   006250  012737  000041  001102            MOV     #STN-1,$TSTNM   ;SET UP TEST NUMBER
 774   006256  005077  173074                     CLR     @ADBUFF         ;SET VERNIER DAC = 0
 775   006262  005037  001444                     CLR     CHANL           ;SET UP TO CONVERT ON CHANNEL 0
 776   006266  004537  015366                     JSR     R5,CONVCD       ;CONV. CH0, DIRECT VERNIER DAC
 777   006272  013704  001426                     MOV     TEMP,R4         ;SAVE VALUE IN R4
 778   006276  012777  000377  173052    1$:      MOV     #377,@ADBUFF    ;SET VERNIER DAC = 377
 779   006304  004537  015366                     JSR     R5,CONVCD       ;CONVERT IT
 780   006310  160437  001426                     SUB     R4,TEMP         ;TEMP=DIFF. BETWEEN VALUE & PREVIOUS
 781   006314  004537  015502                     JSR     R5,COMPAR       ;COMPARE RESULTS
 782   006320  000005                             5
 783   006322  016324                             V2
 784   006324  104004                             ERROR   4
 785                                     ;;************* ***************************************
 (3)                                     ;*TEST 42        OFFSET ON CH0
 (3)                                     ;;************* ***************************************
 (2)   006326  000004                    TST42:  SCOPE
 (1)   006330  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
 786   006336  104401  017237                     TYPE    ,OFSET          ;INFORM OPER. TEST NAME
 787   006342  004737  026572                     JSR     PC,WHICHU       ;GET UNIT #
 788   006346  013746  001514                     MOV     UNITBD,-(SP)    ;PUSH IT
 789   006352  104403                             TYPOS                   ;TELL OPER.
 790   006354     001     000             .BYTE   1,0             ;LOAD CHANNEL
 791   006356  005037  001444                     CLR     CHANL           ;LOAD CHANNEL
 792   006362  005037  001442                     CLR     DUMMY           ;LOAD DUMMY
 793   006366  004737  007670                     JSR     PC,OFFSET       ;FIND OFFSET
 794   006372  104401  022775                     TYPE    ,MOFSET         ;TYPE 'OFFSET='
 795   006376  004737  007772                     JSR     PC,TOFF         ;TYPE OFFSET
 796   006402  004537  015502                     JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
 797   006406  000000                             0
 798   006410  016332                             V50D
 799   006412  000401                             BR      OFFERR          ;NO-ERROR
 800   006414  000407                             BR      OFFOK           ;YES-OK
 801   006416  104401  021606            OFFERR:  TYPE    ,ERMSG
 802   006422  004737  026564                     JSR     PC,WHICHV               ;INDICATE BAD UNIT
 803   006426  005237  001112                     INC     $ERTTL                  ;UPDATE ERROR COUNT
 804   006432  000402                             BR      TST43           ;;GO TO NEXT TEST
 805   006434  104401  021132            OFFOK:   TYPE    ,OKMSG
```

```
807                                    ;:************************************************************
(3)                                    ;*TEST 43          TEST RAMP RANGE, CH3
(3)                                    ;:************************************************************
(2)   006440  000004           TST43:  SCOPE
808   006442  012737  000001  001160   MOV    #1,$TIMES              ;DO THIS ONCE
809   006450  012703  007777           MOV    #7777,R3              ;INIT R3 VALUE
810   006454  005004                   CLR    R4                    ;AND R4
811   006456  012777  001400  172666   MOV    #1400,@STREG          ;SETUP FOR CH3
812   006464  012702  047040           MOV    #20000.,R2            ;SETUP FOR 20,000 CONVERSIONS
813   006470  105277  172656    1$:    INCB   @STREG
814   006474  105777  172652    2$:    TSTB   @STREG
815   006500  100375                   BPL    2$
816   006502  027704  172650           CMP    @ADBUFF,R4
817   006506  003402                   BLE    3$
818   006510  017704  172642           MOV    @ADBUFF,R4            ;HIT A NEW HIGH
819   006514  027703  172636    3$:    CMP    @ADBUFF,R3
820   006520  002002                   BGE    4$
821   006522  017703  172630           MOV    @ADBUFF,R3            ;HIT A NEW LOW
822   006526  005302           4$:     DEC    R2
823   006530  001357                   BNE    1$
824   006532  010337  001426           MOV    R3,TEMP
825   006536  004537  015502           JSR    R5,COMPAR
826   006542  000000                   0
827   006544  016322                   V0
828   006546  104004                   ERROR  4                     ;RAMP DIDN'T REACH LOW END OF RANGE
829   006550  010437  001426           MOV    R4,TEMP
830   006554  004537  015502           JSR    R5,COMPAR
831   006560  007777                   7777
832   006562  016322                   V0
833   006564  104004                   ERROR  4                     ;RAMP DIDN'T REACH HIGH END OF RANGE
```

```
 835                                    ;;************************************************************
 (3)                                    ;*TEST 44          NOISE TEST, 1 EDGE
 (3)                                    ;;************************************************************
 (2)  006566  000004         TST44:  SCOPE
 (1)  006570  012737  000001  001160          MOV     #1,$TIMES          ;;DO 1 ITERATION
 836  006576  005037  001416                  CLR     WIDE               ;CLEAR ENTRY FLAG
 837  006602  004737  006612                  JSR     PC,NOITST          ;RUN NOISE TEST
 838  006606  000137  007366                  JMP     NOIJMP             ;NEXT TEST
 839  006612  104401  017154         NOITST: TYPE    ,NOIMSG
 840  006616  004737  026572                  JSR     PC,WHICHU          ;DETERMINE UNIT #
 841  006622  013746  001514                  MOV     UNITBD,-(SP)
 842  006626  104403                          TYPOS                      ;TELL OPER.
 843  006630     001     000                  .BYTE   1,0
 844  006632  104401  001165                  TYPE    ,$CRLF
 845  006636  005737  001476                  TST     WFTEST             ;RUNNING ON THE TESTER
 846  006642  100007                          BPL     1$                 ;;BR IF NOT
 847  006644  012737  000020  001444          MOV     #20,CHANL          ;ASSUME TESTING AM
 848  006652  022737  000057  001372          CMP     #57,BASEND         ;TESTING AM?
 849  006660  001403                          BEQ     2$                 ;;YES
 850  006662  013737  001370  001444  1$:     MOV     BASECH,CHANL       ;LOAD CHANNEL 0
 851  006670  012737  006700  007770  2$:     MOV     #3$,ERRADR         ;SET UP ERROR RETRY ADDRESS
 852  006676  005005                          CLR     R5                 ;CLEAR RETRY COUNT
 853  006700  005205                  3$:     INC     R5                 ;INCREMENT COUNT
 854  006702  022705  000006                  CMP     #6,R5              ;IS COUNT = 6?
 855  006706  001450                          BEQ     5$                 ;;YES, CHANNEL TOO WIDE OR NOISY
 856  006710  013737  001444  001442          MOV     CHANL,DUMMY        ;LOAD DUMMY CHANNEL
 857  006716  004537  015360                  JSR     R5,CONVTC          ;GET EDGE VALUE
 858  006722  013737  001426  001470          MOV     TEMP,EDGE          ;SET UP EDGE VALUE
 859  006730  004537  013224                  JSR     R5,SARSUB          ;DO SAR ROUTINE AT 16%
 860  006734  000020                          16.
 861  006736  004737  007744                  JSR     PC,TSTDAC          ;CHECK VERNIER DAC SETTING
 862  006742  013737  001464  001454          MOV     DAC,RMS            ;ADD RESULT TO RMS
 863  006750  004537  013224                  JSR     R5,SARSUB          ;DO SAR ROUTINE AT 84%
 864  006754  000124                          84.
 865  006756  004737  007744                  JSR     PC,TSTDAC          ;CHECK VERNIER DAC SETTING
 866  006762  163737  001464  001454          SUB     DAC,RMS            ;SUBTRACT RESULT FROM RMS
 867  006770  012737  000001  013164          MOV     #1,EDGFLG
 868  006776  104401  021063          4$:     TYPE    ,RMSNOI
 869  007002  013702  001454                  MOV     RMS,R2
 870  007006  004737  015314                  JSR     PC,TYPRP           ;TYPE RMS VALUES
 871  007012  023737  001454  016360          CMP     RMS,VHR            ;WITHIN LIMITS?
 872  007020  003007                          BGT     6$                 ;NO
 873  007022  104401  021132                  TYPE    ,OKMSG
 874  007026  000412                          BR      7$                 ;;
 875  007030  012737  000377  001454  5$:     MOV     #255.,RMS          ;SET RMS TO MAX ERROR
 876  007036  000757                          BR      4$                 ;;
 877  007040  104401  021606          6$:     TYPE    ,ERMSG
 878  007044  004737  026564                  JSR     PC,WHICHV          ;INDICATE BAD UNIT
 879  007050  005237  001112                  INC     $ERTTL             ;UPDATE ERROR TOTAL
 880  007054  012737  007064  007770  7$:     MOV     #8$,ERRADR         ;SET UP ERROR RETRY ADDRESS
 881  007062  005005                          CLR     R5                 ;CLEAR RETRY COUNT
 882  007064  005205                  8$:     INC     R5                 ;INCREMENT COUNT
 883  007066  022705  000006                  CMP     #6,R5              ;IS COUNT = 6?
 884  007072  001450                          BEQ     10$                ;;YES, CHANNEL TOO WIDE OR NOISY
```

```
885  007074  013737  001444  001442       MOV    CHANL,DUMMY      ;LOAD DUMMY CHANNEL
886  007102  004537  015360               JSR    R5,CONVTC        ;GET EDGE VALUE
887  007106  013737  001426  001470       MOV    TEMP,EDGE        ;SET UP EDGE VALUE
888  007114  004537  013224               JSR    R5,SARSUB        ;DO SAR ROUTINE AT 1%
889  007120  000001                        1
890  007122  004737  007744               JSR    PC,TSTDAC        ;CHECK VERNIER DAC SETTING
891  007126  013737  001464  001456       MOV    DAC,PEAK         ;ADD RESULT TO PEAK
892  007134  004537  013224               JSR    R5,SARSUB        ;DO SAR ROUTINE AT 99%
893  007140  000143                        99.
894  007142  004737  007744               JSR    PC,TSTDAC        ;CHECK VERNIER DAC SETTING
895  007146  163737  001464  001456       SUB    DAC,PEAK         ;SUBTRACT RESULT FROM PEAK
896  007154  012737  000001  013164       MOV    #1,EDGFLG
897  007162  104401  021077        9$:    TYPE   ,PKNOI
898  007166  013702  001456               MOV    PEAK,R2
899  007172  004737  015314               JSR    PC,TYPRP         ;TYPE PEAK VALUES
900  007176  023737  001456  016362       CMP    PEAK,VNP         ;WITHIN LIMITS?
901  007204  003007                       BGT    11$              ;NO
902  007206  104401  021132               TYPE   ,OKMSG
903  007212  000412                       BR     12$              ;;
904  007214  012737  000377  001456  10$: MOV    #255.,PEAK       ;SET PEAK TO MAX ERROR
905  007222  000757                       BR     9$               ;;
906  007224  104401  021606       11$:    TYPE   ,ERMSG
907  007230  004737  026564               JSR    PC,WHICHV        ;INDICATE BAD UNIT
908  007234  005237  001112               INC    $ERTTL           ;UPDATE ERROR COUNT
909  007240  104401  001165       12$:    TYPE   ,$CRLF           ;LEAVE A BLANK LINE
910  007244  005237  001444               INC    CHANL            ;GET NEXT CHANNEL
911  007250  005737  001416               TST    WIDE             ;CHECK ENTRY FLAG
912  007254  001023                       BNE    15$              ;BR IF MANUAL ENTRY
913  007256  022737  000003  001444       CMP    #3,CHANL         ;CHANNEL 3 (RAMP CHANNEL)?
914  007264  001404                       BEQ    13$              ;;YES
915  007266  022737  000007  001444       CMP    #7,CHANL         ;CHANNEL 7 (EDC INPUT CHANNEL)?
916  007274  001002                       BNE    14$              ;;NO
917  007276  005237  001444       13$:    INC    CHANL            ;CHANNELS 3 AND 7 ARE SKIPED
918  007302  005737  001476       14$:    TST    WFTEST           ;RUNNING WITHOUT DWARF/TESTER ?
919  007306  001006                       BNE    15$              ;;BR IF DWARF/TESTER
920  007310  022737  000004  001444       CMP    #4,CHANL         ;DONE?
921  007316  001422                       BEQ    17$              ;;YES,GO TO NEXT TEST
922  007320  000137  006670               JMP    2$
923  007324                       15$:
(1)  007324  100406                       BMI    16$              ;;BR IF TESTER MODE
924  007326  023737  001372  001444       CMP    BASEND,CHANL     ;DONE?
925  007334  103413                       BLO    17$              ;;YES
926  007336  000137  006670               JMP    2$               ;NO, CONTINUE TESTING
927                            ;ON TESTER - DON'T RUN GOOD MNCAM
928  007342  013705  001372       16$:    MOV    BASEND,R5        ;GET LAST CHANNEL
929  007346  162705  000017               SUB    #17,R5           ;GET LAST CHANNEL TO TEST
930  007352  020537  001444               CMP    R5,CHANL         ;DONE?
931  007356  001402                       BEQ    17$              ;;YES, GO TO NEXT TEST
932  007360  000137  006670  ::           JMP    2$               ;NO, CONTINUE TESTING
933  007364  000207               17$:    RTS    PC               ;EXIT
934  007366                       NOIJMP:
```

```
  936                                 ;;*************************************************************
  (3)                                 ;*TEST 45          INTERCHANNEL SETTLING TEST, 1 EDGE
  (3)                                 ;;*************************************************************
  (2)  007366  000004        TST45:   SCOPE
  (1)  007370  012737  000001  001160          MOV     #1,$TIMES            ;;DO 1 ITERATION
  937  007376  104401  017204          TYPE    ,SETMSG              ;TYPE "SETTLING TEST"
  938  007402  004737  026572          JSR     PC,WHICHU            ;DETERMINE THE UNIT #
  939  007406  013746  001514          MOV     UNITBD,-(SP)         ;SAVE IT
  940  007412  104403                  TYPOS                        ;TYPE IT
  941  007414     001     000          .BYTE   1,0
  942  007416  104401  001165          TYPE    ,$CRLF
  943  007422  005737  001476          TST     WFTEST               ;RUNNING ON THE TESTER
  944  007426  100006                  BPL     1$                   ;;BR IF NOT
  945  007430  012700  000024          MOV     #24,R0               ;GET MUX CHANNEL INCASE TESTING MNCAM
  946  007434  022737  000057  001372          CMP     #57,BASEND           ;TESTING MNCAM?
  947  007442  001405                  BEQ     2$                   ;;NO
  948  007444  013700  001370    1$:   MOV     BASECH,R0            ;IS CHANNEL ZERO?
  949  007450  001410                  BEQ     3$                   ;;YES
  950  007452  062700  000004          ADD     #4,R0                ;SET UP CHANNELS TO SETTLE BETWEEN
  951  007456  010037  001432    2$:   MOV     R0,CH1
  952  007462  005200                  INC     R0
  953  007464  010037  001434          MOV     R0,CH2
  954  007470  000406                  BR      4$                   ;;
  955  007472  012737  000001  001432  3$:   MOV     #1,CH1               ;DO TEST BETWEEN CHANNEL 1 AND 2
  956  007500  012737  000002  001434          MOV     #2,CH2
  957  007506  005005            4$:   CLR     R5                   ;CLEAR RETRY COUNT
  958  007510  005205            5$:   INC     R5                   ;INCREMENT COUNT
  959  007512  022705  000006          CMP     #6,R5                ;IS COUNT = 6?
  960  007516  001444                  BEQ     7$                   ;;YES
  961  007520  013737  001434  001444          MOV     CH2,CHANL
  962  007526  004537  015360          JSR     R5,CONVTC            ;GET EDGE VALUES
  963  007532  013737  001426  001470          MOV     TEMP,EDGE            ;SET UP EDGE VALUE
  964  007540  005002                  CLR     R2
  965  007542  004737  013036          JSR     PC,SET1A             ;SCALING = .02 LSB
  966  007546  000760                  BR      5$                   ;ERROR RECOVERY JUMP
  967  007550  004737  013036          JSR     PC,SET1A             ;MAKE IT .01 LSB
  968  007554  000755                  BR      5$                   ;ERROR RECOVERY JUMP
  969  007556  005702                  TST     R2                   ;TEST RESULTS
  970  007560  100001                  BPL     6$
  971  007562  005402                  NEG     R2                   ;MAKE IT POSITIVE
  972  007564  010204            6$:   MOV     R2,R4
  973  007566  012737  000001  013164          MOV     #1,EDGFLG
  974  007574  004737  012704          JSR     PC,TYPSET            ;TYPE SETTLING INFORMATION
  975  007600  023737  001434  001432          CMP     CH2,CH1              ;DONE?
  976  007606  103413                  BLO     TST46                ;;YES
  977  007610  013702  001432          MOV     CH1,R2               ;SETTLE THE OTHER WAY
  978  007614  013737  001434  001432          MOV     CH2,CH1
  979  007622  010237  001434          MOV     R2,CH2
  980  007626  000727                  BR      4$                   ;;
  981  007630  012702  000377    7$:   MOV     #255.,R2                        ;SET SETTLING TO MAX ERROR
  982  007634  000753                  BR      6$                   ;;
```

D 4

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 17
CVMNAA.P11   T46   DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST (CHANNEL 3)   SEQ 0042

```
  984                                   ;;************************************************************
  (3)                                   ;*TEST 46        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST (CHANNEL 3)
  (3)                                   ;;************************************************************
  (2)  007636  000004          TST46:   SCOPE
  (1)  007640  012737  000001  001160            MOV     #1,$TIMES        ;;DO 1 ITERATION
  985  007646  005737  001370            TST     BASECH           ;TESTING CHANNEL 3?
  986  007652  001005                    BNE     LEND             ;;NO
  987  007654  005737  001176            TST     $PASS                      ;FIRST TIME-SKIP DIFLIN
  988  007660  001402                    BEQ     LEND
  989  007662  004737  013424            JSR     PC,DIFLIN
  990  007666  000207          LEND:    RTS     PC                         ;RETURN TO TEST SECTION
  991
  992                                   ;********
  993
  994  007670  012737  004001  001470 OFFSET: MOV     #4001,EDGE                 ;4000,4001 EDGE
  995  007676  004537  013224            JSR     R5,SARSUB
  996  007702  000062                    50.
  997  007704  013737  001464  001426    MOV     DAC,TEMP
  998  007712  012737  004000  001470    MOV     #4000,EDGE                 ;3777,4000 EDGE
  999  007720  004537  013224            JSR     R5,SARSUB
 1000  007724  000062                    50.
 1001  007726  063737  001464  001426    ADD     DAC,TEMP
 1002  007734  162737  000400  001426    SUB     #400,TEMP
 1003  007742  000207                    RTS     PC
 1004
 1005
 1006                                   ;************************
 1007                                   ; ROUTINE TO TEST DAC SETTING FROM SARSUB
 1008                                   ; JUMPS TO ADDRESS IN ERRADR IF DAC SETTING IS EITHER 0 OR 377
 1009                                   ; OTHERWISE RETURNS TO CALL+1
 1010  007744  005737  001464 TSTDAC: TST     DAC              ;IS DAC = 0 ?
 1011  007750  001405                    BEQ     1$               ;;YES
 1012  007752  022737  000377  001464    CMP     #377,DAC         ;IS DAC = 377 ?
 1013  007760  001401                    BEQ     1$               ;;YES
 1014  007762  000207                    RTS     PC
 1015  007764  005726          1$:      TST     (SP)+            ;POP CALL OFF STACK
 1016  007766  000137                    JMP     @(PC)+          ;JUMP TO ADDRESS IN ERRADR
 1017  007770  000000          ERRADR: 0
```

E 4

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 18
CVMNAA.P11   T46   DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST (CHANNEL 3)   SEQ 0043

```
1019                                          ;SUBROUTINE TO INSERT ''+'' AND TYPE # ON THE STACK
1020
1021   007772  013702  001426      TOFF:    MOV    TEMP,R2
1022   007776  100402                        BMI    1$              ;;IS THE NUMBER POSITIVE?
1023   010000  104401  021604                TYPE   ,POSITV
1024   010004  104416              1$:      TYPDC
1025   010006  104401  023010                TYPE   ,MLSB           ;TYPE ASCIZ STRING
1026   010012  000207                        RTS    PC
1027
1028                                          ;SUBROUTINE TO WAIT FOR OPERATOR'S 'RETURN'' THEN CHECK TOLERANCES
1029
1030   010014  005303              TCHK:    DEC    R3              ;DECREMENT COUNT
1031   010016  001005                        BNE    1$              ;;
1032   010020  012703  000005                MOV    #5,R3           ;RESET COUNT
1033   010024  104401  001165                TYPE   ,SCRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
1034   010030  000402                        BR     2$              ;;
1035   010032  104401  021014      1$:      TYPE   ,SPACE          ;TYPE FOUR (4) SPACES
1036   010036  005037  001466      2$:      CLR    DELAY           ;CLEAR DELAY
1037   010042  005077  171076                CLR    @$TKS           ;CLEAR INTERRUPT ENABLE
1038   010046  105777  171072      3$:      TSTB   @$TKS           ;IS KEYBOARD FLAG SET?
1039   010052  100404                        BMI    4$              ;;YES
1040   010054  005237  001466                INC    DELAY           ;IS DELAY ZERO?
1041   010060  001372                        BNE    3$              ;;NO
1042   010062  000416                        BR     6$              ;;
1043   010064  005777  171056      4$:      TST    @$TKB           ;CLEAR FLAG
1044   010070  012777  000100  171046        MOV    #100,@$TKS      ;SET INTERRUPT ENABLE
1045   010076  004537  015502                JSR    R5,COMPAR       ;TEST LAST CONVERSION
1046   010102  000000                        0
1047   010104  016326                        V10                    ;TOLERANCE .10 LSB
1048   010106  000402                        BR     5$              ;;
1049   010110  062716  000002                ADD    #2,(SP)         ;BUMP RETURN ADDRESS
1050   010114  062716  000002      5$:      ADD    #2,(SP)         ;BUMP RETURN ADDRESS 2 WORDS
1051   010120  000207              6$:      RTS    PC
```

```
1053                                    .SBTTL  CALIBRATION SECTION
1054  010122  104401  021143            BEGINC: TYPE    .CCHAN              ;ASK FOR CHANNEL
1055  010126  104413                            RDOCT                      ;READ CHANNEL NUMBER
1056  010130  012637  001444                    MOV     (SP)+,CHANL        ;STORE CHANNEL NUMBER
1057  010134  013737  001444  001442            MOV     CHANL,DUMMY        ;LOAD DUMMY
1058  010142  104401  021231    1$:     TYPE    .SEL               ;SELECT OFFSET OR GAIN ADJUST
1059  010146  104412                            RDLIN                      ;GET TEST
1060  010150  012600                            MOV     (SP)+,R0           ;MOVE POINTER TO R0
1061  010152  121027  000117                    CMPB    (R0),#'O           ;IS IT 'O'?
1062  010156  001406                            BEQ     AJOFF              ;;YES, GO TO ADJUST OFFSET
1063  010160  121027  000107                    CMPB    (R0),#'G           ;IS IT 'G'?
1064  010164  001430                            BEQ     AJGAIN             ;;YES, GO TO ADJUST GAIN
1065  010166  104401  001164                    TYPE    .$QUES             ;TYPE "?"
1066  010172  000763                            BR      1$                 ;;
1067
1068                                    ;SUBROUTINE TO CHECK OFFSET ADJUSTMENT VALUES
1069  010174  104401  021424            AJOFF:  TYPE    .IGND              ;GROUND CHANNEL
1070  010200  104412                            RDLIN                      ;WAIT FOR CR
1071  010202  005726                            TST     (SP)+              ;POP 1 WORD OFF STACK
1072  010204  104401  021322    1$:     TYPE    .XADJ              ;ADJUST MESSAGE
1073  010210  012703  000005                    MOV     #5,R3              ;SET UP COUNT
1074  010214  004737  007670    2$:     JSR     PC,OFFSET          ;TEST AND TYPE OFFSET ERROR
1075  010220  004737  007772                    JSR     PC,TOFF            ;TYPE OFFSET
1076  010224  004737  010014                    JSR     PC,TCHK            ;CHECK FOR A CHARACTER AND DELAY
1077  010230  000771                            BR      2$                 ;;
1078  010232  000402                            BR      3$                 ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1079  010234  000137  001564                    JMP     BEG2
1080  010240  104401  021606    3$:     TYPE    .ERMSG             ;TELL OPER. 'ERROR'
1081  010244  000757                            BR      1$
1082                                    ;SUBROUTINE TO CHECK THE GAIN ADJUSTMENT
1083  010246  104401  021523            AJGAIN: TYPE    .IVOLT             ;INPUT +5.115 VOLTS ON CHANNEL
1084  010252  104401  021463                    TYPE    .CRWR
1085  010256  104412                            RDLIN                      ;WAIT FOR CR
1086  010260  005726                            TST     (SP)+              ;POP 1 WORD OFF STACK
1087  010262  104401  021567    1$:     TYPE    .YADJ              ;ADJUST MESSAGE
1088  010266  104401  021336                    TYPE    .MOLSB             ;TYPE " FOR 0.00 LSB ERROR"
1089  010272  012703  000005                    MOV     #5,R3              ;SET UP COUNT
1090  010276  012737  007777  001470  2$:  MOV  #7777,EDGE          ;LOOK FOR 7776,7777 EDGE
1091  010304  004537  013224                    JSR     R5,SARSUB
1092  010310  000062                            50.
1093  010312  013737  001464  001426            MOV     DAC,TEMP           ;SAVE DAC
1094  010320  012737  007776  001470            MOV     #7776,EDGE         ;LOOK FOR 7775,7776 EDGE
1095  010326  004537  013224                    JSR     R5,SARSUB
1096  010332  000062                            50.
1097  010334  063737  001464  001426            ADD     DAC,TEMP           ;ADD RESULTS
1098  010342  162737  000400  001426            SUB     #400,TEMP          ;OFFSET RESULT
1099  010350  004737  007772                    JSR     PC,TOFF            ;TYPE GAIN
1100  010354  004737  010014                    JSR     PC,TCHK            ;CHECK FOR CHARACTER AND DELAY
1101  010360  000746                            BR      2$                 ;;
1102  010362  000402                            BR      3$                 ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1103  010364  000137  001564                    JMP     BEG2
1104  010370  104401  021606    3$:     TYPE    .ERMSG             ;TELL OPER. 'ERROR'
1105  010374  000732                            BR      1$
```

```
1107                                                 .SBTTL    SWITCH GAIN MANUAL INTERVENTION TEST
1108  010376  104401  021143          BEGINM: TYPE   ,CCHAN             ;ASK FOR CHANNEL
1109  010402  104413                          RDOCT                     ;READ CHANNEL NUMBER
1110  010404  012600                          MOV    (SP)+,R0           ;GET CHANNEL NUMBER
1111  010406  000300                          SWAB   R0                 ;PUT CHANNEL NUMBER IN HIGH BYTE
1112  010410  052700  000010                  BIS    #BIT3,R0           ;SET STATUS ENABLE BIT
1113  010414  010077  170732                  MOV    R0,@STREG          ;LOAD CHANNEL AND STATUS ENABLE
1114  010420  104401  020551                  TYPE   ,SCM               ;ASK MODE BE SET TO CURRENT
1115  010424  012737  030000  001124          MOV    #BIT13!BIT12,$GDDAT     ;SET UP EXPECTED
1116  010432  104401  020656          1$:     TYPE   ,GHLF              ;ASK GAIN BE SET TO .5
1117  010436  104417                          TESTID                    ;GO TEST FOR ID CODE
1118  010440  104001                          ERROR  1
1119  010442  104401  020673                  TYPE   ,GAIN5             ;ASK GAIN BE SET TO 5
1120  010446  104417                          TESTID                    ;GO TEST ID CODE
1121  010450  104001                          ERROR  1
1122  010452  104401  020713                  TYPE   ,GAIN50            ;ASK GAIN BE SET TO 50
1123  010456  104417                          TESTID                    ;GO TEST ID CODE
1124  010460  104001                          ERROR  1
1125  010462  104401  020734                  TYPE   ,GAIN5M            ;ASK GAIN BE SET TO 500
1126  010466  104417                          TESTID                    ;GO TEST ID CODE
1127  010470  104001                          ERROR  1
1128  010472  022737  070000  001124          CMP    #70000,$GDDAT      ;READY TO DO RESISTANCE?
1129  010500  001003                          BNE    2$                 ;;NO
1130  010502  104401  020577                  TYPE   ,SRM               ;ASK MODE BE SET TO RESISTANCE
1131  010506  000751                          BR     1$
1132  010510  022737  130000  001124  2$:     CMP    #130000,$GDDAT     ;READY TO DO VOLTS?
1133  010516  001003                          BNE    3$                 ;;NO, DONE WITH TEST
1134  010520  104401  020630                  TYPE   ,SVM               ;ASK MODE BE SET TO VOLTS
1135  010524  000742                          BR     1$
1136  010526  000137  001564          3$:     JMP    BEG2
1137
1138  010532  062737  010000  001124  TPRMP:  ADD    #BIT12,$GDDAT      ;INDEX EXPECTED ID
1139  010540  104401  021463                  TYPE   ,CRWR              ;ASK FOR CR WHEN READY
1140  010544  104412                          RDLIN                     ;WAIT FOR CR
1141  010546  005726                          TST    (SP)+              ;POP 1 WORD OFF STACK
1142  010550  005277  170576                  INC    @STREG             ;START A CONVERSION
1143  010554  105777  170572          1$:     TSTB   @STREG             ;WAIT TILL DONE
1144  010560  100375                          BPL    1$                 ;
1145  010562  017737  170570  001126          MOV    @ADBUFF,$BDDAT     ;GET RESULTS
1146  010570  042737  007777  001126          BIC    #7777,$BDDAT       ;CLEAR CONVERTED VALUE
1147  010576  023737  001124  001126          CMP    $GDDAT,$BDDAT      ;IS ID RIGHT?
1148  010604  001002                          BNE    2$                 ;;NO,TAKE ERROR RETURN
1149  010606  062716  000002                  ADD    #2,(SP)            ;BUMP RETURN ADDRESS
1150  010612  000002          2$:     RTI
```

```
1152                                    .SBTTL          PRINT VALUES ROUTINE
1153   010614  005077  170532   BEGINP: CLR     @STREG                  ;CLEAR STATUS REGISTER
1154   010620  104401  022642           TYPE    ,HEAD5                  ;TYPE OUT HEADING
1155   010624  005046                   CLR     -(SP)                   ;CLEAR PSW
1156   010626  012746  010634           MOV     #1$,-(SP)
1157   010632  000002                   RTI
1158   010634  017700  170300   1$:     MOV     @SWR,R0                 ;READ CHANNEL FROM SWITCH REG.
1159   010640  042700  177700           BIC     #177700,R0             ;ISOLATE MUX BITS
1160   010644  032777  020000  170266   BIT     #BIT13,@SWR            ;IS BIT 13 SET?
1161   010652  001005                   BNE     2$                     ;;YES,SKIP TYPEOUT
1162   010654  104401  021011           TYPE    ,CH
1163   010660  010046                   MOV     R0,-(SP)               ;;SAVE R0 FOR TYPEOUT
  (1)                                                                   ;;TYPE CHANNEL
  (1)   010662  104403                   TYPOS                          ;;GO TYPE--OCTAL ASCII
  (1)   010664     002                   .BYTE   2                      ;;TYPE 2 DIGIT(S)
  (1)   010665     000                   .BYTE   0                      ;;SUPPRESS LEADING ZEROS
1164   010666  012777  001540  170464   2$:     MOV     #RETURN,@VECTOR        ;ADDRESS AFTER INTRPT.
1165   010674  000300                   SWAB    R0                     ;SWITCH BYTES
1166   010676  052700  000100           BIS     #BIT6,R0
1167   010702  010077  170444           MOV     R0,@STREG              ;LOAD THE CHANNEL
1168   010706  012702  000010           MOV     #10,R2                 ;TYPEOUT COUNTER
1169   010712  005277  170434   3$:     INC     @STREG                 ;START CONVERSION
1170   010716  000001                   WAIT                           ;WAIT FOR INTRPT.
1171   010720  017700  170432           MOV     @ADBUFF,R0             ;READ CONVERTED VALUE
1172   010724  032777  020000  170206   BIT     #BIT13,@SWR            ;IS BIT 13 SET?
1173   010732  001403                   BEQ     4$                     ;NOT SET, TYPE OUT LIST
1174   010734  010077  170202           MOV     R0,@DISPLAY            ;PUT VALUE IN DISPLAY FOR DISPLAY CONTROL
1175   010740  000735                   BR      1$                     ;REPEAT CONVERSION
1176   010742  104401  021014   4$:     TYPE    ,SPACE
1177   010746  010046                   MOV     R0,-(SP)               ;;SAVE R0 FOR TYPEOUT
  (1)                                                                   ;;PRINT OCTAL CONVERTED VALUE
  (1)   010750  104403                   TYPOS                          ;;GO TYPE--OCTAL ASCII
  (1)   010752     004                   .BYTE   4                      ;;TYPE 4 DIGIT(S)
  (1)   010753     001                   .BYTE   1                      ;;TYPE LEADING ZEROS
1178   010754  012701  010000           MOV     #10000,R1
1179   010760  005301           5$:     DEC     R1
1180   010762  001376                   BNE     5$
1181   010764  005302                   DEC     R2                     ;DECREMENT THE COUNTER
1182   010766  001351                   BNE     3$                     ;NO CARRIAGE RETURN
1183   010770  104401  001165           TYPE    ,$CRLF                 ;CARRIAGE RETURN
1184   010774  000717                   BR      1$                     ;REPEAT CONVERSION
```

```
1186                                       .SBTTL        LOGIC TEST SECTION START-UP
1187   010776  004737  011254              BEGL:  JSR    PC,WFCHK        ;CHECK I D CODE IF WESTFIELD MODE
1188   011002  012737  011010  016610             MOV    #2$,AGTST       ;LOAD EOP RETURN IF NO A/D
1189   011010  004737  002772              2$:    JSR    PC,TESTAD       ;SIZE THE NUMBER OF MNCAD'S
1190   011014  004737  003266              1$:    JSR    PC,BEGINL       ;LOGIC TESTS
1191   011020  004737  012410                     JSR    PC,BUMPAD       ;MORE TO TEST?
1192   011024  000773                             BR     1$              ;TEST NEXT A/D
1193   011026  012737  011014  016610             MOV    #1$,AGTST       ;ADDRESS FOR EOP
1194   011034  000137  016412                     JMP    $EOP            ;TYPE END OF PASS
1195
1196                                       .SBTTL        AUTO TEST START-UP
1197   011040  004737  002772              BEGINA: JSR   PC,TESTAD       ;SIZE THE # OF MNCAD'S
1198   011044  004737  011254                     JSR    PC,WFCHK        ;SET UP IF WESTFIELD MODE
1199   011050  004737  003266              1$:    JSR    PC,BEGINL       ;LOGIC TESTS
1200   011054  004737  012410                     JSR    PC,BUMPAD       ;UPDATE THE ADDRESSES IF ANY
1201   011060  000773                             BR     1$              ;BR AND DO NEXT UNIT
1202   011062  104401  022045                     TYPE   ,MEND           ;TELL OPER. END OF LOGIC TEST
1203   011066  004737  011446              2$:    JSR    PC,TCHANL       ;REPORT A/D CONFIG. AND IF DWARF MODE
1204                                                                     ;ASK FOR THE CHANNELS TO TEST
1205   011072  004737  005406                     JSR    PC,WRAP
1206   011076  004737  012410                     JSR    PC,BUMPAD       ;BUMP THE ADDRESSES
1207   011102  000771                             BR     2$              ;BR AND DO NEXT UNIT
1208   011104  012737  011050  016610             MOV    #1$,AGTST       ;ADDRESS FOR EOP
1209   011112  000137  016412                     JMP    $EOP            ;TYPE END OF PASS
1210
1211                                       .SBTTL        WRAPAROUND TEST START-UP
1212   011116  004737  002772              BEGINW: JSR   PC,TESTAD       ;SIZE THE # OF MNCAD'S
1213   011122  004737  011254                     JSR    PC,WFCHK        ;SET UP IF WESTFIELD MODE
1214   011126  004737  011446              1$:    JSR    PC,TCHANL       ;REPORT THE A/D CONFIG. AND IF DWARF
1215                                                                     ;MODE ASK FOR THE CHANNELS TO TEST
1216   011132  004737  005406                     JSR    PC,WRAP         ;WRAPAROUND TESTS
1217   011136  004737  012410                     JSR    PC,BUMPAD       ;UPDATE BUS ADDRESSES
1218   011142  000771                             BR     1$              ;BR AND TEST NEXT UNIT
1219   011144  012737  011126  016610             MOV    #1$,AGTST
1220   011152  000137  016412                     JMP    $EOP            ;INCREMENTS $PASS
1221
1222                                       .SBTTL        NOISE TEST START-UP
1223   011156  004737  012510              BEGINN: JSR   PC,FIXONE       ;ENSURE BASE AND VECTOR SETUP
1224   011162  005037  001440                     CLR    NMBEXT          ;CLEAR MULTIPLE UNIT FLAG
1225   011166  104401  017076                     TYPE   ,SCHAN          ;ASK FOR STARTING NOISE CHANNEL
1226   011172  104413                             RDOCT                  ;GET OPER. CHANNEL INPUT
1227   011174  012637  001370                     MOV    (SP)+,BASECH    ;SAVE 1ST CHANNEL
1228   011200  104401  017126                     TYPE   ,ECHAN          ;ASK FOR END NOISE CHANNEL
1229   011204  104413                             RDOCT                  ;GET OPER. CHANNEL INPUT
1230   011206  012637  001372                     MOV    (SP)+,BASEND    ;SAVE LAST CHANNEL
1231   011212  001006                             BNE    1$              ;BR IF NON-ZERO
1232   011214  013737  001370  001372             MOV    BASECH,BASEND   ;TAKE CARE IF ONLY 1 CHANNEL
1233   011222  000240                             NOP
1234   011224  000240                             NOP
1235   011226  000240                             NOP
1236   011230  012737  000001  001416     1$:    MOV    #1,WIDE         ;SET MANUAL ENTRY FLAG
1237   011236  004737  006612                     JSR    PC,NOITST       ;RUN NOISE TEST
1238   011242  012737  011230  016610             MOV    #1$,AGTST       ;LOAD RETRURN POINTER
1239   011250  000137  016412                     JMP    $EOP            ;AND REPORT END OF PASS
```

```
1241                                      ;*ROUTINE TO CHECK FOR PROPER I D CODE IF TESTER MODE
1242                                      ;*IF ON TESTER, SET UP BASECH AND BASEND FOR CHANNELS BEING TESTED
1243
1244  011254  005737  001476      WFCHK:  TST    WFTEST           ;RUNNING  ON TESTER?
1245  011260  100063                      BPL    5$               ;;BR IF NOT
1246  011262  005037  001370              CLR    BASECH           ;CLEAR STARTING CHANNEL
1247  011266  012777  020000  170056      MOV    #20000,@STREG    ;IS CHANNEL 40 PRESENT?
1248  011274  032777  000002  170050      BIT    #BIT1,@STREG     ;IS THE NON-EXSISTENT CHANNEL BIT SET?
1249  011302  001414                      BEQ    1$               ;;NO, TESTING A/D AND AM
1250  011304  012737  000037  001372      MOV    #37,BASEND       ;SET UP LAST CHANNEL TO TEST
1251  011312  104401  020000              TYPE   ,TSTAD           ;TYPE TESTING A/D MESSAGE
1252  011316  017700  170072              MOV    @DRVDIR,R0       ;GET I D BITS
1253  011322  042700  177417              BIC    #177417,R0       ;CLEAR UNWANTED BITS
1254  011326  022700  000060              CMP    #60,R0           ;IS THE I D CODE CORRECT?
1255  011332  000413                      BR     2$               ;;
1256  011334  012737  000057  001372  1$: MOV    #57,BASEND       ;SET UP LAST CHANNEL TO TEST
1257  011342  104401  020022              TYPE   ,TSTADM          ;TYPE TESTING A/D AND AM MESSAGE
1258  011346  017700  170042              MOV    @DRVDIR,R0       ;GET I D BITS
1259  011352  042700  177417              BIC    #177417,R0       ;CLEAR UNWANTED BITS
1260  011356  022700  000340              CMP    #340,R0          ;IS THE I D CODE CORRECT?
1261  011362                          2$:
 (1)  011362  001422                      BEQ    5$               ;;RETURN
1262  011364  104401  020451              TYPE   ,BADID           ;TYPE BAD I D CODE MESSAGE
1263  011370  104401  020476          3$: TYPE   ,YORNO           ;TYPE CONTINUE TESTING MESSAGE
1264  011374  104412                      RDLIN                   ;GET RESPONSE
1265  011376  052777  000100  167540      BIS    #100,@$TKS       ;ENABLE KEYBOARD INTERRUPTS
1266  011404  005046                      CLR    -(SP)            ;CLEAR PSW
1267  011406  012746  011414              MOV    #4$,-(SP)
1268  011412  000002                      RTI
1269  011414  012600              4$:     MOV    (SP)+,R0         ;READ ANSWER
1270  011416  142710  000040              BICB   #40,(R0)         ;CONVERT OT UPPER CASE
1271  011422  121027  000131              CMPB   (R0),#'Y         ;IS IT Y?
1272  011426  001001                      BNE    6$               ;;NO, CHECK FOR 'N'
1273  011430  000207              5$:     RTS    PC               ;RETURN
1274  011432  121027  000116      6$:     CMPB   (R0),#'N         ;IS IT N?
1275  011436  001354                      BNE    3$               ;;NO, ASK AGAIN
1276  011440  000000                      HALT
1277  011442  000137  001564              JMP    @#BEG2           ;RESTART IF CONTINUED
1278
```

```
1280                                  ;*ROUTINE TO TYPE OUT A/D CONFIGURATION
1281                                  ;*IF RUNNING IN TEST MODULE MODE, ASK FOR CHANNELS TO TEST
1282
1283  011446  005037  001370   TCHANL: CLR    BASECH          ;CLEAR FIRST CHANNEL TO TEST
1284  011452  005037  001372          CLR    BASEND          ;CLEAR LAST CHANNEL TO TEST
1285  011456  104401  001165          TYPE   ,$CRLF          ;FRESH LINE
1286  011462  005737  001176          TST    $PASS           ;TEST IF FIRST PASS
1287  011466  001017                  BNE    22$             ;BR IF NOT
1288  011470  005737  001476          TST    WFTEST          ;TEST WESTFIELD FLAG
1289  011474  001406                  BEQ    1$              ;;RUNNING WITH NO TEST MODULE/TESTER
1290  011476  100003                  BPL    21$             ;;RUNNING IN TEST MODULE MODE BUT NO TESTER
1291  011500  104401  020171          TYPE   ,SDDIF          ;TYPE SET DWARF TO DIFFERENTIAL MESSAGE
1292  011504  000402                  BR     1$              ;;
1293  011506  104401  020125   21$:   TYPE   ,SDSE           ;TYPE SET DWARF TO SINGLE ENDED MESSAGE
1294  011512  104401  020044   1$:    TYPE   ,SADTST         ;TYPE SET A/D TO TEST MESSAGE
1295  011516  104401  021463          TYPE   ,CRWR           ;TYPE CARRIAGE RETURN WHEN READY MESSAGE
1296  011522  104412                  RDLIN                  ;WAIT FOR CARRIAGE RETURN
1297  011524  005726                  TST    (SP)+           ;POP 1 WORD OFF STACK
1298  011526  104401  017524   22$:   TYPE   ,VTMSG          ;REPORT UNIT #
1299  011532  004737  026572          JSR    PC,WHICHU       ;DETERMINE ASCII UNIT #
1300  011536  013746  001514          MOV    UNITBD,-(SP)
1301  011542  104403                  TYPOS
1302  011544    001     000           .BYTE  1,0
1303  011546  104401  001165          TYPE   ,$CRLF          ;LEAVE A BLANK LINE
1304  011552  005001                  CLR    R1              ;SET UP STARTING CHANNEL
1305  011554  005000                  CLR    R0              ;SET UP FIRST I.D. STATUS
1306  011556  000407                  BR     4$              ;;GO TYPE RESULTS
1307  011560  005277  167566   2$:    INC    @STREG          ;START A CONVERSION
1308  011564  105777  167562   3$:    TSTB   @STREG          ;WAIT FOR CONVERSION TO FINISH
1309  011570  100375                  BPL    3$
1310  011572  017700  167560          MOV    @ADBUFF,R0      ;GET RESULTS
1311  011576                   4$:
 (1)  011576  010146                  MOV    R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
 (1)  011600  104403                  TYPOS                  ;;GO TYPE--OCTAL ASCII
 (1)  011602    002                   .BYTE  2               ;;TYPE 2 DIGIT(S)
 (1)  011603    000                   .BYTE  0               ;;SUPPRESS LEADING ZEROS
1312  011604  104401  017347          TYPE   ,MDASH          ;TYPE A DASH
1313  011610  062701  000003          ADD    #3,R1           ;ADD 3 TO CHANNEL FOR DIFFERENTIAL
1314  011614  042700  007777          BIC    #7777,R0        ;IS CHANNEL SINGLE ENDED
1315  011620  001002                  BNE    5$              ;;CHANNEL IS NOT SINGLE ENDED
1316  011622  062701  000004          ADD    #4,R1           ;ADD 4 CHANNELS FOR SINGLE ENDED
1317  011626  022701  000100   5$:    CMP    #100,R1         ;IS CHANNEL > LAST POSSIBLE CHANNEL
1318  011632  101002                  BHI    6$              ;;NO
1319  011634  012701  000077          MOV    #77,R1          ;YES, SET TO LAST CHANNEL
1320  011640                   6$:
 (1)  011640  010146                  MOV    R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
 (1)  011642  104403                  TYPOS                  ;;GO TYPE--OCTAL ASCII
 (1)  011644    002                   .BYTE  2               ;;TYPE 2 DIGIT(S)
 (1)  011645    000                   .BYTE  0               ;;SUPPRESS LEADING ZEROS
1321  011646  005700                  TST    R0              ;IS CHANNEL SINGLE ENDED?
1322  011650  001003                  BNE    7$              ;;NO
1323  011652  104401  017353          TYPE   ,MSE            ;TYPE SINGLE ENDED MESSAGE
1324  011656  000410                  BR     9$              ;;GO TEST MORE CHANNELS
1325  011660  032700  140000   7$:    BIT    #BIT15!BIT14,R0 ;DOES CHANNEL HAVE PREAMP?
```

```
1326  011664  001003                      BNE    8$            ;:YES, HAS PREAMP
1327  011666  104401  017373              TYPE   ,MDIF         ;TYPE DIFFERENTIAL MESSAGE
1328  011672  000402                      BR     9$            ;:GO TEST MORE CHANNELS
1329  011674  104401  017413      8$:     TYPE   ,MPRMP        ;TYPE PREAMP MESSAGE
1330  011700  005201              9$:     INC    R1            ;SET CHANNEL TO NEXT SET OF CHANNELS
1331  011702  022701  000100              CMP    #100,R1       ;DONE?
1332  011706  001414                      BEQ    10$           ;:YES
1333  011710  010100                      MOV    R1,R0         ;GET CHANNEL
1334  011712  000300                      SWAB   R0            ;PUT CHANNEL NUMBER IN HIGH BYTE
1335  011714  052700  000010              BIS    #BIT³,R0      ;SET STATUS ENABLE BIT
1336  011720  010077  167426              MOV    R0,aSTREG     ;LOAD INTO A/D STATUS REGISTER
1337  011724  032777  000002  167420      BIT    #BIT1,aSTREG  ;IS NON-EXSISTENT CHANNEL BIT SET?
1338  011732  001712                      BEQ    2$            ;:NO
1339  011734  104401  001165              TYPE   ,$CRLF
1340                                ;IF USING TEST MODULE OR TESTER MODE, DO MORE TESTING
1341                                ;IF NOT THEN EXIT
1342  011740  022737  000001  001476  10$: CMP   #1,WFTEST     ;RUNNING DWARF MODE?
1343  011746  001117                      BNE    20$           ;:NO
1344  011750  005001                      CLR    R1            ;SET UP TO ASK FOR FIRST GROUP
1345  011752  004737  012210      11$:    JSR    PC,ASKC       ;ASK TO TEST CHANNELS
1346  011756  000434                      BR     14$           ;:YES
1347  011760  062701  000010              ADD    #10,R1        ;INDEX TO NEXT CHANNEL BANK
1348  011764  010100              12$:    MOV    R1,R0         ;PUT CHANNEL INTO R0
1349  011766  022700  000100              CMP    #100,R0       ;ANY MORE CHANNELS?
1350  011772  001762                      BEQ    10$           ;:NO
1351  011774  000300                      SWAB   R0            ;PUT CHANNEL IN HIGH BYTE
1352  011776  052700  000010              BIS    #BIT3,R0      ;SET STATUS ENABLE BIT
1353  012002  010077  167344              MOV    R0,aSTREG     ;LOAD INTO A/D STATUS REGISTER
1354  012006  032777  000002  167336      BIT    #BIT1,aSTREG  ;IS THE NON-EXSISTENT CHANNEL BIT SET?
1355  012014  001351                      BNE    10$           ;:YES
1356  012016  005277  167330              INC    aSTREG        ;START A CONVERSION
1357  012022  105777  167324      13$:    TSTB   aSTREG        ;WAIT FOR CONVERSION TO FINISH
1358  012026  100375                      BPL    13$           ;:
1359  012030  017700  167322              MOV    aADBUFF,R0    ;GET RESULTS
1360  012034  042700  007777              BIC    #7777,R0      ;IS CHANNEL SINGLE ENDED?
1361  012040  001744                      BEQ    11$           ;:YES
1362  012042  062701  000004              ADD    #4,R1         ;INDEX TO NEXT CHANNEL BANK
1363  012046  000746                      BR     12$           ;:
1364  012050  010137  001370      14$:    MOV    R1,BASECH     ;SAVE FIRST CHANNEL TO TEST
1365  012054  062701  000010      15$:    ADD    #10,R1        ;INDEX TO NEXT BANK
1366  012060  010100              16$:    MOV    R1,R0         ;PUT CHANNEL INTO R0
1367  012062  022700  000100              CMP    #100,R0       ;ANY MORE BANKS?
1368  012066  001426                      BEQ    19$           ;:NO
1369  012070  000300                      SWAB   R0            ;PUT CHANNEL INTO HIGH BYTE
1370  012072  052700  000010              BIS    #BIT3,R0      ;SET STATUS ENABLE BIT
1371  012076  010077  167250              MOV    R0,aSTREG     ;LOAD INTO A/D STATUS REGISTER
1372  012102  032777  000002  167242      BIT    #BIT1,aSTREG  ;IS THE NON-EXSISTENT CHANNEL BIT SET?
1373  012110  001015                      BNE    19$           ;:YES
1374  012112  005277  167234              INC    aSTREG        ;START A CONVERSION
1375  012116  105777  167230      17$:    TSTB   aSTREG        ;WAIT FOR CONVERSION TO FINISH
1376  012122  100375                      BPL    17$           ;:
1377  012124  017700  167226              MOV    aADBUFF,R0    ;GET RESULTS
1378  012130  042700  007777              BIC    #7777,R0      ;IS CHANNEL SINGLE ENDED?
1379  012134  001003                      BNE    19$           ;:BR IF NOT
```

```
1380  012136  004737  012210     18$:   JSR     PC,ASKC         ;ASK TO TEST CHANNELS
1381  012142  000744                     BR      15$             ;;
1382  012144  005301             19$:   DEC     R1              ;DECREMENT CHANNEL
1383  012146  010137  001372            MOV     R1,BASEND       ;SAVE LAST CHANNEL TO TEST
1384  012152  104401  017721            TYPE    ,TCHAN          ;TYPE "TESTING CHANNELS "
1385  012156  013746  001370            MOV     BASECH,-(SP)    ;;SAVE BASECH FOR TYPEOUT
 (1)  012162  104403                     TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  012164     002                     .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  012165     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
1386  012166  104401  017347            TYPE    ,MDASH          ;TYPE " - "
1387  012172  013746  001372            MOV     BASEND,-(SP)    ;;SAVE BASEND FOR TYPEOUT
 (1)  012176  104403                     TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  012200     002                     .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  012201     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
1388  012202  104401  001165            TYPE    ,SCRLF          ;TYPE A CARRIDGE RETURN, LINE FEED
1389  012206  000207             20$:   RTS     PC              ;RETURN
1390
1391                             ;*ROUTINE TO ASK CHANNELS TO TEST
1392
1393  012210  104401  017721     ASKC:  TYPE    ,TCHAN          ;TYPE "TEST CHANNELS "
1394  012214  010146                     MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
 (1)  012216  104403                     TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  012220     002                     .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  012221     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
1395  012222  104401  017347            TYPE    ,MDASH          ;TYPE " - "
1396  012226  010100                     MOV     R1,R0           ;PUT CHANNEL INTO R0
1397  012230  062700  000007            ADD     #7,R0           ;GET LAST CHANNEL IN GROUP
1398  012234  010046                     MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
 (1)  012236  104403                     TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  012240     002                     .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  012241     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
1399  012242  104401  017345            TYPE    ,QUEST          ;TYPE " ? "
1400  012246  104412                     RDLIN                   ;GET RESPONSE
1401  012250  012600                     MOV     (SP)+,R0        ;GET ADDRESS OF RESPONSE TEXT
1402  012252  142710  000040            BICB    #40,(R0)        ;MAKE CHARACTER UPPER CASE
1403  012256  122710  000131            CMPB    #'Y,(R0)        ;IS IT A Y?
1404  012262  001410                     BEQ     2$              ;;YES
1405  012264  122710  000116            CMPB    #'N,(R0)        ;IS IT AN N?
1406  012270  001403                     BEQ     1$              ;;YES
1407  012272  104401  017745            TYPE    ,YESNO          ;TYPE "TYPE Y FOR YES, N FOR NO"
1408  012276  000744                     BR      ASKC            ;;
1409  012300  062716  000002     1$:    ADD     #2,(SP)         ;SKIP OVER BRANCH
1410  012304  000207             2$:    RTS     PC              ;RETURN
```

```
1412
1413                                      ;SUBROUTINE TO CHANGE BASE AND VECTOR ADDRESSES
1414   012306  104401  017425   BASEXC: TYPE    ,MADR                    ;ASK FOR MODULE ADDRESS
1415   012312  013746  001244           MOV     $BASE,-(SP)             ;;SAVE $BASE FOR TYPEOUT
 (1)   012316  104402                    TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1416   012320  104401  017517           TYPE    ,ENCOM
1417   012324  104413                    RDOCT
1418   012326  005726                    TST     (SP)+                    ;DEFAULT ADDRESS ?
1419   012330  001403                    BEQ     5$                       ;NO BRANCH
1420   012332  016637  177776  001244    MOV     -2(SP),$BASE             ;SAVE ADDRESS IN $BASE
1421   012340  104401  017461   5$:     TYPE    ,MVCT                    ;ASK FOR MODULE VECTOR
1422   012344  013701  001240           MOV     $VECT1,R1                ;GET VECTOR
1423   012350  010146                    MOV     R1,-(SP)                ;;SAVE R1 FOR TYPEOUT
 (1)   012352  104403                    TYPOS                           ;;GO TYPE--OCTAL ASCII
 (1)   012354    003                     .BYTE   3                       ;;TYPE 3 DIGIT(S)
 (1)   012355    001                     .BYTE   1                       ;;TYPE LEADING ZEROS
1424   012356  104401  017517           TYPE    ,ENCOM
1425   012362  104413                    RDOCT
1426   012364  005726                    TST     (SP)+                    ;TAKE DEFAULT ?
1427   012366  001403                    BEQ     7$
1428   012370  016637  177776  001240    MOV     -2(SP),$VECT1
1429   012376  052737  100000  001240 7$: BIS    #BIT15,$VECT1            ;SET PRIORITY LEVEL
1430   012404  000137  002536           JMP     MTEST1                   ;RESTART
```

```
1432                                    .SBTTL       DETERMINE IF MORE MNCAD'S TO BE TESTED
1433   012410  005737  001436    BUMPAD: TST    NBEXT                    ;ADDITIONAL AD'S?
1434   012414  001433                    BEQ    FIXADR                   ;NO-INITIALIZE ADDRESSES
1435   012416  006337  001512            ASL    MASKNM           ;MOVE BIT TO NEXT MODULE
1436   012422  005001                    CLR    R1
1437   012424  013700  001512            MOV    MASKNM,R0                ;GET MASK NUMBER
1438   012430  006200            1$:     ASR    R0                       ;MOVE RIGHT
1439   012432  001403                    BEQ    2$                       ;BR IF DONE
1440   012434  062701  000004            ADD    #4,R1                    ;UPDATE INDEX VALUE
1441   012440  000773                    BR     1$
1442   012442  016137  001332  001352  2$: MOV   MNCAD0(R1),STREG        ;GET NEW ADDRESS
1443   012450  062701  000002            ADD    #2,R1                    ;NEW NEXT INDEX
1444   012454  016137  001332  001360    MOV    MNCAD0(R1),VECTOR        ;GET NEW VECTOR
1445   012462  013737  001352  001354    MOV    STREG,ADST1              ;PRIME OTHER ADDRESSES
1446   012470  013737  001352  001356    MOV    STREG,ADBUFF
1447   012476  005337  001436            DEC    NBEXT                    ;ONE LESS MNCAD
1448   012502  000427                    BR     BYPASS
1449   012504  062716  000002    FIXADR: ADD    #2,(SP)
1450   012510  012737  016612  000004  FIXONE: MOV  #IOTRD,@#ERRVEC ;SET UP ERRVEC
1451   012516  012737  000001  001512    MOV    #1,MASKNM                ;INIT. MODULE ERROR TEST BIT
1452   012524  013737  001244  001352    MOV    $BASE,STREG              ;RELOAD INITIAL ADDRESSES
1453   012532  013737  001244  001354    MOV    $BASE,ADST1
1454   012540  013737  001244  001356    MOV    $BASE,ADBUFF
1455   012546  013737  001240  001360    MOV    $VECT1,VECTOR            ;GET DEFAULT VECTOR
1456   012554  013737  001440  001436    MOV    NMBEXT,NBEXT             ;RESET UNIT COUNTER
1457   012562  005237  001354    BYPASS: INC    ADST1
1458   012566  062737  000002  001356    ADD    #2,ADBUFF
1459   012574  042737  170000  001360    BIC    #170000,VECTOR
1460   012602  013737  001360  001362    MOV    VECTOR,VECTR1
1461   012610  062737  000002  001362    ADD    #2,VECTR1
1462   012616  013737  001360  001364    MOV    VECTOR,VECTR2
1463   012624  062737  000004  001364    ADD    #4,VECTR2
1464   012632  013737  001360  001366    MOV    VECTOR,VECTR3
1465   012640  062737  000006  001366    ADD    #6,VECTR3
1466                              ;;LOAD .+2 AND JSR PC,R0 TRAP CATCHER;;
1467   012646  012700  000222            MOV    #222,R0                  ;FILL .+2
1468   012652  012701  000220            MOV    #220,R1                  ;LOAD JSR PC,R0
1469   012656  010021            1$:     MOV    R0,(R1)+
1470   012660  012721  004700            MOV    #4700,(R1)+
1471   012664  010100                    MOV    R1,R0
1472   012666  005720                    TST    (R0)+
1473   012670  020027  001002            CMP    R0,#1002
1474   012676  001370                    BNE    1$
1475   012676  004737  026572            JSR    PC,WHICHU        ;DETERMINE UNIT #
1476   012702  000207                    RTS    PC                       ;TEST NEXT A/D
```

```
1478   012704   104416                          TYPSET: TYPDC
1479   012706   104401   021021                         TYPE    ,LSB
1480   012712   013746   001434                         MOV     CH2,-(SP)               ;;SAVE CH2 FOR TYPEOUT
 (1)                                                                                     ;;TYPE CH
 (1)   012716   104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
 (1)   012720   002                                     .BYTE   2                       ;;TYPE 2 DIGIT(S)
 (1)   012721   000                                     .BYTE   0                       ;;SUPPRESS LEADING ZEROS
1481   012722   104401   021056                         TYPE    ,ATMSG                  ;TYPE ASCIZ STRING
1482   012726   004737   013122                         JSR     PC,TYPEDG
1483   012732   104401   021034                         TYPE    ,SETCH
1484   012736   013746   001432                         MOV     CH1,-(SP)               ;;SAVE CH1 FOR TYPEOUT
 (1)                                                                                     ;;TYPE CH
 (1)   012742   104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
 (1)   012744   002                                     .BYTE   2                       ;;TYPE 2 DIGIT(S)
 (1)   012745   000                                     .BYTE   0                       ;;SUPPRESS LEADING ZEROS
1485   012746   104401   021056                         TYPE    ,ATMSG
1486   012752   013737   001432   012772                MOV     CH1,1$
1487   012760   012777   000200   166370                MOV     #200,@ADBUFF
1488   012766   004537   015352                         JSR     R5,CONVRT
1489   012772   000000                          1$:     0
1490   012774   013746   001426                         MOV     TEMP,-(SP)              ;;SAVE TEMP FOR TYPEOUT
 (1)                                                                                     ;;TYPE VALUE
 (1)   013000   104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
 (1)   013002   004                                     .BYTE   4                       ;;TYPE 4 DIGIT(S)
 (1)   013003   001                                     .BYTE   1                       ;;TYPE LEADING ZEROS
1491   013004   020437   016364                         CMP     R4,VSET
1492   013010   003003                                  BGT     ERR
1493   013012   104401   021132                         TYPE    ,OKMSG
1494   013016   000207                                  RTS     PC
1495   013020   104401   021606                  ERR:    TYPE    ,ERMSG
1496   013024   004737   026564                         JSR     PC,WHICHV               ;INDICATE BAD UNIT
1497   013030   005237   001112                         INC     $ERTTL                  ;UPDATE ERROR TOTAL
1498   013034   000207                                  RTS     PC
1499
1500                                             ;;SUBROUTINE FOR SETTLING TESTS;;
1501   013036   012737   013120   007770         SET1A:  MOV     #1$,ERRADR              ;SET UP ERROR RECOVERY ADDRESS
1502   013044   013737   001434   001442                 MOV     CH2,DUMMY               ;LOAD DUMMY
1503   013052   004537   013224                         JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
1504   013056   000062                                  50.
1505   013060   004737   007744                         JSR     PC,TSTDAC               ;CHECK VERNIER DAC SETTING
1506   013064   063702   001464                         ADD     DAC,R2                  ;ADD RESULT TO R2
1507   013070   013737   001432   001442                MOV     CH1,DUMMY               ;CHANGE DUMMY VALUE
1508   013076   004537   013224                         JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
1509   013102   000062                                  50.
1510   013104   004737   007744                         JSR     PC,TSTDAC               ;CHECK VERNIER DAC SETTING
1511   013110   163702   001464                         SUB     DAC,R2                  ;SUBTRACT RESULT FROM R2
1512   013114   062716   000002                         ADD     #2,(SP)                 ;BUMP RETURN ADDRESS TO SKIP OVER BRANCH
1513   013120   000207                          1$:     RTS     PC                      ;RETURN
```

D 5

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 28   SEQ 0055
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED

```
1515                                          ;;SUBROUTINE TO TYPE EDGE VALUES;;
1516  013122  013703  001470          TYPEDG: MOV    EDGE,R3
1517  013126  010346                          MOV    R3,-(SP)              ;;SAVE R3 FOR TYPEOUT
(1)                                                                        ;;TYPE OCTAL VALUE OF EDGE
(1)   013130  104403                          TYPOS                        ;;GO TYPE--OCTAL ASCII
(1)   013132    004                           .BYTE  4                     ;;TYPE 4 DIGIT(S)
(1)   013133    001                           .BYTE  1                     ;;TYPE LEADING ZEROS
1518  013134  023727  013164  000001          CMP    EDGFLG,#1
1519  013142  001407                          BEQ    RET
1520  013144  062703  000007                  ADD    #7,R3
1521  013150  104401  017343          TYPE    ,MINUS                       ;TYPE ASCIZ STRING
1522  013154  010346                          MOV    R3,-(SP)              ;;SAVE R3 FOR TYPEOUT
(1)                                                                        ;;TYPE EDGE VALUE
(1)   013156  104403                          TYPOS                        ;;GO TYPE--OCTAL ASCII
(1)   013160    004                           .BYTE  4                     ;;TYPE 4 DIGIT(S)
(1)   013161    001                           .BYTE  1                     ;;TYPE LEADING ZEROS
1523  013162  000207          RET:    RTS     PC
1524  013164  000000          EDGFLG: 0
1525                                  ;SUBROUTINE TO LOAD VECTOR AREA WITH TRAP CATCHER
1526  013166  012700  000222  SETINT: MOV     #222,R0                      ;LOAD UP POINTER
1527  013172  012701  000220          MOV     #220,R1                      ;LOAD ADDRESS
1528  013176  010021          2$:     MOV     R0,(R1)+                     ;LOAD POINTER TO NEXT WORD
1529  013200  012721  004700          MOV     #4700,(R1)+                  ;LOAD 'BAD' INSTRUCTION
1530  013204  010100                  MOV     R1,R0                        ;LOAD NEW ADDRESS POINTER
1531  013206  005720                  TST     (R0)+                        ;BUMP VALUE
1532  013210  022700  001002          CMP     #1002,R0                     ;FINISHED?
1533  013214  001370                  BNE     2$                           ;BR IF NOT
1534  013216  000240                  NOP
1535  013220  000240                  NOP
1536  013222  000207                  RTS     PC                           ;EXIT
```

```
1538
1539
1540                                         ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
1541                                         ;CALL=JSR    R5,SARSUB
1542                                         ;    XXX;XXX=PERCENT
1543                                         ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
1544   013224   012537   001502    SARSUB:  MOV    (R5)+,PERCNT           ;GET PERCENT
1545   013230   006337   001502             ASL    PERCNT
1546   013234   006337   001502             ASL    PERCNT
1547   013240   006337   001502             ASL    PERCNT                 ;RESCALE PERCENT FOR 1600.
1548   013244   006337   001502             ASL    PERCNT                 ;POINTS PER BURST
1549   013250   012737   000200    001472   MOV    #200,BITPNT            ;INITIALIZE BIT POINTER AT MSB
1550   013256   005037   001464             CLR    DAC                    ;INITIALIZE DAC VALUE
1551   013262   005000             TRY:     CLR    R0
1552   013264   063737   001472    001464   ADD    BITPNT,DAC             ;TRY BIT
1553   013272   013777   001464    166056   MOV    DAC,@ADBUFF
1554   013300   012701   003100             MOV    #1600.,R1              ;SET UP FOR 1600. CONVERSIONS
1555   013304   113777   001442    166042   NXTCVT:  MOVB   DUMMY,@ADST1          ;PRESET MUX TO DUMMY CHANNEL
1556   013312   012777   001540    166040   MOV    #RETURN,@VECTOR        ;RETURN ADDRESS
1557   013320   052777   000101    166024   BIS    #101,@STREG            ;CONVERSION ON DUMMY CHANNEL
1558   013326   000001             WAIT                                  ;WAIT FOR INTERRUPT
1559   013330   017704   166022             MOV    @ADBUFF,R4            ;DUMMY READ
1560   013334   013704   001444             MOV    CHANL,R4
1561   013340   000304             SWAB   R4
1562   013342   052704   000101             BIS    #101,R4               ;INTERRUPT ENABLE START
1563   013346   010477   166000             MOV    R4,@STREG             ;JUMP TO CHANNEL + START CONVERT
1564   013352   000001             WAIT                                  ;WAIT FOR INTERRUPT
1565   013354   027737   165776    001470   CMP    @ADBUFF,EDGE
1566   013362   002001             BGE    2$
1567   013364   005200             INC    R0                    ;COUNT RESULTS .LT. EDGE
1568   013366   005301    2$:      DEC    R1
1569   013370   001345             BNE    NXTCVT
1570   013372   020037   001502             CMP    R0,PERCNT
1571   013376   003003             BGT    SHIFT
1572   013400   163737   001472    001464   SUB    BITPNT,DAC            ;TAKE THE BIT OUT
1573   013406   006237   001472    SHIFT:   ASR    BITPNT
1574   013412   001323             BNE    TRY
1575   013414   000205             RTS    R5
1576
1577                                         ;ROUTINE TO DELAY IF PROCESSER CAN NOT DO SOB INSTRUCTION
1578
1579   013416   005300             DELAY4:  DEC    R0                    ;DECREMENT R0, IS IT ZERO?
1580   013420   001376             BNE    DELAY4                ;NO
1581   013422   000002             RTI                          ;RETURN
```

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 30
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED   SEQ 0057

F 5
C
C

```
1583                                              ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
1584   013424  104401  022203                     DIFLIN: TYPE    ,MSG20               ;IDENTIFY TEST
1585   013430  004737  026572                             JSR     PC,WHICHU            ;DETERMINE UNIT #
1586   013434  013746  001514                             MOV     UNITBD,-(SP)
1587   013440  104403                                     TYPOS                        ;TELL OPER. THE #
1588   013442  001     000                               .BYTE   1,0
1589   013444  104401  001165                             TYPE    ,$CRLF
1590   013450  013702  001446                             MOV     RNA,R2               ;SET UP RANDOM NUMBER GENERATOR
1591   013454  013704  001450                             MOV     RNB,R4
1592   013460  013705  001452                             MOV     RNC,R5
1593   013464  012700  031604                             MOV     #BUFFER,R0
1594   013470  012701  010000                             MOV     #4096.,R1            ;4096 WORDS FOR HISTOGRAM
1595   013474  005020                            CLEAR1: CLR     (R0)+                 ;CLEAR BUFFER AREA
1596   013476  005301                                     DEC     R1
1597   013500  001375                                     BNE     CLEAR1
1598   013502  012700  030764                             MOV     #DIST,R0             ;DISTRIBUTION BUFFER POINTER
1599   013506  012701  000310                             MOV     #200.,R1             ;200. WORDS FOR DISTRIBUTION
1600   013512  005003                                     CLR     R3
1601   013514  005037  001504                             CLR     OUT
1602   013520  005037  001416                             CLR     WIDE
1603   013524  005037  001420                             CLR     NARROW
1604   013530  005037  001422                             CLR     FIRST
1605   013534  005037  001424                             CLR     SKIPST
1606   013540  005020                            CLEAR2: CLR     (R0)+                 ;CLEAR DISTRIBUTION BUFFER AREA
1607   013542  005301                                     DEC     R1
1608   013544  001375                                     BNE     CLEAR2
1609   013546  012700  000003                             MOV     #3,R0                ;CHANNEL 3
1610   013552  000300                                     SWAB    R0                   ;LOAD MUX BITS
1611   013554  052700  000100                             BIS     #100,R0
1612   013560  010077  165566                             MOV     R0,@STREG
1613   013564  012737  001440  001466                     MOV     #800.,DELAY          ;NOMINAL STATE WIDTH - 1 LSB
1614   013572  012737  001550  165560                     MOV     #RET1,@VECTOR
1615   013600  012701  007776                    AGAIN:  MOV     #4094.,R1
1616   013604  060402                            NEXT1:  ADD     R4,R2                 ;GENERATE A RANDOM NUMBER
1617   013606  060502                                     ADD     R5,R2
1618   013610  005502                                     ADC     R2
1619   013612  010200                                     MOV     R2,R0                ;PUT RANDOM NUMBER IN R0
1620   013614  042700  177770                             BIC     #177770,R0           ;MASK IT TO 3 BITS ONLY
1621   013620  001401                                     BEQ     CONVR1
1622   013622  077001                            DELAY1: SOB     R0,DELAY1             ;STALL TIME
1623   013624  005277  165522                    CONVR1: INC     @STREG                ;START CONVERSION
1624   013630  000001                                     WAIT
1625   013632  000240                                     NOP
1626   013634  017700  165516                             MOV     @ADBUFF,R0           ;GET CONVERTED VALUE
1627   013640  001416                                     BEQ     LODLY1               ;IGNORE IF =0
1628   013642  020027  007777                             CMP     R0,#7777             ;IGNORE IF =7777
1629   013646  001416                                     BEQ     HIDLY1
1630   013650  006300                                     ASL     R0
1631   013652  005260  031604                             INC     BUFFER(R0)           ;MAKE HISTOGRAM
1632   013656  100016                                     BPL     OKAY1
1633   013660  012760  077777  031604                     MOV     #077777,BUFFER(R0)            ;PREVENT OVERFLOW
1634   013666  000412                                     BR      OKAY1
1635   013670  005037  001426                    NOTOK1: CLR     TEMP
1636   013674  000407                                     BR      OKAY1
```

```
1637  013676  020027  007777         LODLY1:  CMP    R0,#7777               ;EQUALIZE LOOP TIME
1638  013702  001400                          BEQ    HIDLY1                 ;WITH DUMMY INSTR.
1639  013704  005201         HIDLY1:  INC    R1
1640  013706  005263  001426                  INC    TEMP(R3)
1641  013712  100766                          BMI    NOTOK1
1642  013714  005301         OKAY1:   DEC    R1
1643  013716  001514                          BEQ    AROUND
1644  013720  060204                          ADD    R2,R4                  ;GENERATE A RANDOM NUMBER
1645  013722  060504                          ADD    R5,R4
1646  013724  005504                          ADC    R4
1647  013726  010400                          MOV    R4,R0                  ;PUT RANDOM NUMBER IN R0
1648  013730  042700  177770                  BIC    #177770,R0             ;MASK IT TO 3 BITS ONLY
1649  013734  001401                          BEQ    CONVR2
1650  013736  077001         DELAY2:  SOB    R0,DELAY2              ;STALL TIME
1651  013740  005277  165406  CONVR2:  INC    @STREG                 ;START CONVERSION
1652  013744  000001                          WAIT
1653  013746  000240                          NOP
1654  013750  017700  165402                  MOV    @ADBUFF,R0             ;GET CONVERTED VALUE
1655  013754  001416                          BEQ    LODLY2                 ;IGNORE IF =0
1656  013756  020027  007777                  CMP    R0,#7777               ;IGNORE IF =7777
1657  013762  001416                          BEQ    HIDLY2
1658  013764  006300                          ASL    R0
1659  013766  005260  031604                  INC    BUFFER(R0)             ;MAKE HISTOGRAM
1660  013772  100016                          BPL    OKAY2
1661  013774  012760  077777  031604          MOV    #077777,BUFFER(R0)        ;PREVENT OVERFLOW
1662  014002  000412                          BR     OKAY2
1663  014004  005037  001426  NOTOK2:  CLR    TEMP
1664  014010  000407                          BR     OKAY2
1665  014012  020027  007777  LODLY2:  CMP    R0,#7777               ;EQUALIZE LOOP TIME
1666  014016  001400                          BEQ    HIDLY2                 ;WITH DUMMY INSTR.
1667  014020  005201         HIDLY2:  INC    R1
1668  014022  005263  001426                  INC    TEMP(R3)
1669  014026  100766                          BMI    NOTOK2
1670  014030  005301         OKAY2:   DEC    R1
1671  014032  001446                          BEQ    AROUND
1672  014034  060205                          ADD    R2,R5                  ;GENERATE A RANDOM NUMBER
1673  014036  060405                          ADD    R4,R5
1674  014040  005505                          ADC    R5
1675  014042  010500                          MOV    R5,R0                  ;PUT RANDOM NUMBER IN R0
1676  014044  042700  177770                  BIC    #177770,R0             ;MASK IT TO 3 BITS ONLY
1677  014050  001401                          BEQ    CONVR3
1678  014052  077001         DELAY3:  SOB    R0,DELAY3              ;STALL TIME
1679  014054  005277  165272  CONVR3:  INC    @STREG                 ;START CONVERSION
1680  014060  000001                          WAIT
1681  014062  000240                          NOP
1682  014064  017700  165266                  MOV    @ADBUFF,R0             ;GET CONVERTED VALUE
1683  014070  001416                          BEQ    LODLY3                 ;IGNORE IF =0
1684  014072  020027  007777                  CMP    R0,#7777               ;IGNORE IF =7777
1685  014076  001416                          BEQ    HIDLY3
1686  014100  006300                          ASL    R0
1687  014102  005260  031604                  INC    BUFFER(R0)             ;MAKE HISTOGRAM
1688  014106  100016                          BPL    OKAY3
1689  014110  012760  077777  031604          MOV    #077777,BUFFER(R0)        ;PREVENT OVERFLOW
1690  014116  000412                          BR     OKAY3
```

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 30-2
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED

H 5

SEQ 0059

```
1691  014120  005037  001426     NOTOK3: CLR     TEMP
1692  014124  000407                      BR      OKAY3
1693  014126  020027  007777     LODLY3: CMP     R0,#7777         ;EQUALIZE LOOP TIME
1694  014132  001400                      BEQ     HIDLY3           ;WITH DUMMY INSTR.
1695  014134  005201             HIDLY3: INC     R1
1696  014136  005263  001426             INC     TEMP(R3)
1697  014142  100766                      BMI     NOTOK3
1698  014144  005301             OKAY3:  DEC     R1
1699  014146  001216                      BNE     NEXT1
1700  014150  005337  001466     AROUND: DEC     DELAY
1701  014154  001211                      BNE     AGAIN
1702                             ;TAKE THE CONTENTS OF THE ACQUIRED DATA BUFFER AND
1703                             ;      TEST IF WITHIN CERTAIN LIMITS
1704                             ;      AND CREATE A STATE DISTRIBUTION BUFFER
1705                             ;      AND SORT THE VALUES INTO 'BINS'
1706  014156  012700  007776             MOV     #4094.,R0
1707  014162  012701  031606             MOV     #BUFFER+2,R1
1708  014166  012102             READ:   MOV     (R1)+,R2         ;GET STATE WIDTH
1709  014170  006202                      ASR     R2               ;1 LSB = 800.
1710  014172  006202                      ASR     R2
1711  014174  006202                      ASR     R2
1712  014176  005502                      ADC     R2               ;1 LSB = 100.
1713  014200  020227  000310             CMP     R2,#200.         ;OUT OF RANGE?
1714  014204  002403                      BLT     INRNGE
1715  014206  005237  001504             INC     OUT              ;YES - INCREMENT COUNTER
1716  014212  000423                      BR      TYPBAD
1717  014214  006302             INRNGE: ASL     R2
1718  014216  005262  030764             INC     DIST(R2)         ;MAKE STATE WIDTH DISTRIBUTION
1719  014222  006202                      ASR     R2
1720  014224  020227  000062             CMP     R2,#50.          ;IS IT 1/2 LSB?
1721  014230  002007                      BGE     NOTNAR
1722  014232  005237  001420             INC     NARROW
1723  014236  005702                      TST     R2               ;IS IT A SKIPPED STATE?
1724  014240  001002                      BNE     31$
1725  014242  005237  001424             INC     SKIPST
1726  014246  000405             31$:    BR      TYPBAD
1727  014250  020227  000226     NOTNAR: CMP     R2,#150.         ;IS IT 1.5 LSB?
1728  014254  003425                      BLE     LAST
1729  014256  005237  001416             INC     WIDE
1730  014262  005737  001422     TYPBAD: TST     FIRST
1731  014266  001004                      BNE     60$
1732  014270  005237  001422             INC     FIRST
1733  014274  104401  020771             TYPE    ,STATE
1734  014300  010103             60$:    MOV     R1,R3
1735  014302  162703  031606             SUB     #BUFFER+2,R3
1736  014306  006203                      ASR     R3
1737  014310  010346                      MOV     R3,-(SP)         ;;SAVE R3 FOR TYPEOUT
  (1)                                                              ;;TYPE STATE
  (1)  014312  104403                     TYPOS                    ;;GO TYPE--OCTAL ASCII
  (1)  014314  004                        .BYTE   4                ;;TYPE 4 DIGIT(S)
  (1)  014315  001                        .BYTE   1                ;;TYPE LEADING ZEROS
1738  014316  104401  020765             TYPE    ,DASH
1739  014322  104416                      TYPDC
1740  014324  104401  020756             TYPE    ,LSBMSG
```

I 5

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 30-3
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED

SEQ 0060

```
1741  014330  005300                  LAST:   DEC     R0
1742  014332  001315                          BNE     READ
1743                                  ;REPORT TO THE OPERATOR THE DIFFERENT STATE VALUES
1744                                  ;       IN THE FORM OF A GENERAL STATUS AND INDICATE OK/ERROR
1745  014334  112737  000177  024545          MOVB    #177,DECPNT
1746  014342  013702  001424                  MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
1747  014346  104416                           TYPDC                          ;TYPE IT
1748  014350  104401  021623                  TYPE    ,SKPMSG        ;TYPE MESSAGE
1749  014354  005737  001424                  TST     SKIPST
1750  014360  001407                          BEQ     1$
1751  014362  104401  021606                  TYPE    ,ERMSG         ;TYPE 'ERROR'
1752  014366  004737  026564                  JSR     PC,WHICHV      ;INDICATE BAD UNIT
1753  014372  005237  001112                  INC     $ERTTL         ;UPDATE ERROR COUNT
1754  014376  000402                          BR      NAR
1755  014400  104401  021132          1$:     TYPE    ,OKMSG         ;TYPE #OK#
1756  014404  013702  001420          NAR:    MOV     NARROW,R2              ;GET NO. OF NARROW STATES
1757  014410  104416                           TYPDC                          ;TYPE IT
1758  014412  104401  021645                  TYPE    ,NARMSG        ;TYPE MESSAGE
1759  014416  013702  001416                  MOV     WIDE,R2
1760  014422  063702  001504                  ADD     OUT,R2
1761  014426  104416                           TYPDC                          ;TYPE NO. OF WIDE STATES
1762  014430  104401  021704                  TYPE    ,WIDMSG        ;TYPE MESSAGE
1763  014434  013702  001504                  MOV     OUT,R2
1764  014440  104416                           TYPDC                          ;TYPE NO. OF STATES OUTSIDE 2 LSB
1765  014442  104401  0?1743                  TYPE    ,OUTMSG        ;TYPE MESSAGE
1766  014446  005737  001504                  TST     OUT
1767  014452  001407                          BEQ     11$
1768  014454  104401  021606                  TYPE    ,ERMSG         ;TYPE 'ERROR'
1769  014460  004737  026564                  JSR     PC,WHICHV      ;DETERMINE BAD UNIT
1770  014464  005237  001112                  INC     $ERTTL         ;UPDATE ERROR COUNT
1771  0?4470  000402                          BR      HALF
1772  014472  104401  021132          11$:    TYPE    ,OKMSG         ;TYPE 'OK'
1773  014476  013702  001420          HALF:   MOV     NARROW,R2
1774  014502  063702  001416                  ADD     WIDE,R2
1775  014506  063702  001504                  ADD     OUT,R2
1776  014512  010200                          MOV     R2,R0
1777  014514  104416                           TYPDC                          ;TYPE NO. OF STATES OUTSIDE LIMITS
1778  014516  112737  000056  024545          MOVB    #56,DECPNT
1779  014524  104401  021776                  TYPE    ,HAFMSG
1780  014530  020027  000051                  CMP     R0,#41.        ;COMPARE IT TO NOMINAL
1781  014534  003407                          BLE     21$
1782  014536  104401  021606                  TYPE    ,ERMSG         ;TYPE 'ERROR'
1783  014542  004737  026564                  JSR     PC,WHICHV      ;INDICATE BAD UNIT
1784  014546  005237  001112                  INC     $ERTTL         ;UPDATE ERROR COUNT
1785  014552  000402                          BR      SWDIST
1786  014554  104401  021132          21$:    TYPE    ,OKMSG         ;TYPE 'OK'
1787                                  ;DETERMINE IF VT55 TYPE TERMINAL IS CONNECTED
1788                                  ;       IF NOT BYPASS THIS SECTION
1789                                  ;       IF VT55/VT105 GRAHIC TERMINAL REPORT THE DISTRIBUTION CURVE
1790  014560  005737  001460          SWDIST: TST     FLAG           ;BIT MAP TERMINAL AVAILABLE?
1791  014564  001426                          BEQ     RELACC         ;BR IF NOT
1792  014566  004737  015254                  JSR     PC,DELCLR      ;WAIT AWHILE, THEN CLEAR BIT MAP TERMINAL
1793  014572  104401  022247                  TYPE    ,MSG16
1794  014576  104401  023027                  TYPE    ,BUFF1         ;TYPE BUFF1-PRINT GRID
```

```
1795  014602  012700  030764              MOV     #DIST,R0      ;POINTER TO STATE WIDTH DISTRIBUTION
1796  014606  012701  000310              MOV     #200.,R1      ;GO 200. TIMES UP TO 2 LSB
1797  014612  012002           NXTY1:     MOV     (R0)+,R2
1798  014614  004737  015750              JSR     PC,LOADY
1799  014620  005002                      CLR     R2
1800  014622  004737  015750              JSR     PC,LOADY
1801  014626  005301                      DEC     R1
1802  014630  001370                      BNE     NXTY1
1803  014632  104401  022765              TYPE    ,C2           ;TYPE ASCIZ STRING
1804  014636  004737  015254              JSR     PC,DELCLR
```

```
1806                                     ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR
1807
1808   014642  005001          RELACC: CLR     R1                      ;RUNNING ERROR = 0
1809   014644  005003                  CLR     R3                      ;MAXIMUM ERROR = 0
1810   014646  104401  022574          TYPE    ,MSG21
1811   014652  012700  031606          MOV     #BUFFER+2,R0
1812   014656  011002          NXTSTA: MOV     (R0),R2                 ;STATE WIDTH = R2
1813   014660  162702  001440          SUB     #800.,R2                ;STATE WIDTH ERROR IN R2
1814   014664  060201                  ADD     R2,R1                   ;UPDATE RUNNING ERROR
1815   014666  010120                  MOV     R1,(R0)+                ;SAVE IN BUFFER
1816   014670  010104                  MOV     R1,R4                   ;SAVE IN R4 ALSO
1817   014672  100001                  BPL     PLUS                    ;IS IT POSITIVE?
1818   014674  005404                  NEG     R4                      ;NO - MAKE IT POSITIVE
1819   014676  020403          PLUS:   CMP     R4,R3                   ;CHECK AGAINST PREVIOUS MAX. ERROR
1820   014700  003405                  BLE     NOTNEW                  ;NOT A NEW MAXIMUM
1821   014702  010403                  MOV     R4,R3                   ;UPDATE MAXIMUM IN R3
1822   014704  010005                  MOV     R0,R5
1823   014706  162705  031606          SUB     #BUFFER+2,R5
1824   014712  006205                  ASR     R5                      ;R5=EDGE VALUE AT MAX. RELACC
1825   014714  020027  051602  NOTNEW: CMP     R0,#BUFFER+8190.   ;DONE?
1826   014720  001356                  BNE     NXTSTA                  ;NO - REPEAT
1827   014722  006203                  ASR     R3                      ;RESCALE FROM 1 LSB = 800. SCALING
1828   014724  006203                  ASR     R3                      ;TO 1 LSB = 100. SCALING
1829   014726  006203                  ASR     R3
1830   014730  005503                  ADC     R3
1831   014732  010302                  MOV     R3,R2
1832   014734  104416                  TYPDC
1833   014736  104401  022621          TYPE    ,LINEA
1834   014742  010546                  MOV     R5,-(SP)                ;;SAVE R5 FOR TYPEOUT
   (1)                                                                 ;;TYPE VALUE
   (1)   014744  104403                  TYPOS                         ;;GO TYPE--OCTAL ASCII
   (1)   014746  004                     .BYTE   4                     ;;TYPE 4 DIGIT(S)
   (1)   014747  001                     .BYTE   1                     ;;TYPE LEADING ZEROS
1835   014750  104401  021130          TYPE    ,SLASH                          ;PRINT '/'
1836   014754  005205                  INC     R5
1837   014756  010546                  MOV     R5,-(SP)                ;;SAVE R5 FOR TYPEOUT
   (1)                                                                 ;;TYPE VALUE
   (1)   014760  104403                  TYPOS                         ;;GO TYPE--OCTAL ASCII
   (1)   014762  004                     .BYTE   4                     ;;TYPE 4 DIGIT(S)
   (1)   014763  001                     .BYTE   1                     ;;TYPE LEADING ZEROS
1838   014764  020337  016366          CMP     R3,VLIN
1839   014770  003407                  BLE     41$
1840   014772  104401  021606          TYPE    ,ERMSG
1841   014776  004737  026564          JSR     PC,WHICHV               ;INDICATE BAD UNIT
1842   015002  005237  001112          INC     $ERTTL                  ;UPDATE ERROR COUNT
1843   015006  000402                  BR      42$
1844   015010  104401  021132  41$:    TYPE    ,OKMSG
1845   015014  005737  001460  42$:    TST     FLAG                    ;BIT MAP TERMINAL ?
1846   015020  001503                  BEQ     LO2                     ;BR IF NOT
1847   015022  012700  031604          MOV     #BUFFER,R0
1848   015026  012701  010000          MOV     #4096.,R1
```

```
1850   015032   011002                   GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
1851   015034   006202                           ASR     R2              ;RESCALE IT TO 1 LSB = 100.
1852   015036   006202                           ASR     R2
1853   015040   006202                           ASR     R2
1854   015042   005502                           ADC     R2
1855   015044   062702   000166                  ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
1856   015050   010220                           MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
1857   015052   005301                           DEC     R1
1858   015054   001366                           BNE     GETDAT
1859   015056   012700   031604                  MOV     #BUFFER,R0
1860   015062   012704   031604                  MOV     #BUFFER,R4
1861   015066   012705   031606                  MOV     #BUFFER+2,R5
1862   015072   012701   001000                  MOV     #512.,R1
1863   015076   012702   000007         NXT8:    MOV     #7.,R2
1864   015102   012003                           MOV     (R0)+,R3
1865   015104   010337   001474                  MOV     R3,MIN          ;MINIMUM
1866   015110   010337   001500                  MOV     R3,MAX          ;MAXIMUM
1867   015114   012003                   NXTCMP: MOV     (R0)+,R3
1868   015116   020337   001474                  CMP     R3,MIN
1869   015122   002002                           BGE     MAXTST
1870   015124   010337   001474                  MOV     R3,MIN          ;NEW MINIMUM
1871   015130   020337   001500         MAXTST:  CMP     R3,MAX
1872   015134   003402                           BLE     TST8
1873   015136   010337   001500                  MOV     R3,MAX          ;NEW MAXIMUM
1874   015142   005302                   TST8:   DEC     R2
1875   015144   001363                           BNE     NXTCMP
1876   015146   013724   001474                  MOV     MIN,(R4)+
1877   015152   013725   001500                  MOV     MAX,(R5)+
1878   015156   022425                           CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
1879   015160   005301                           DEC     R1
1880   015162   001345                           BNE     NXT8
1881   015164   104401   022143                  TYPE    ,MSG18
1882   015170   104401   023055                  TYPE    ,BUFF2          ;TYPE BUFF2
1883   015174   012700   031604                  MOV     #BUFFER,R0
1884   015200   004737   015232                  JSR     PC,LOAD
1885   015204   104401   022773                  TYPE    ,C3             ;TYPE ASCIZ STRING
1886   015210   012700   031606                  MOV     #BUFFER+2,R0
1887   015214   004737   015232                  JSR     PC,LOAD
1888   015220   104401   022765                  TYPE    ,C2             ;TYPE ASCIZ STRING
1889   015224   004737   015254                  JSR     PC,DELCLR
1890   015230   000207                   LO2:    RTS     PC
1891   015232   012701   001000         LOAD:    MOV     #512.,R1
1892   015236   012002                   LOAD0:  MOV     (R0)+,R2
1893   015240   005720                           TST     (R0)+
1894   015242   004737   015750                  JSR     PC,LOADY
1895   015246   005301                           DEC     R1
1896   015250   001372                           BNE     LOAD0
1897   015252   000207                           RTS     PC
```

```
1899  015254  032777  002000  163656  DELCLR: BIT     #BIT10,@SWR        ;TEST FOR HALT FOR DISPLAY
1900  015262  001402                          BEQ     1$                 ;;DON'T HALT FOR DISPLAY
1901  015264  000000                          HALT
1902  015266  000407                          BR      3$                 ;;
1903  015270  005000           1$:            CLR     R0
1904  015272  012701  000020                  MOV     #20,R1             ;DELAY BEFORE CLEANING SCREEN
1905  015276  005300           2$:            DEC     R0
1906  015300  001376                          BNE     2$
1907  015302  005301                          DEC     R1
1908  015304  001374                          BNE     2$
1909  015306  104401  023114   3$:            TYPE    ,VTINIT
1910  015312  000207                          RTS     PC
1911                          ;;TYPE RMS AND PEAK VALUES;;
1912  015314  005702          TYPRP:  TST     R2                 ;IS NOISE POSITIVE?
1913  015316  100001                          BPL     POSNOI             ;YES
1914  015320  005002                          CLR     R2                 ;R2<0,SET R2=0
1915  015322  104416          POSNOI: TYPDC
1916  015324  104401  023016                  TYPE    ,MLSBAT            ;TYPE " LSB AT "
1917  015330  004737  013122                  JSR     PC,TYPEDG
1918  015334  104401  021113                  TYPE    ,CHAN              ;TYPE " ON CHANNEL "
1919  015340  013746  001444                  MOV     CHANL,-(SP)        ;;SAVE CHANL FOR TYPEOUT
 (1)                                                                     ;;TYPE CHANL
 (1)  015344  104403                          TYPOS                      ;;GO TYPE--OCTAL ASCII
 (1)  015346     002                          .BYTE   2                  ;;TYPE 2 DIGIT(S)
 (1)  015347     000                          .BYTE   0                  ;;SUPPRESS LEADING ZEROS
1920  015350  000207                          RTS     PC
```

```
1922                                   ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
1923   015352  012500                  CONVRT: MOV    (R5)+,R0                    ;GET CHANNEL VALUE
1924   015354  010037  001444                  MOV    R0,CHANL
1925   015360  012777  000200  163770  CONVTC: MOV    #200,@ADBUFF               ;LOAD VERNIER DAC
1926   015366  113700  001444          CONVCD: MOVB   CHANL,R0                    ;GET CHANNEL
1927   015372  000300                          SWAB   R0                         ;SET UP A/D STATUS REGISTER
1928   015374  052700  000100                  BIS    #100,R0                     ;ENABLE INTERRUPTS
1929   015400  010077  163746                  MOV    R0,@STREG
1930   015404  012700  010000                  MOV    #10000,R0                   ;DAC SETTLING DELAY
1931   015410  005300                  1$:     DEC    R0
1932   015412  001376                          BNE    1$
1933   015414  005037  001426                  CLR    TEMP
1934   015420  012777  001540  163732          MOV    #RETURN,@VECTOR            ;LOAD VECTOR
1935   015426  012777  000200  163726          MOV    #200,@VECTR1               ;SET UP NEW PSW
1936   015434  012700  000010                  MOV    #10,R0                      ;SET UP COUNTER
1937   015440  005277  163706          2$:     INC    @STREG                      ;START CONVERSION
1938   015444  000001                          WAIT                               ;WAIT FOR CONVERSION
1939   015446  067737  163704  001426          ADD    @ADBUFF,TEMP               ;READ BUFFER
1940   015454  005300                          DEC    R0
1941   015456  001370                          BNE    2$                          ;DO 8 TIMES
1942   015460  006237  001426                  ASR    TEMP                        ;AVERAGE VALUE
1943   015464  006237  001426                  ASR    TEMP
1944   015470  006237  001426                  ASR    TEMP
1945   015474  005537  001426                  ADC    TEMP
1946   015500  000205                          RTS    R5                          ;RETURN
1947
1948                                   ;COMPARE $GDDAT AND $BDDAT;;
1949   015502  012537  001124          COMPAR: MOV    (R5)+,$GDDAT               ;GET GOOD DATA
1950   015506  013537  001462                  MOV    @(R5)+,SPREAD              ;GET SPREAD
1951   015512  013737  001426  001126          MOV    TEMP,$BDDAT                ;GET BAD(ACTUAL) DATA
1952   015520  013701  001126                  MOV    $BDDAT,R1
1953   015524  013700  001124                  MOV    $GDDAT,R0
1954   015530  160100                          SUB    R1,R0                       ;GET DIFFERENCE
1955   015532  100001                          BPL    7$
1956   015534  005400                          NEG    R0
1957   015536  020037  001462          7$:     CMP    R0,SPREAD                   ;COMPARE IT TO SPREAD
1958   015542  003001                          BGT    10$                         ;GO TO ERROR PRINTOUT
1959   015544  005725                          TST    (R5)+                       ;BUMP RETURN POINTER AROUND ERROR CALL
1960   015546  000205                  10$:    RTS    R5
```

```
1962                                    ;;ROUTINE TO AVERAGE 8 CONVERSIONS ON GOOD AD;;
1963   015550  012500          GCONVT:  MOV     (R5)+,R0                    ;GET CHANNEL VALUE
1964   015552  010037  001444           MOV     R0,CHANL
1965   015556  000300                    SWAB    R0
1966   015560  005037  001426           CLR     TEMP
1967   015564  010077  163606           MOV     R0,@GSTREG                  ;LOAD CHANNEL INTO MIX BITS
1968   015570  012700  010000           MOV     #10000,R0
1969   015574  005300          2$:      DEC     R0
1970   015576  001376                    BNE     2$
1971   015600  012777  001540  163574   MOV     #RETURN,@GVECT             ;LOAD VECTOR
1972   015606  012777  000200  163570   MOV     #200,@GVECT+2              ;SET UP NEW PRIORITY
1973   015614  012700  000010           MOV     #10,R0                     ;SET UP COUNTER
1974   015620  152777  000101  163550  1$:  BISB  #101,@GSTREG             ;SET INTRPT. EN., START CONV.
1975   015626  000001                    WAIT                             ;WAIT FOR CONVERSION
1976   015630  067737  163544  001426   ADD     @GADBUF,TEMP               ;READ BUFFER
1977   015636  005300                    DEC     R0
1978   015640  001367                    BNE     1$                         ;DO 8 TIMES
1979   015642  006237  001426           ASR     TEMP             ;AVERAGE VALUE
1980   015646  006237  001426           ASR     TEMP
1981   015652  006237  001426           ASR     TEMP
1982   015656  005537  001426           ADC     TEMP
1983   015662  000205                    RTS     R5                 ;RETURN
1984
1985                                    ;;SUBROUTINE TO CONVERT 2.60 VOLTS TO 15.00 VOLTS;;
1986                                    ;;FUNNY NUMBER CALCULATED BY:
1987                                    ;;      (15*2.56/(VOLTAGE))/0.0025
1988
1989   015664  032703  004000  CONV15:  BIT     #BIT11,R3          ;IS RESULT MINUS?
1990   015670  001003                    BNE     1$                 ;;NO
1991   015672  005403                    NEG     R3                 ;YES, MAKE IT PLUS
1992   015674  104401  017343           TYPE    ,MINUS             ;TYPE '-'
1993   015700  042703  174000  1$:      BIC     #174000,R3         ;CLEAR UPPER 5 BITS
1994   015704  005002                    CLR     R2                 ;CLEAR RESULT REGISTER
1995   015706  012701  013424           MOV     #5908.,R1          ;PUT FUNNY NUMBER INTO R1
1996   015712  012700  002000           MOV     #BIT10,R0          ;SETUP TEST BIT
1997   015716  030003          2$:      BIT     R0,R3              ;MULTIPLY TEMP BY FUNNY NUMBER
1998   015720  001401                    BEQ     3$                 ;;
1999   015722  060102                    ADD     R1,R2
2000   015724  006201          3$:      ASR     R1
2001   015726  006200                    ASR     R0
2002   015730  001372                    BNE     2$                 ;;NOT FINISHED YET
2003   015732  006202                    ASR     R2                 ;SCALE TO .01 VOLTS / BIT
2004   015734  006202                    ASR     R2
2005   015736  005502                    ADC     R2
2006   015740  104416                    TYPDC                      ;TYPE RESULTS
2007   015742  104401  020442           TYPE    ,VOLTS             ;TYPE 'VOLTS'
2008   015746  000207                    RTS     PC
2009
```

```
2011
2012                                  ;SUBROUTINE LOADY;
2013   015750  005702        LOADY:   TST    R2                    ;ROUTINE TO LOAD VLAUE INTO R2
2014   015752  100001                 BPL    PLUSR2                ;AS A VT55 Y-VALUE
2015   015754  005002                 CLR    R2
2016   015756  020227  000353 PLUSR2: CMP    R2,#235.
2017   015762  002402                 BLT    LESS
2018   015764  012702  000353         MOV    #235.,R2
2019   015770  010203        LESS:    MOV    R2,R3
2020   015772  042702  i77740         BIC    #177740,R2
2021   015776  052702  000040         BIS    #40,R2
2022   016002  105777  163142 B10:    TSTB   a$TPS                 ;PRINT CHARACTER
2023   016006  100375                 BPL    B10
2024   016010  110277  163136         MOVB   R2,a$TPB
2025   016014  006203                 ASR    R3
2026   016016  006203                 ASR    R3
2027   016020  006203                 ASR    R3
2028   016022  006203                 ASR    R3
2029   016024  006203                 ASR    R3
2030   016026  042703  177770         BIC    #177770,R3
2031   016032  052703  000040         BIS    #40,R3
2032   016036  105777  163106 B11:    TSTB   a$TPS                 ;PRINT CHARACTER
2033   016042  100375                 BPL    B11
2034   016044  110377  163102         MOVB   R3,a$TPB
2035   016050  000207                 RTS    PC
```

D 6

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 37
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED   SEQ 0068

```
2037                                        ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
2038                                        ;;IN R2 AS X.XX;;
2039  016052  005702              DECTYP:  TST    R2                      ;TEST VALUE TO BE TYPED
2040  016054  100003                       BPL    POS
2041  016056  104401   017343              TYPE   ,MINUS                      ;TYPE MINUS SIGN
2042  016062  005402                       NEG    R2
2043  016064  020227   023417     POS:     CMP    R2,#9999.                 ;>9999. REPLACE IT WITH 9999.
2044  016070  003402                       BLE    OKAYD
2045  016072  012702   023417              MOV    #9999.,R2
2046  016076  105037   024547     OKAYD:   CLRB   ONES            ;CLEAR ONES
2047  016102  105037   024546              CLRB   TENS            ;CLEAR TENS
2048  016106  105037   024544              CLRB   HUNS            ;CLEAR HUNS
2049  016112  105037   024543              CLRB   THOUS           ;CLEAR THOUS
2050  016116  005702              TESTR2:  TST    R2                   ;CONVERT VALUE TO A DECIMAL VALUE
2051  016120  001434                       BEQ    TYPOUT
2052  016122  005302                       DEC    R2
2053  016124  105237   024547              INCB   ONES
2054  016130  123727   024547  000012      CMPB   ONES,#10.
2055  016136  001367                       BNE    TESTR2
2056  016140  105037   024547              CLRB   ONES
2057  016144  105237   024546              INCB   TENS
2058  016150  123727   024546  000012      CMPB   TENS,#10.
2059  016156  001357                       BNE    TESTR2
2060  016160  105037   024546              CLRB   TENS
2061  016164  105237   024544              INCB   HUNS
2062  016170  123727   024544  000012      CMPB   HUNS,#10.
2063  016176  001347                       BNE    TESTR2          ;;
2064  016200  105037   024544              CLRB   HUNS
2065  016204  105237   024543              INCB   THOUS
2066  016210  000742                       BR     TESTR2
2067  016212  152737   000060  024543  TYPOUT:  BISB   #60,THOUS             ;PREPARE FOR TYPOUT
2068  016220  152737   000060  024544      BISB   #60,HUNS
2069  016226  152737   000060  024546      BISB   #60,TENS
2070  016234  152737   000060  024547      BISB   #60,ONES
2071  016242  123727   024543  000060      CMPB   THOUS,#60
2072  016250  001403                       BEQ    1$              ;;
2073  016252  104401   024543              TYPE   ,THOUS
2074  016256  000002                       RTI
2075  016260  104401   024544     1$:      TYPE   ,HUNS           ;TYPE VALUE
2076  016264  000002                       RTI
```

E 6

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 38
CVMNAA.P11   DETERMINE IF MORE MNCAD'S TO BE TESTED   SEQ 0069

```
2078                                          ;SUBROUTINE TO SENSE THE 'WFTEST' FLAG AND USE WIDE/NARROW ERROR TOLERANCES
2079
2080   016266  012701  016360     WFADJ:  MOV    #VNR,R1           ;SUBROUTINE TO SET LIMITS
2081   016272  005737  001476             TST    WFTEST            ;RUNNING ON TESTER ?
2082   016276  100403                      BMI    1$               ;;YES
2083   016300  012702  016372             MOV    #VARLT1,R2        ;WFTEST NOT MINUS, USE NORMAL LIMITS
2084   016304  000402                      BR     2$               ;;
2085   016306  012702  016402     1$:     MOV    #VARLT2,R2        ;WFTEST MINUS, USE OPTION AREA LIMITS
2086   016312  012221             2$:     MOV    (R2)+,(R1)+       ;SET UP LIMITS
2087   016314  005711                      TST    (R1)             ;DONE?
2088   016316  100375                      BPL    2$               ;;NO
2089   016320  000207                      RTS    PC
2090
2091   016322  000000     V0:     0                                ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
2092   016324  000002     V2:     2
2093   016326  000012     V10:    10.
2094   016330  000012     V12:    12
2095   016332  000062     V50D:   50.
2096   016334  000144     V100D:  100.
2097   016336  000326     V326:   326
2098
2099                                          ;*VOLTAGE TABLE OF EXPECTED VALUES (SINGLE ENDED) <TEST MODULE>
2100   016340  005560     VTABLE: 5560              ;+2.2 VOLTS      <CH10, 20, 30 ETC>
2101   016342  002220             2220              ;-2.2 VOLTS
2102   016344  004670             4670              ;+1.1 VOLTS
2103   016346  003110             3110              ;-1.1 VOLTS
2104   016350  007340             7340              ;+4.4 VOLTS      <CH14, 24, 34 ETC>
2105   016352  000440             0440              ;-4.4 VOLTS
2106   016354  006450             6450              ;+3.3 VOLTS
2107   016356  001330             1330              ;-3.3 VOLTS      <CH17, 27, 37 ETC>
2108
2109   016360  000041     VNR:    33.               ;.33 LSB,NORMAL LIMITS FOR SYSTEM
2110   016362  000310     VNP:    200.              ;2 LSB,   INTEGRATION AND FIELD USE ON SPEC TESTS
2111   016364  000144     VSET:   100.              ;1 LSB
2112   016366  000175     VLIN:   125.              ;1.25 LSB
2113   016370  100000             BIT15
2114
2115                                 ;LIMITS FOR NON-TESTER
2116
2117   016372  000050     VARLT1: 40.               ;.4 LSB, NORMAL LIMITS FOR SYSTEM
2118   016374  000310             200.              ;2 LSB,   INTEGRATION AND FIELD USE ON SPEC TESTS
2119   016376  000144             100.              ;1 LSB
2120   016400  000175             125.              ;1.25 LSB
2121
2122                                 ;LIMITS FOR TESTER
2123
2124   016402  000041     VARLT2: 33.                         ;.33 LSB RMS NOISE LIMIT
2125   016404  000226             150.                        ;1.5 LSB PEAK NOISE LIMIT
2126   016406  000132             90.                         ;.9 LSB INTER-CHANNEL SETTLING LIMIT
2127   016410  000144             100.                        ;1 LSB RELATIVE ACCURACY ERROR LIMIT
2128
```

```
2130                                     .SBTTL   END OF PASS ROUTINE
 (1)
 (2)                                     ;;*****************************************************************
 (1)                                     ;*INCREMENT THE PASS NUMBER ($PASS)
 (1)                                     ;*TYPE 'END PASS #XXXXX'' (WHERE XXXXX IS A DECIMAL NUMBER)
 (1)                                     ;*IF THERES A MONITOR GO TO IT
 (1)                                     ;*IF THERE ISN'T JUMP TO EXTMSG
 (1)
 (1)  016412                     $EOP:
 (2)  016412  000240                     NOP
 (1)  016414  005037  001102             CLR      $TSTNM          ;;ZERO THE TEST NUMBER
 (1)  016420  005037  001160             CLR      $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 (1)  016424  005237  001176             INC      $PASS           ;;INCREMENT THE PASS NUMBER
 (1)  016430  042737  100000  001176     BIC      #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 (1)  016436  005327                     DEC      (PC)+           ;;LOOP?
 (1)  016440  000001             $EOPCT: .WORD    1
 (1)  016442  003022                     BGT      $DOAGN          ;;YES
 (1)  016444  012737                     MOV      (PC)+,a(PC)+    ;;RESTORE COUNTER
 (1)  016446  000001             $ENDCT: .WORD    1
 (1)  016450  016440                     $EOPCT
 (1)  016452  104401  016517             TYPE     ,$ENDMG         ;;TYPE 'END PASS #'
 (2)  016456  013746  001176             MOV      $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 (2)  016462  104405                     TYPDS                    ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)  016464  104401  016514             TYPE     ,$ENULL         ;;TYPE A NULL CHARACTER
 (1)  016470  013700  000042     $GET42: MOV      a#42,R0         ;;GET MONITOR ADDRESS
 (1)  016474  001405                     BEQ      $DOAGN          ;;BRANCH IF NO MONITOR
 (1)  016476  000005                     RESET                    ;;CLEAR THE WORLD
 (1)  016500  004710             $ENDAD: JSR      PC,(R0)         ;;GO TO MONITOR
 (1)  016502  000240                     NOP                      ;;SAVE ROOM
 (1)  016504  000240                     NOP                      ;;FOR
 (1)  016506  000240                     NOP                      ;;ACT11
 (1)  016510                     $DOAGN:
 (1)  016510  000137                     JMP      a(PC)+          ;;RETURN
 (1)  016512  016534             $RTNAD: .WORD    EXTMSG
 (1)  016514     377     377     000 $ENULL: .BYTE    -1,-1,0         ;;NULL CHARACTER STRING
 (1)  016517     015  042412  042116 $ENDMG: .ASCIZ   <15><12>/END PASS #/
 (1)  016524  050040  051501  020123
 (1)  016532  000043
2131  016534  052777  000100  162402 EXTMSG: BIS      #BIT6,a$TKS     ;ENABLE KRB INTR.
2132  016542  005737  001112             TST      $ERTTL          ;ANY ERRORS
2133  016546  001415                     BEQ      1$              ;BR IF NOT
2134  016550  104401  022721             TYPE     ,ERRTOT         ;TYPE TOTAL ERROR COUNT PRIMER
2135  016554  013746  001112             MOV      $ERTTL,-(SP)    ;GET VALUE
2136  016560  104405                     TYPDS                    ;REPORT IT
2137  016562  005737  001440             TST      NMBEXT          ;TEST IF MULTIPLE
2138  016566  001405                     BEQ      1$              ;BR IF NOT
2139  016570  104401  022750             TYPE     ,MESGD          ;TYPE BAD UNIT PRIMER
2140  016574  013746  001510             MOV      BADUNT,-(SP)
2141  016600  104406                     TYPBN                    ;REPORT 1 + 0'S
2142  016602  104401  016514     1$:     TYPE,    $ENULL          ;ENSURE ALL TEXT GET TYPED
2143  016606  000137                     JMP      a(PC)+          ;RETURN
2144  016610  001556             AGTST:  BEGIN
```

```
2146                                            ;*
2147                                            ;*THIS ROUTINE WILL PROTECT THE PROGRAM
2148                                            ;*FROM INTERUPTS (BAD ONES).
2149                                            ;*
2150                                            ;*THE TRAP CATCHER IS SET UP FOR
2151                                            ;*      .WORD   .+2
2152                                            ;*      JSR PC,RO
2153                                            ;*
2154                                            ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
2155                                            ;*GOTO THE VECTOR AND PCITK UP THE ''.+2'' AS AN ADDRESS
2156
2157                                            ;*AND ''4700'' AS NEW STATUS.
2158                                            ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE ''JSR PC,RO'' (AN ILLEGAL INSTR.).
2159                                            ;*AND TRAP TO LOCATION '4''. IN LOCATION 4 WE HAVE A
2160                                            ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4.
2161                                            ;*WE WILL REPORT IT IN THE SAME MANNER THAT WER WOULD
2162                                            ;*REPORT ANY OTHER ERROR.
2163                                            ;*IF A BUSS ERROR TRAP DID OCCUT AND CAUSE A TRAP TO 4.
2164                                            ;*WE WILL HALT.
2165
2166  016612  011637  017072        IOTRD:  MOV     (SP),TRTO       ;GET WHERE WE CAME TO.
2167  016616  162737  000004  017072         SUB     #4,TRTO         ;FORM READ ADDR.
2168  016624  023727  017072  001000         CMP     TRTO,#1000      ;DID TRAP FROM LESS THAN ADDR. 1000?
2169  016632  003402                          BLE     2$              ;NO-CONTINUE.
2170  016634  000000        1$:     HALT                    ;A BUSS ERROR TIME OUT TRAP BROUGHT US HERE.
2171                                                                ;ADDRESS CONTAINED IN TRTO.
2172  016636  000776                          BR      1$              ;DON'T ALLOW CONTINUE.
2173  016640  016637  000004  017074  2$:  MOV     4(SP),TRFRO     ;GET TRAPPED FROM ADDR.
2174  016646  122737  000021  001102         CMPB    #21,$TSTNM      ;LESS THAN INTERRUPT TESTS?
2175  016654  003402                          BLE     3$              ;NO MUST BE WRONG VECTOR.
2176                             ;/////////////////////////////////////////////////////////////////////
2177  016656  104003                          ERROR   3               ;ERROR! ILLEGAL INTERRUPT OR
2178                                                                ;INTERRUPT TO WRONG VECTOR.
2179                                                                ;IF TEST NO. IS LESS THAN 10,ITS
2180                                                                ;LIKELY(BUT NO EXCLUSIVELY)TO BE A
2181                                                                ;DEVICE OTHER THAN THE DEVICE UNDER TEST.
2182                                                                ;IF THE INTERRUPT OCCURED
2183                                                                ;DURING AN INTERRUPT TEST, I'D
2184                                                                ;SUSPECT A PROBLEM WITH THE DEVICE UNDER TEST.
2185                                                                ;IF THE ADDRESS THE INTERRUPT
2186                                                                ;VECTORED TO IS WITHIN THE RANGE OF
2187                                                                ;VECTORS ASSIGNED TO THE DEVICE,
2188                                                                ;THEN I'D SUSPECT THE DEVICE
2189                                                                ;INTERRUPTD ILLEGALLY.
2190                                                                ;IF THE ADDRESS THE INTERRUPT
2191                                                                ;VECTORED TO IS OUTSIDE OF THE
2192                                                                ;RANGE ASSIGNED TO THE DEVICE
2193                                                                ;I'D SUSPECT THAT THE
2194                                                                ;DEVICE PUT THE WRONG INTERRUPT
2195                                                                ;VECTOR ON THE BUS DURING THE INTERRUPT
2196                                                                ;PROCESS.
2197                                                                ;       NOTE:
2198                                                                ;FOR THIS ERROR - DON'T USE
2199                                                                ;''LOOP ON ERROR'' OPTION.
```

```
2200                                                        ;ALSO EXPECT THAT THE INTERRUPT TEST TO
2201                                                        ;WILL REPOT THAT THE DEVICE DIDN'T
2202                                                        ;INTERRUPT.
2203                                                        ;FOLLOW THE RECOMMENDED PROCEEDURE
2204                                                        ; IN THE DOCUMENT (ON THIS DIAGNOSTIC)
2205                                                        ;FOR LOOPING ON TEST.
2206
2207                                    ;////////////////////////////////////////////;///////////////////////////////////////
2208   016660  000002                          RTI
2209   016662  022626             3$:    CMP    (SP)+,(SP)+   ;POP OFF JSR TRAP
2210   016664  022626                    CMP    (SP)+,(SP)+   ;POP OFF WRONG INTR.
2211   016666  005737  001176             TST    $PASS        ;IS THIS THE  IRST PASS?
2212   016672  001025                     BNE    4$           ;;NO, DON'T REPORT
2213   016674  104401  017524             TYPE   .VTMSG       ;TYPE 'EXPECTED INTR. AT ''
2214   016700  004737  026572             JSR    PC,WHICHU    ;DETERMINE THE UNIT #
2215   016704  013746  001202             MOV    $UNIT,-(SP)
2216   016710  104405                     TYPDS
2217   016712  104401  017550             TYPE   .VTMSG3      ;REPORT INTR. TO
2218   016716  013746  001360             MOV    VECTOR,-(SP) ;;SAVE VECTOR FOR TYPEOUT
 (1)   016722  104403                     TYPOS               ;;GO TYPE--OCTAL ASCII
 (1)   016724    003                      .BYTE  3            ;;TYPE 3 DIGIT(S)
 (1)   016725    001                      .BYTE  1            ;;TYPE LEADING ZEROS
2219   016726  104401  017601             TYPE   .VTMSG1      ;TYPE '' RECEIVED INTR. AT ''
2220   016732  013746  017072             MOV    TRTO,-(SP)   ;;SAVE TRTO FOR TYPEOUT
 (1)   016736  104403                     TYPOS               ;;GO TYPE--OCTAL ASCII
 (1)   016740    003                      .BYTE  3            ;;TYPE 3 DIGIT(S)
 (1)   016741    001                      .BYTE  1            ;;TYPE LEADING ZEROS
2221   016742  104401  017631             TYPE   .VTMSG2      ;TYPE 'RESTARTING TEST''
2222   016746  013777  001362  162404  4$:  MOV  VECTR1,@VECTOR
2223   016754  013777  001366  162402       MOV  VECTR3,@VECTR2
2224   016762  012777  004700  162372        MOV  #4700,@VECTR1
2225   016770  012777  004700  162370        MOV  #4700,@VECTR3
2226   016776  013737  017072  001360        MOV  TRTO,VECTOR
2227   017004  042737  000003  001360        BIC  #3,VECTOR
2228   017012  013737  001360  001362        MOV  VECTOR,VECTR1
2229   017020  062737  000002  001362        ADD  #2,VECTR1
2230   017026  013737  001360  001364        MOV  VECTOR,VECTR2
2231   017034  062737  000004  001364        ADD  #4,VECTR2
2232   017042  013737  001364  001366        MOV  VECTR2,VECTR3
2233   017050  062737  000002  001366        ADD  #2,VECTR3
2234   017056  005077  162270               CLR  @STREG
2235   017062  005777  162270               TST  @ADBUFF     ;READ A/D BUFFER TO CLEAR DONE FLAG
2236   017066  000177  162014               JMP  @$LPADR      ;START TEST OVER AGAIN.
2237   017072  000000             TRTO:     .WORD  0          ;CONTAINS ADDR. WE TRAPPED OR INTERRUPTED TO.
2238   017074  000000             TRFRO:    .WORD  0          ;CONTAINS ADDR. WE TRAPPED OR INTR. FROM.
```

```
2240                                                .SBTTL          ASCII MESSAGES
2241    017076    051600    040524    052122    SCHAN:  .ASCIZ  <200>\STARTING ON CHANNEL = \
        017104    047111    020107    047117
        017112    041440    040510    047116
        017120    046105    036440    000040
2242    017126    042600    042116    047111    ECHAN:  .ASCIZ  <200>\ENDING ON CHANNEL = \
        017134    020107    047117    041440
        017142    040510    047116    046105
        017150    036440    000040
2243    017154    005015    047516    051511    NOIMSG: .ASCIZ  <15><12>/NOISE TEST ON UNIT # /
        017162    020105    042524    052123
        017170    047440    020116    047125
        017176    052111    021440    000040
2244    017204    005015    042523    052124    SETMSG: .ASCIZ  <15><12>/SETTLING TEST ON UNIT # /
        017212    044514    043516    052040
        017220    051505    020124    047117
        017226    052440    044516    020124
        017234    020043    000
2245    017237    200       043117    051506    OFSET:  .ASCIZ  <200>/OFFSET TEST ON UNIT # /
        017244    052105    052040    051505
        017252    020124    047117    052440
        017260    044516    020124    020043
        017266    000
2246    017267    111       020123    044124    DWRF:   .ASCIZ  \IS THE MNCAD (A/D) TEST MODULE CONNECTED ? \
        017274    020105    047115    040503
        017302    020104    040450    042057
        017310    020051    042524    052123
        017316    046440    042117    046125
        017324    020105    047503    047116
        017332    041505    042524    020104
        017340    020077    000
2247    017343    055       000                 MINUS:  .BYTE   55,0
2248    017345    077       000                 QUEST:  .BYTE   77,0
2249    017347    040       020055    000       MDASH:  .ASCIZ  / - /
2250    017353    040       044523    043516    MSE:    .ASCIZ  / SINGLE ENDED/<15><12>
        017360    042514    042440    042116
        017366    042105    005015    000
2251    017373    040       044504    043106    MDIF:   .ASCIZ  / DIFFERENTIAL/<15><12>
        017400    051105    047105    044524
        017406    046101    005015    000
2252    017413    040       051120    040505    MPRMP:  .ASCIZ  / PREAMP/<15><12>
        017420    050115    005015    000
2253    017425    200       047115    040503    MADR:   .ASCIZ  <200>\MNCAD (A/D) BASE ADDRESS <\
        017432    020104    040450    042057
        017440    020051    040502    042523
        017446    040440    042104    042522
        017454    051523    036040    000
2254    017461    200       047115    040503    MVCT:   .ASCIZ  <200>\MNCAD (A/D) VECTOR ADDRESS <\
        017466    020104    040450    042057
        017474    020051    042526    052103
        017502    051117    040440    042104
        017510    042522    051523    036040
        017516    000
2255    017517    076       037440    000040    ENCOM:  .ASCIZ  #> ? #
```

```
2256   017524   046600   041516   042101   VTMSG:  .ASCIZ  <200>\MNCAD (A/D) UNIT #\
       017532   024040   027501   024504
       017540   052440   044516   020124
       017546      000043
2257   017550   005015   054105   042520   VTMSG3: .ASCIZ  <15><12>/EXPECTED INTERRUPT AT /
       017556   052103   042105   044440
       017564   052116   051105   052522
       017572   052120   040440   020124
       017600        000
2258   017601        040   042522   042503   VTMSG1: .ASCIZ  / RECEIVED INTERRUPT AT /
       017606   053111   042105   044440
       017614   052116   051105   052522
       017622   052120   040440   020124
       017630        000
2259   017631        200   046120   040505   VTMSG2: .ASCII  <200>/PLEASE CHECK VECTOR SWITCHES/
       017636   042523   041440   042510
       017644   045503   053040   041505
       017652   047524   020122   053523
       017660   052111   044103   051505
2260   017666   005015   051011   051505           .ASCIZ  <15><12>/       RESTARTING LOGIC TEST/<15><12>
       017674   040524   052122   047111
       017702   020107   047514   044507
       017710   020103   042524   052123
       017716   005015      000
2261   017721        015   052012   051505   TCHAN:  .ASCIZ  <15><12>/TESTING CHANNELS /
       017726   044524   043516   041440
       017734   040510   047116   046105
       017742   020123      000
2262   017745        124   050131   020105   YESNO:  .ASCIZ  /TYPE Y FOR YES, N FOR NO/<15><12>
       017752   020131   047506   020122
       017760   042531   026123   047040
       017766   043040   051117   047040
       017774   006517   000012
2263   020000   005015   042524   052123   TSTAD:  .ASCIZ  <15><12>/TESTING MNCAD/<15><12>
       020006   047111   020107   047115
       020014   040503   006504   000012
2264   020022   005015   042524   052123   TSTADM: .ASCIZ  <15><12>/TESTING MNCAM/<15><12>
       020030   047111   020107   047115
       020036   040503   006515   000012
2265   020044   042523   020124   047115   SADTST: .ASCIZ  #SET MNCAD (A/D) FRONT PANEL SWITCHES TO 'TEST'#<15><12>
       020052   040503   020104   040450
       020060   042057   020051   051106
       020066   047117   020124   040520
       020074   042516   020114   053523
       020102   052111   044103   051505
       020110   052040   020117   052042
       020116   051505   021124   005015
       020124        000
2266   020125        015   051412   052105   SDSE:   .ASCIZ  <15><12>/SET TEST MODULE TO SINGLE ENDED/<15><12>
       020132   052040   051505   020124
       020140   047515   052504   042514
       020146   052040   020117   044523
       020154   043516   042514   042440
       020162   042116   042105   005015
```

```
        020170      000
2267    020171      015    051412    052105    SDDIF:  .ASCIZ  <15><12>/SET TEST MODULE TO DIFFERENTIAL/<15><12>
        020176    052040    051505    020124
        020204    047515    052504    042514
        020212    052040    020117    044504
        020220    043106    051105    047105
        020226    044524    046101    005015
        020234      000
2268    020235      015    051412    052105    DIFM:   .ASCIZ  <15><12>/SET TEST MODULE ON CHANNELS UNDER TEST TO DIFFERENTIAL/<15><12>
        020242    052040    051505    020124
        020250    047515    052504    042514
        020256    047440    020116    044103
        020264    047101    042516    051514
        020272    052440    042116    051105
        020300    052040    051505    020124
        020306    047524    042040    043111
        020314    042506    042522    052116
        020322    040511    006514    000012
2269    020330    005015    051120    051505    EXTST:  .ASCIZ  <15><12>\PRESS EXTERNAL START ON MNCAD (A/D) TEST MODULE ON UNIT #\
        020336    020123    054105    042524
        020344    047122    046101    051440
        020352    040524    052122    047440
        020360    020116    047115    040503
        020366    020104    040450    042057
        020374    020051    042524    052123
        020402    046440    042117    046125
        020410    020105    047117    052440
        020416    044516    020124    000043
2270    020424    005015    030453    036465    TP15:   .ASCIZ  <15><12>/+15=/
        020432      000
2271    020433      015    026412    032461    TM15:   .ASCIZ  <15><12>/-15=/
        020440    000075
2272    020442    053040    046117    051524    VOLTS:  .ASCIZ  / VOLTS/
        020450      000
2273    020451      015    044412    050115    BADID:  .ASCIZ  <15><12>/IMPROPER I.D. CODE/
        020456    047522    042520    020122
        020464    027111    027104    041440
        020472    042117    000105
2274    020476    005015    047503    052116    YORNO:  .ASCIZ  <15><12>/CONTINUE TESTING? (Y FOR YES, N FOR NO):/
        020504    047111    042525    052040
        020512    051505    044524    043516
        020520    020077    054450    043040
        020526    051117    054440    051505
        020534    020054    020116    047506
        020542    020122    047516    035051
        020550      000
2275    020551      123    052105    046440    SCM:    .ASCIZ  /SET MODE TO CURRENT, /
        020556    042117    020105    047524
        020564    041440    051125    042522
        020572    052116    020054      000
2276    020577      123    052105    046440    SRM:    .ASCIZ  /SET MODE TO RESISTANCE, /
        020604    042117    020105    047524
        020612    051040    051505    051511
        020620    040524    041516    026105
```

```
        020626  000040
2277    020630  042523  020124  047515  SVM:    .ASCIZ  /SET MODE TO VOLTAGE, /
        020636  042504  052040  020117
        020644  047526  052114  043501
        020652  026105  000040
2278    020656  040507  047111  052040  GHLF:   .ASCIZ  /GAIN TO .5/<15><12>
        020664  020117  032456  005015
        020672     000
2279    020673     123  052105  043440  GAIN5:  .ASCIZ  /SET GAIN TO 5/<15><12>
        020700  044501  020116  047524
        020706  032440  005015     000
2280    020713     123  052105  043440  GAIN50: .ASCIZ  /SET GAIN TO 50/<15><12>
        020720  044501  020116  047524
        020726  032440  006460  000012
2281    020734  042523  020124  040507  GAIN5M: .ASCIZ  /SET GAIN TO 500/<15><12>
        020742  047111  052040  020117
        020750  030065  006460  000012
2282    020756  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
        020764     000
2283    020765     055  020055     000  DASH:   .ASCIZ  /-- /
2284    020771     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
        020776  026455  053440  042111
        021004  044124  005015     000
2285    021011     103  000110          CH:     .ASCIZ  /CH/
2286    021014  020040  020040     000  SPACE:  .ASCIZ  /    /
2287    021021     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
        021026  047117  041440  000110
2288    021034  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
        021042  047111  020107  051106
        021050  046517  041440  000110
2289    021056  040440  020124     000  ATMSG:  .ASCIZ  / AT /
2290    021063     122  051515  020040  RMSNOI: .ASCIZ  /RMS  NOISE /
        021070  047516  051511  020105
        021076     000
2291    021077     120  040505  020113  PKNOI:  .ASCIZ  /PEAK NOISE /
        021104  047516  051511  020105
        021112     000
2292    021113     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
        021120  040510  047116  046105
        021126  000040
2293    021130  000057                  SLASH:  .ASCIZ  #/#
2294    021132  020040  020040  045517  OKMSG:  .ASCIZ  /   OK/<15><12>
        021140  005015     000
2295    021143     015  052012  050131  CCHAN:  .ASCIZ  <15><12>/TYPE IN OCTAL CHANNEL NUMBER AND DEPRESS 'RETURN': /
        021150  020105  047111  047440
        021156  052103  046101  041440
        021164  040510  047116  046105
        021172  047040  046525  042502
        021200  020122  047101  020104
        021206  042504  051120  051505
        021214  020123  051042  052105
        021222  051125  021116  020072
        021230     000
2296    021231     015  052012  050131  SEL:    .ASCIZ  <15><12>/TYPE 'O' FOR OFFSET, 'G' FOR GAIN & DEPRESS 'RETURN': /
```

```
        021236  020105  047442  020042
        021244  047506  020122  043117
        021252  051506  052105  020054
        021260  043442  020042  047506
        021266  020122  040507  047111
        021274  023040  042040  050105
        021302  042522  051523  021040
        021310  042522  052524  047122
        021316  035042  000040
2297    021322  005015  042101  052512   XADJ:    .ASCII  <15><12>/ADJUST R83/
        021330  052123  051040  031470
2298    021336  043040  051117  030040   MOLSB:   .ASCII  / FOR 0.00 LSB ERROR/
        021344  030056  020060  051514
        021352  020102  051105  047522
        021360          122
2299    021361          015  042012  050105            .ASCIZ  <15><12>/DEPRESS 'RETURN' WHEN ADJUSTED/<15><12>
        021366  042522  051523  021040
        021374  042522  052524  047122
        021402  020042  044127  047105
        021410  040440  045104  051525
        021416  042524  006504  000012
2300    021424  005015  047111  052520   IGND:    .ASCII  <15><12>/INPUT A GROUND ON THE CHANNEL/ ;MUST BE JUST BEFORE ''CRWR''
        021432  020124  020101  051107
        021440  052517  042116  047440
        021446  020116  044124  020105
        021454  044103  047101  042516
        021462          114
2301    021463          015  042012  050105   CRWR:    .ASCIZ  <15><12>/DEPRESS 'RETURN' WHEN READY/<15><12>
        021470  042522  051523  021040
        021476  042522  052524  047122
        021504  020042  044127  047105
        021512  051040  040505  054504
        021520  005015          000
2302    021523          015  044412  050116   IVOLT:   .ASCIZ  <15><.2>/INPUT +5.115 VOLTS ON THE CHANNEL/
        021530  052125  025440  027065
        021536  030461  020065  047526
        021544  052114  020123  047117
        021552  052040  042510  041440
        021560  040510  047116  046105
        021566          000
2303    021567          015  040412  045104   YADJ:    .ASCIZ  <15><12>/ADJUST R84/
        021574  051525  020124  034122
        021602  000064
2304    021604  000053                          POSITV:  .ASCIZ  /+/
2305    021606  025040  042452  051122   ERMSG:   .ASCIZ  / **ERROR**/<15><12>
        021614  051117  025052  005015
        021622          000
2306    021623          040  045523  050111   SKPMSG:  .ASCIZ  / SKIPPED STATE(S)/
        021630  042520  020104  052123
        021636  052101  024105  024523
        021644          000
2307    021645          040  040516  051122   NARMSG:  .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
        021652  053517  024040  020074
        021660  027461  020062  051514
```

```
         021666  024502  051440  040524
         021674  042524  051450  006451
         021702  000012
2308     021704  053440  042111  020105    WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
         021712  037050  030440  030440
         021720  031057  046040  041123
         021726  020051  052123  052101
         021734  024105  024523  005015
         021742     000
2309     021743     040  052123  052101    OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
         021750  024105  024523  053440
         021756  042111  051105  052040
         021764  040510  020116  020062
         021772  051514  000102
2310     021776  051440  040524  042524    HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
         022004  053455  042111  044124
         022012  051450  020051  052517
         022020  051524  042111  020105
         022026  020053  051117  026440
         022034  030440  031057  046040
         022042  041123     000
2311     022045     015  042412  042116    MEND:   .ASCIZ  <15><12>/END OF LOGIC TESTS  /
         022052  047440  020106  047514
         022060  044507  020103  042524
         022066  052123  020123  000040
2312     022074  050200  047522  051107    FOUND1: .ASCIZ  <200>/PROGRAM DETECTED /
         022102  046501  042040  052105
         022110  041505  042524  020104
         022116     000
2313     022117     040  047115  040503    FOUND2: .ASCIZ  \ MNCAD (A/D)'S   \<15><12>
         022124  020104  040450  042057
         022132  023451  020123  020040
         022140  005015     000
2314     022143     012  005012  030453    MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
         022150  031057  046040  041123
         022156  005015  005012  005012
         022164  005012  005012  005012
         022172     012
2315     022173     055  027461  046062            .ASCIZ  \-1/2LSB\
         022200  041123     000
2316     022203     015  042012  043111    MSG20:  .ASCIZ  <15><12>/DIFFERENTIAL LINEARITY ON UNIT # /
         022210  042506  042522  052116
         022216  040511  020114  044514
         022224  042516  051101  052111
         022232  020131  047117  052440
         022240  044516  020124  020043
         022246     000
2317     022247     040  020040  020040    MSG16:  .ASCII  /                      STATE-WIDTH DISTRIBUTION/<15><12><12><12>
         022254  020040  020040  020040
         022262  020040  020040  020040
         022270  020040  051440  040524
         022276  042524  053455  042111
         022304  044124  042040  051511
         022312  051124  041111  052125
```

```
              022320  047511  006516  005012
              022326     012
      2318    022327     040  021440  047440        .ASCII  / # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
              022334  020106  052123  052101
              022342  051505  005012  005012
              022350  005012  005012  005012
              022356  005012  005012  005012
              022364  005012     012
      2319    022367     040  020040  020040        .ASCII  /                                        STATE WIDTH (LSB)/<15>
              022374  020040  020040  020040
              022402  020040  020040  020040
              022410  020040  020040  020040                                                              .
              022416  020040  020040  020040
              022424  020040  020040  020040
              022432  020040  020040  020040
              022440  020040  020040  020040
              022446  020040  052123  052101
              022454  020105  044527  052104
              022462  020110  046050  041123
              022470  006451     012
      2320    022473     040  020060  020040        .ASCIZ  # 0               1/2             1           1 1/2           2#
              022500  020040  020040  020040
              022506  020040  020040  030440
              022514  031057  020040  020040
              022522  020040  020040  020040
              022530  020040  030440  020040
              022536  020040  020040  020040
              022544  020040  020040  020061
              022552  027461  020062  020040
              022560  020040  020040  020040
              022566  020040  020040  000062
      2321    022574  042522  040514  044524  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
              022602  042526  040440  041503
              022610  051125  041501  035131
              022616  005015     000
      2322    022621     040  051514  020102  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
              022626  040515  044530  052515
              022634  020115  052101  000040
      2323    022642  005015  051120  047111  HEAD5:  .ASCII  <15><12>/PRINT VALUES--/
              022650  020124  040526  052514
              022656  051505  026455
      2324    022662  051440  052105  041440        .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
              022670  040510  047116  046105
              022676  044440  020116  053523
              022704  020122  047514  020127
              022712  054502  042524  005015
              022720     000
      2325    022721     040  052073  052117  ERRTOT: .ASCIZ  / ;TOTAL ERROR COUNT = /
              022726  046101  042440  051122
              022734  051117  041440  052517
              022742  052116  036440  000040
      2326    022750  035440  040502  020104  MESGD:  .ASCIZ  / ;BAD UNITS /
              022756  047125  052111  020123
              022764     000
```

```
2327  022765    033   015462  005110  C2:      .ASCIZ  <33><62><33><110><12>    ;EXIT GRAPH MODE, HOME AND LINE FEED
      022772    000
2328  022773    112      000           C3:      .ASCIZ  <112>                    ;ASCII "I"
2329  022775    015   047412  043106  MOFSET:  .ASCIZ  <15><12>/OFFSET =/
      023002  042523  020124  000075
2330  023010  046040  041123  000040  MLSB:    .ASCIZ  / LSB /
2331  023016  046040  041123  040440  MLSBAT:  .ASCIZ  / LSB AT /
      023024  020124     000
2332                                           ;CODE TO SETUP BIT MAP VIEWING SCREEN
2333  023027    033      061           BUFF1:   .BYTE   33,61                   ;GRAPH ON
2334  023031    101      061                    .BYTE   101,61                  ;ENABLE HISTOGRAM 0
2335  023033    111      062                    .BYTE   111,62                  ;CLEAR DATA + ENABLE VERT LINES
2336  023035    114      041     060            .BYTE   114,41,60               ;LOAD VERT LINE CORD
2337  023040    045      063                    .BYTE   45,63
2338  023042    051      066                    .BYTE   51,66
2339  023044    055      071                    .BYTE   55,71
2340  023046    061      074                    .BYTE   61,74
2341  023050    110      041     040            .BYTE   110,41,40               ;LOAD STARTING CORD.
2342  023053    112      000                    .BYTE   112,0                   ;LOAD GRAPH 1 COMMAND <DATA TO FOLLOW>
2343  023055    033      061           BUFF2:   .BYTE   33,61                   ;GRAPH ON
2344  023057    101      047                    .BYTE   101,47                  ;ENABLE GRAPH 0 AND 1
2345  023061    111      061                    .BYTE   111,61                  ;ENABLE DISPLAY
2346  023063    104      050     065            .BYTE   104,50,65,44,62 ;LOAD HORIZ CORDINATES
      023066    044      062
2347  023070    110      040     040            .BYTE   110,40,40               ;LOAD STARTING GRAPH CORD.
2348  023073    102      000                    .BYTE   102,0                   ;LOAD GRAPH 0 <DATA TO FOLLOW>
2349  023075    033      061           INITVT:  .BYTE   33,61                   ;GRAPH ON
2350  023077    101      040     040            .BYTE   101,40,40               ;DISABLE SCREEN
2351  023102    111      060     040            .BYTE   111,60,40               ;SET RECTANGEL ASPECT RATIO
2352  023105    033      062                    .BYTE   33,62                   ;EXIT GRAPH MODE
2353                                  ;        .BYTE   33,133,77,62,105        ;ENSURE "ASCII" <CAUSES HOLD SCREEN ON VT55>
2354  023107    033      110                    .BYTE   33,110                  ;"HOME"
2355  023111    033      112     000            .BYTE   33,112,0                ;"ERASE SCREEN"
2356  023114    033      110           VTINIT:  .BYTE   33,110                  ;"HOME"
2357  023116    033      112                    .BYTE   33,112                  ;"ERASE SCREEN"
2358  023120    033      061                    .BYTE   33,61                   ;ENTER GRAPHIC MODE
2359  023122    101      040                    .BYTE   101,40                  ;CLEAR GRAPH DATA
2360  023124    033      062     000            .BYTE   33,62,0                 ;EXIT GRAPHIC MODE
2361
2362
2363                                           .SBTTL  ASCII TEXT MESSAGES
2364
2365  023127    200   020114  020075  PRIME1:  .ASCII  <200>/L = LOGIC TEST/
      023134  047514  044507  020103
      023142  042524  052123
2366  023146  053600  036440  053440           .ASCII  <200>/W = WRAPAROUND ANALOG TEST/
      023154  040522  040520  047522
      023162  047125  020104  047101
      023170  046101  043517  052040
      023176  051505     124
2367  023201    200   020101  020075           .ASCII  <200>/A = AUTO TEST/
      023206  052501  047524  052040
      023214  051505     124
2368  023217    200   020116  020075           .ASCII  <200>/N = NOISE TESTS ON SELECTED CHANNELS/
```

```
        023224  047516  051511  020105
        023232  042524  052123  020123
        023240  047117  051440  046105
        023246  041505  042524  020104
        023254  044103  047101  042516
        023262  051514
2369    023264  053200  036440  053040        .ASCII   <200>/V = VIEDO BIT MAP OUTPUT AVAILABLE (IE VT105, VT55)/
        023272  042511  047504  041040
        023300  052111  046440  050101
        023306  047440  052125  052520
        023314  020124  053101  044501
        023322  040514  046102  020105
        023330  044450  020105  052126
        023336  030061  026065  053040
        023344  032524  024465
2370    023350  050200  036440  050040        .ASCII   <200>/P = PRINT CONVERTED ANALOG VALUE LOOP/
        023356  044522  052116  041440
        023364  047117  042526  052122
        023372  042105  040440  040516
        023400  047514  020107  040526
        023406  052514  020105  047514
        023414  050117
2371    023416  041600  036440  041440        .ASCII   <200>/C = CALIBRATION LOOP FOR MNCAD/
        023424  046101  041111  040522
        023432  044524  047117  046040
        023440  047517  020120  047506
        023446  020122  047115  040503
        023454          104
2372    023455     200  020102  020075        .ASCII   <200>/B = BASE AND VECTOR ADDRESS CHANGES/
        023462  040502  042523  040440
        023470  042116  053040  041505
        023476  047524  020122  042101
        023504  051104  051505  020123
        023512  044103  047101  042507
        023520          123
2373    023521     200  020107  020075        .ASCII   <200>/G = GET NEW SWITCH REGISTER VALUE/
        023526  042507  020124  042516
        023534  020127  053523  052111
        023542  044103  051040  043505
        023550  051511  042524  020122
        023556  040526  052514     105
2374    023563     200  020110  020075        .ASCIZ   <200>/H = HELP THE OPERATOR AND RETYPE THIS LIST      /
        023570  042510  050114  052040
        023576  042510  047440  042520
        023604  040522  047524  020122
        023612  047101  020104  042522
        023620  054524  042520  052040
        023626  044510  020123  044514
        023634  052123  020040  020040
        023642     000
2375    023643     015     012        DOT:    .BYTE    15,12
2376    023645     124  050131  020105        .ASCIZ   /TYPE THE "TEST CHARACTER" THEN DEPRESS "RETURN KEY"  /
        023652  044124  020105  052042
        023660  051505  020124  044103
```

```
        023666  051101  041501  042524
        023674  021122  052040  042510
        023702  020116  042504  051120
        023710  051505  020123  051042
        023716  052105  051125  020116
        023724  042513  021131  020040
        023732     000
2377    023733     115  041516  042101  EM1:   .ASCIZ  \MNCAD (A/D)     STATUS REG. ERROR\
        023740  024040  027501  024504
        023746  051411  040524  052524
        023754  020123  042522  027107
        023762  042440  051122  051117
        023770     000
2378    023771     115  041516  042101  EM2:   .ASCIZ  \MNCAD (A/D)     FAILED TO INTERRUPT\
        023776  024040  027501  024504
        024004  043011  044501  042514
        024012  020104  047524  044440
        024020  052116  051105  052522
        024026  052120     000
2379    024031     115  041516  042101  EM3:   .ASCIZ  \MNCAD (A/D)     UNEXPECTED INTERRUPT\
        024036  024040  027501  024504
        024044  052411  042516  050130
        024052  041505  042524  020104
        024060  047111  042524  051122
        024066  050125  000124
2380    024072  047115  040503  020104  EM4:   .ASCIZ  #MNCAD (A/D)     ERROR ON A/D CHANNEL#
        024100  040450  042057  004451
        024106  051105  047522  020122
        024114  047117  040440  042057
        024122  041440  040510  047116
        024130  046105     000
2381    024133     115  041516  042101  EM5:   .ASCIZ  \MNCAD (A/D)     EXISTING MNCAD NOW FAIL'S TO RESPOND\
        024140  024040  027501  024504
        024146  042411  044530  052123
        024154  047111  020107  047115
        024162  040503  020104  047516
        024170  020127  040506  046111
        024176  051447  052040  020117
        024204  042522  050123  047117
        024212  000104
2382    024214  047115  040503  020104  EM6:   .ASCIZ  \MNCAD (A/D)     DOES NOT EXIST <BUS ERROR> CHECK ADDRESS SWITCHES\
        024222  040450  042057  004451
        024230  047504  051505  047040
        024236  052117  042440  044530
        024244  052123  036040  052502
        024252  020123  051105  047522
        024260  037122  041440  042510
        024266  045503  040440  042104
        024274  042522  051523  051440
        024302  044527  041524  042510
        024310  000123
2383    024312  047125  052111  042411  DH1:   .ASCIZ  /UNIT    ERRPC    STREG  EXPECTED ACTUAL/
        024320  051122  041520  020040
        024326  051440  051124  043505
```

```
        024334  020040  054105  042520
        024342  052103  042105  040440
        024350  052103  040525  000114
2384    024356  047125  052111  042411  DH2:    .ASCIZ  /UNIT    ERRPC    STREG    CHANNEL NOMINAL  TOL.    ACTUAL/
        024364  051122  041520  020040
        024372  051440  051124  043505
        024400  020040  041440  040510
        024406  047116  046105  047040
        024414  046517  047111  046101
        024422  020040  047524  027114
        024430  020040  040440  052103
        024436  040525  000114
2385    024442  047125  052111  042411  DH3:    .ASCIZ  /UNIT    ERRPC    STREG    ACTUAL/
        024450  051122  041520  020040
        024456  051440  051124  043505
        024464  020040  040440  052103
        024472  040525  000114
2386    024476  047125  052111  042411  DH5:    .ASCIZ  /UNIT    ERRPC    WERE     ARE/
        024504  051122  041520  053411
        024512  051105  004505  051101
        024520  000105
2387    024522  047125  052111  042411  DH6:    .ASCIZ  /UNIT    ERRPC    STREG/
        024530  051122  041520  051411
        024536  051124  043505     000
2388    024543     000                   THOUS:  .BYTE   0
2389    024544     000                   HUNS:   .BYTE   0
2390    024545     056                   DECPNT: .BYTE   56
2391    024546     000                   TENS:   .BYTE   0
2392    024547     000     000           ONES:   .BYTE   0,0
2393            024552                    .EVEN                .
2394
2395    024552  001514  001116  001352  DT1:    UNITBD,$ERRPC, STREG, $GDDAT, $BDDAT,0
        024560  001124  001126  000000
2396    024566  001514  001116  001352  DT2:    UNITBD,$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
        024574  001444  001124  001462
        024602  001126  000000
2397    024606  001514  001116  001352  DT3:    UNITBD,$ERRPC,STREG,$BDDAT,0
        024614  001126  000000
2398    024620  001514  001116  001202  DT5:    UNITBD,$ERRPC,$UNIT,TEMP,0
        024626  001426  000000
2399    024632  001514  001116  001352  DT6:    UNITBD,$ERRPC,STREG,0
        024640  000000
2400    024642     000     000     000  DF1:    .BYTE   0,0,0,0,0,0,0,0
        024645     000     000     000
        024650     000     000
```

```
2402                                        .SBTTL  TTY INPUT ROUTINE
 (1)
 (2)                                 ;;************************************************************************
 (1)                                        .ENABL  LSB
 (1)  024652  000000                 $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
 (1)  024654  000000                 $TKQIN: .WORD   0               ;;INPUT POINTER
 (1)  024656  000000                 $TKQOUT: .WORD  0               ;;OUTPUT POINTER
 (1)  024660  000040                 $TKQSRT: .BLKB  32.             ;;TTY KEYBOARD QUEUE
 (1)          024720                  $TKQEND=.
 (1)
 (1)                                 ;*TK INITIALIZE ROUTINE
 (1)                                 ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 (1)                                 ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 (1)                                 ;
 (1)                                 ;*CALL:
 (1)                                 ;*      JSR     PC,$TKINT
 (1)                                 ;*      RETURN
 (1)                                 ;
 (1)  024720  005037  024652         $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 (1)  024724  012737  024660  024654         MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 (1)  024732  013737  024654  024656         MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 (1)  024740  012737  024770  000060         MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 (1)  024746  012737  000200  000062         MOV     #200,@#TKVEC+2  ;;'BR'' LEVEL 4
 (1)  024754  005777  154166                 TST     @$TKB           ;;CLEAR DONE FLAG
 (1)  024760  012777  000100  154156         MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 (1)  024766  000207                         RTS     PC              ;;RETURN TO CALLER
 (1)
 (1)                                 ;*TK SERVICE ROUTINE
 (1)                                 ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 (1)                                 ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 (1)                                 ;*IT IN THE QUEUE.
 (1)                                 ;*IF THE CHARACTER IS A ''CONTROL-C'' (^C) $TKINT IS CALLED AND
 (1)                                 ;*UPON RETURN EXIT IS MADE TO THE ''CONTROL-C'' RESTART ADDRESS (BEG2)
 (1)                                 ;
 (1)  024770  117746  154152         $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 (1)  024774  042716  177600                 BIC     #^C177,(SP)     ;;STRIP THE JUNK
 (1)  025000  021627  000003                 CMP     (SP),#3         ;;IS IT A CONTROL C?
 (1)  025004  001007                         BNE     1$              ;;BRANCH IF NO
 (1)  025006  104401  026140                 TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
 (1)  025012  004737  024720                 JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 (1)  025016  005726                         TST     (SP)+           ;;CLEAN UP STACK
 (1)  025020  000137  001564                 JMP     BEG2            ;;CONTROL C RESTART
 (1)  025024  021627  000007         1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 (1)  025030  001004                         BNE     2$              ;;BRANCH IF NO
 (1)  025032  022737  000176  001140         CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 (1)  025040  001500                         BEQ     6$              ;;GO TO SWR CHANGE
 (1)
 (1)  025042                         2$:
 (1)  025042  022737  000040  024652         CMP     #32.,$TKCNT     ;;IS THE QUEUE FULL?
 (1)  025050  001004                         BNE     3$              ;;BRANCH IF NO
 (1)  025052  104401  026134                 TYPE    ,$BELL          ;;RING THE TTY BELL
 (1)  025056  005726                         TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 (1)  025060  000451                         BR      5$              ;;EXIT
 (1)  025062  021627  000023         3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
```

```
(1)  025066  001021                        BNE     32$          ;;BRANCH IF NO
(1)  025070  005077  154050                CLR     @$TKS        ;;DISABLE TTY KEYBOARD INTERRUPTS
(1)  025074  005726                        TST     (SP)+        ;;CLEAN CHAR OFF STACK
(1)  025076  105777  154042        31$:    TSTB    @$TKS        ;;WAIT FOR A CHAR
(1)  025102  100375                        BPL     31$          ;;LOOP UNTIL ITS THERE
(1)  025104  117746  154036                MOVB    @$TKB,-(SP)  ;;GET THE CHARACTER
(1)  025110  042716  177600                BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
(1)  025114  022627  000021                CMP     (SP)+,#21    ;;IS IT A CONTROL-Q?
(1)  025120  001366                        BNE     31$          ;;BRANCH IF NO
(1)  025122  012777  000100  154014        MOV     #100,@$TKS   ;;REENABLE TTY KEYBOARD INTERRUPTS
(1)  025130  000002                        RTI                  ;;RETURN
(1)  025132  005237  024652        32$:    INC     $TKCNT       ;;COUNT THIS CHARACTER
(1)  025136  021627  000140                CMP     (SP),#140    ;;IS IT UPPER CASE?
(1)  025142  002405                        BLT     4$           ;;BRANCH IF YES
(1)  025144  021627  000175                CMP     (SP),#175    ;;IS IT A SPECIAL CHAR?
(1)  025150  003002                        BGT     4$           ;;BRANCH IF YES
(1)  025152  042716  000040                BIC     #40,(SP)     ;;MAKE IT UPPER CASE
(1)  025156  112677  177472        4$:     MOVB    (SP)+,@$TKQIN ;;AND PUT IT IN QUEUE
(1)  025162  005237  024654                INC     $TKQIN       ;;UPDATE THE POINTER
(1)  025166  023727  024654  024720        CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
(1)  025174  001003                        BNE     5$           ;;BRANCH IF NO
(1)  025176  012737  024660  024654        MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
(1)  025204  000002                5$:     RTI                  ;;RETURN
(1)
(2)                                 ;;**************************************************************
(1)                                 ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)                                 ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
(1)                                 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
(1)                                 ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
(1)  025206  022737  000176  001140 $CKSWR: CMP     #SWREG,SWR   ;;IS THE SOFT-SWR SELECTED
(1)  025214  001124                        BNE     15$          ;;EXIT IF NOT
(1)  025216  105777  153722                TSTB    @$TKS        ;;IS A CHAR WAITING?
(1)  025222  100121                        BPL     15$          ;;IF NOT, EXIT
(1)  025224  117746  153716                MOVB    @$TKB,-(SP)  ;;YES
(1)  025230  042716  177600                BIC     #^C177,(SP)  ;;MAKE IT  7-BIT ASCII
(1)  025234  021627  000007                CMP     (SP),#7      ;;IS IT A CONTROL-G?
(1)  025240  001300                        BNE     2$           ;;IF NOT, PUT IT IN THE TTY QUEUE
(1)                                                              ;;AND EXIT
(1)
(2)                                 ;;**************************************************************
(1)                                 ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
(1)                                 ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
(1)                                 ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
(1)  025242  123727  001134  000001 6$:     CMPB    $AUTOB,#1    ;;ARE WE RUNNING IN AUTO-MODE?
(1)  025250  001674                        BEQ     2$           ;;BRANCH IF YES
(1)  025252  005726                        TST     (SP)+        ;;CLEAR CONTROL-G OFF STACK
(1)  025254  004737  024720                JSR     PC,$TKINT    ;;FLUSH THE TTY INPUT QUEUE
(1)  025260  005077  153660                CLR     @$TKS        ;;DISABLE TTY KEYBOARD INTERRUPTS
(1)  025264  112737  000001  001135        MOVB    #1,$INTAG    ;;SET INTERRUPT MODE INDICATOR
(1)
(1)  025272  104401  026152                TYPE    ,$CNTLG      ;;ECHO THE CONTROL-G (^G)
(1)  025276  104401  026157        $GTSWR: TYPE    ,$MSWR       ;;TYPE CURRENT CONTENTS
(2)  025302  013746  000176                MOV     SWREG,-(SP)  ;;SAVE SWREG FOR TYPEOUT
(2)  025306  104402                        TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
      (1)  025310  104401  026170                    TYPE    .$MNEW          ;;PROMPT FOR NEW SWR
      (1)  025314  005046                     19$:   CLR     -(SP)           ;;CLEAR COUNTER
      (1)  025316  005046                            CLR     -(SP)           ;;THE NEW SWR
      (1)  025320  105777  153620              7$:   TSTB    @$TKS           ;;CHAR THERE?
      (1)  025324  100375                            BPL     7$              ;;IF NOT TRY AGAIN
      (1)
      (1)  025326  117746  153614                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
      (1)  025332  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
      (1)
      (1)  025336  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL-C?
      (1)  025342  001015                            BNE     9$              ;;BRANCH IF NOT
      (1)  025344  104401  026140                    TYPE    .$CNTLC         ;;YES, ECHO CONTROL-C (^C)
      (1)  025350  062706  000006                    ADD     #6,SP           ;;CLEAN UP STACK
      (1)  025354  123727  001135  000001            CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
      (1)  025362  001003                            BNE     8$              ;;BRANCH IF NO
      (1)  025364  012777  000100  153552            MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
      (1)  025372  000137  001564              8$:   JMP     BEG2            ;;CONTROL-C RESTART
      (1)
      (1)
      (1)  025376  021627  000025              9$:   CMP     (SP),#25        ;;IS IT A CONTROL-U?
      (1)  025402  001005                            BNE     10$             ;;BRANCH IF NOT
      (1)  025404  104401  026145                    TYPE    .$CNTLU         ;;YES, ECHO CONTROL-U (^U)
      (1)  025410  062706  000006             20$:   ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
      (1)  025414  000737                            BR      19$             ;;LET'S TRY IT AGAIN
      (1)
      (1)
      (1)  025416  021627  000015             10$:   CMP     (SP),#15        ;;IS IT A <CR>?
      (1)  025422  001022                            BNE     16$             ;;BRANCH IF NO
      (1)  025424  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
      (1)  025430  001403                            BEQ     11$             ;;BRANCH IF YES
      (1)  025432  016677  000002  153500            MOV     2(SP),@SWR      ;;SAVE NEW SWR
      (1)  025440  062706  000006             11$:   ADD     #6,SP           ;;CLEAR UP STACK
      (1)  025444  104401  001165             14$:   TYPE    .$CRLF          ;;ECHO <CR> AND <LF>
      (1)  025450  123727  001135  000001            CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
      (1)  025456  001003                            BNE     15$             ;;BRANCH IF NOT
      (1)  025460  012777  000100  153456            MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
      (1)  025466  000002                     15$:   RTI                     ;;RETURN
      (1)  025470  004737  027622             16$:   JSR     PC,$TYPEC       ;;ECHO CHAR
      (1)  025474  021627  000060                    CMP     (SP),#60        ;;CHAR < 0?
      (1)  025500  002420                            BLT     18$             ;;BRANCH IF YES
      (1)  025502  021627  000067                    CMP     (SP),#67        ;;CHAR > 7?
      (1)  025506  003015                            BGT     18$             ;;BRANCH IF YES
      (1)  025510  042726  000060                    BIC     #60,(SP)+       ;;STRIP-OFF ASCII
      (1)  025514  005766  000002                    TST     2(SP)           ;;IS THIS THE FIRST CHAR
      (1)  025520  001403                            BEQ     17$             ;;BRANCH IF YES
      (1)  025522  006316                            ASL     (SP)            ;;NO, SHIFT PRESENT
      (1)  025524  006316                            ASL     (SP)            ;;    CHAR OVER TO MAKE
      (1)  025526  006316                            ASL     (SP)            ;;    ROOM FOR NEW ONE.
      (1)  025530  005266  000002             17$:   INC     2(SP)           ;;KEEP COUNT OF CHAR
      (1)  025534  056616  177776                    BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
      (1)  025540  000667                            BR      7$              ;;GET THE NEXT ONE
      (1)  025542  104401  001164             18$:   TYPE    .$QUES          ;;TYPE ?<CR><LF>
      (1)  025546  000720                            BR      20$             ;;SIMULATE CONTROL-U
      (1)                                     .DSABL  LSB
```

```
(1)
(1)
(2)                              ;;***********************************************************
(1)                              ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                              ;*CALL:
(1)                              ;*      RDCHR                     ;;GET A CHARACTER FROM THE QUEUE
(1)                              ;*      RETURN HERE               ;;CHARACTER IS ON THE STACK
(1)                              ;*                               ;;WITH PARITY BIT STRIPPED OFF
(1)                              ;
(1)
(1)  025550  011646             $RDCHR: MOV    (SP),-(SP)         ;;PUSH DOWN THE PC AND
(1)  025552  016666  000004  000002    MOV    4(SP),2(SP)        ;;THE PS
(1)  025560  005066  000004            CLR    4(SP)              ;;GET READY FOR A CHARACTER
(2)  025564  005046                    CLR    -(SP)              ;;PUT NEW PS ON STACK
(2)  025566  012746  025574            MOV    #64$,-(SP)         ;;PUT NEW PC ON  STACK
(2)  025572  000002                    RTI                       ;;POP NEW PC AND PS
(2)  025574                     64$:
(1)  025574  005737  024652     1$:    TST    $TKCNT             ;;WAIT ON A CHARACTER
(1)  025600  001775                    BEQ    1$
(1)  025602  005337  024652            DEC    $TKCNT             ;;DECREMENT THE COUNTER
(1)  025606  117766  177044  000004    MOVB   @$TKQOUT,4(SP)     ;;GET ONE CHARACTER
(1)  025614  005237  024656            INC    $TKQOUT            ;;UPDATE THE POINTER
(1)  025620  023727  024656  024720    CMP    $TKQOUT,#$TKQEND   ;;DID IT GO OFF OF THE END?
(1)  025626  001003                    BNE    2$                 ;;BRANCH IF NO
(1)  025630  012737  024660  024656    MOV    #$TKQSRT,$TKQOUT   ;;RESET THE POINTER
(1)  025636  000002             2$:    RTI                       ;;RETURN
(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)  025640  010346             $RDLIN: MOV    R3,-(SP)          ;;SAVE R3
(1)  025642  005046                    CLR    -(SP)              ;;CLEAR THE RUBOUT KEY
(1)  025644  012703  026074     1$:    MOV    #$TTYIN,R3         ;;GET ADDRESS
(1)  025650  022703  026134     2$:    CMP    #$TTYIN+32.,R3     ;;BUFFER FULL?
(1)  025654  101456                    BLOS   4$                 ;;BR IF YES
(1)  025656  104411                    RDCHR                     ;;GO READ ONE CHARACTER FROM THE TTY
(1)  025660  112613                    MOVB   (SP)+,(R3)         ;;GET CHARACTER
(1)  025662  122713  000177     10$:   CMPB   #177,(R3)          ;;IS IT A RUBOUT
(1)  025666  001022                    BNE    5$                 ;;BR IF NO
(1)  025670  005716                    TST    (SP)               ;;IS THIS THE FIRST RUBOUT?
(1)  025672  001007                    BNE    6$                 ;;BR IF NO
(1)  025674  112737  000134  026072    MOVB   #'\,9$             ;;TYPE A BACK SLASH
(1)  025702  104401  026072            TYPE   ,9$
(1)  025706  012716  177777            MOV    #-1,(SP)           ;;SET THE RUBOUT KEY
(1)  025712  005303             6$:    DEC    R3                 ;;BACKUP BY ONE
(1)  025714  020327  026074            CMP    R3,#$TTYIN         ;;STACK EMPTY?
(1)  025720  103434                    BLO    4$                 ;;BR IF YES
(1)  025722  111337  026072            MOVB   (R3),9$            ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)  025726  104401  026072            TYPE   ,9$                ;;GO TYPE
(1)  025732  000746                    BR     2$                 ;;GO READ ANOTHER CHAR.
(1)  025734  005716             5$:    TST    (SP)               ;;RUBOUT KEY SET?
```

```
(1)  025736  001406                         BEQ     7$              ;;BR IF NO
(1)  025740  112737  000134  026072         MOVB    #'\,9$          ;;TYPE A BACK SLASH
(1)  025746  104401  026072                 TYPE    ,9$             ;;TYPE A BACK SLASH
(1)  025752  005016                         CLR     (SP)            ;;CLEAR THE RUBOUT KEY
(1)  025754  122713  000025         7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
(1)  025760  001003                         BNE     8$              ;;BR IF NO
(1)  025762  104401  026145                 TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
(1)  025766  000726                         BR      1$              ;;GO START OVER
(1)  025770  122713  000022         8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
(1)  025774  001011                         BNE     3$              ;;BRANCH IF NO
(1)  025776  105013                         CLRB    (R3)            ;;CLEAR THE CHARACTER
(1)  026000  104401  001165                 TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
(1)  026004  104401  026074                 TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
(1)  026010  000717                         BR      2$              ;;GO PICKUP ANOTHER CHACTER
(1)  026012  104401  001164         4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  026016  000712                         BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  026020  111337  026072         3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  026024  104401  026072                 TYPE    ,9$
(1)  026030  122723  000015                 CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  026034  001305                         BNE     2$              ;;LOOP IF NOT RETURN
(1)  026036  105063  177777                 CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  026042  104401  001166                 TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  026046  005726                         TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
(1)  026050  012603                         MOV     (SP)+,R3        ;;RESTORE R3
(1)  026052  011646                         MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  026054  016666  000004  000002         MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  026062  012766  026074  000004         MOV     #$TTYIN,4(SP)
(1)  026070  000002                         RTI                     ;;RETURN
(1)  026072    000             9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  026073    000                     .BYTE   0               ;;TERMINATOR
(1)  026074  000040             $TTYIN: .BLKB   32.             ;;RESERVE 32. BYTES FOR TTY INPUT
(1)  026134  177607  000377     $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
(1)  026140  041536  005015   000  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
(1)  026145    136   006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
(1)  026152  043536  005015   000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
(1)  026157    015   051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  026164  036440  000040
(1)  026170  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
(1)  026176  020075    000
(1)          026202                         .EVEN
```

```
2404                                     .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)  026202  011646           $RDOCT:  MOV      (SP),-(SP)              ;;PROVIDE SPACE FOR THE
  (1)  026204  016666  000004  000002    MOV      4(SP),2(SP)             ;;INPUT NUMBER
  (3)  026212  010046           MOV      RO,-(SP)                ;;PUSH RO ON STACK
  (3)  026214  010146           MOV      R1,-(SP)                ;;PUSH R1 ON STACK
  (3)  026216  010246           MOV      R2,-(SP)                ;;PUSH R2 ON STACK
  (1)  026220  104412           1$:      RDLIN                            ;;READ AN ASCIZ LINE
  (1)  026222  012600           MOV      (SP)+,RO                ;;GET ADDRESS OF 1ST CHARACTER
  (1)  026224  005001           CLR      R1                      ;;CLEAR DATA WORD
  (1)  026226  005002           CLR      R2
  (1)  026230  112046           2$:      MOVB     (RO)+,-(SP)             ;;PICKUP THIS CHARACTER
  (1)  026232  001412           BEQ      3$                      ;;IF ZERO GET OUT
  (1)  026234  006301           ASL      R1                      ;;*2
  (1)  026236  006102           ROL      R2
  (1)  026240  006301           ASL      R1                      ;;*4
  (1)  026242  006102           ROL      R2
  (1)  026244  006301           ASL      R1                      ;;*8
  (1)  026246  006102           ROL      R2
  (1)  026250  042716  177770   BIC      #^C7,(SP)               ;;STRIP THE ASCII JUNK
  (1)  026254  062601           ADD      (SP)+,R1                ;;ADD IN THIS DIGIT
  (1)  026256  000764           BR       2$                      ;;LOOP
  (1)  026260  005726           3$:      TST      (SP)+                   ;;CLEAN TERMINATOR FROM STACK
  (1)  026262  010166  000012   MOV      R1,12(SP)               ;;SAVE THE RESULT
  (1)  026266  010237  026302   MOV      R2,$HIOCT
  (3)  026272  012602           MOV      (SP)+,R2                ;;POP STACK INTO R2
  (3)  026274  012601           MOV      (SP)+,R1                ;;POP STACK INTO R1
  (3)  026276  012600           MOV      (SP)+,RO                ;;POP STACK INTO RO
  (1)  026300  000002           RTI                              ;;RETURN
  (1)  026302  000000           $HIOCT:  .WORD    0                       ;;HIGH ORDER BITS GO HERE
```

```
 2406                                    .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)  026304                    $SCOPE:
 (1)  026304  104410                     CKSWR                          ;;TEST  FOR CHANGE IN SOFT-SWR
 (1)  026306  032777  040000  152624  1$: BIT    #BIT14,@SWR            ;;LOOP ON PRESENT TEST?
 (1)  026314  001114                     BNE     $OVER                  ;;YES IF SW14=1
 (1)                            ;#####START OF CODE FOR THE XOR TESTER#####
 (1)  026316  000416           $XTSTR: BR       6$                     ;;IF RUNNING ON THE 'XOR'' TESTER CHANGE
 (1)                                                                    ;;THIS INSTRUCTION TO A 'NOP'' (NOP=240)
 (1)  026320  013746  000004            MOV      @#ERRVEC,-(SP)         ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 (1)  026324  012737  026344  000004    MOV      #5$,@#ERRVEC           ;;SET FOR TIMEOUT
 (1)  026332  005737  177060            TST      @#177060               ;;TIME OUT ON XOR?
 (1)  026336  012637  000004            MOV      (SP)+,@#ERRVEC         ;;RESTORE THE ERROR VECTOR
 (1)  026342  000463                    BR       $SVLAD                 ;;GO TO THE NEXT TEST
 (1)  026344  022626            5$:     CMP      (SP)+,(SP)+            ;;CLEAR THE STACK AFTER A TIME OUT
 (1)  026346  012637  000004            MOV      (SP)+,@#ERRVEC         ;;RESTORE THE ERROR VECTOR
 (1)  026352  000423                    BR       7$                     ;;LOOP ON THE PRESENT TEST
 (1)  026354                    6$:;#####END OF CODE FOR THE XOR TESTER#####
 (1)  026354  032777  000400  152556    BIT      #BIT08,@SWR            ;;LOOP ON SPEC. TEST?
 (1)  026362  001404                    BEQ      2$                     ;;BR IF NO
 (1)  026364  127737  152550  001102    CMPB     @SWR,$TSTNM            ;;ON THE RIGHT TEST?     SWR<7:0>
 (1)  026372  001465                    BEQ      $OVER                  ;;BR IF YES
 (1)  026374  105737  001103    2$:     TSTB     $ERFLG                 ;;HAS AN ERROR OCCURRED?
 (1)  026400  001421                    BEQ      3$                     ;;BR IF NO
 (1)  026402  123737  001115  001103    CMPB     $ERMAX,$ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 (1)  026410  101015                    BHI      3$                     ;;BR IF NO
 (1)  026412  032777  001000  152520    BIT      #BIT09,@SWR            ;;LOOP ON ERROR?
 (1)  026420  001404                    BEQ      4$                     ;;BR IF NO
 (1)  026422  013737  001110  001106  7$: MOV    $LPERR,$LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
 (1)  026430  000446                    BR       $OVER
 (1)  026432  105037  001103    4$:     CLRB     $ERFLG                 ;;ZERO THE ERROR FLAG
 (1)  026436  005037  001160            CLR      $TIMES                 ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 (1)  026442  000415                    BR       1$                     ;;ESCAPE TO THE NEXT TEST
 (1)  026444  032777  004000  152466  3$: BIT    #BIT11,@SWR            ;;INHIBIT ITERATIONS?
 (1)  026452  001011                    BNE      1$                     ;;BR IF YES
 (1)  026454  005737  001176            TST      $PASS                  ;;IF FIRST PASS OF PROGRAM
 (1)  026460  001406                    BEQ      1$                     ;;       INHIBIT ITERATIONS
 (1)  026462  005237  001104            INC      $ICNT                  ;;INCREMENT ITERATION COUNT
 (1)  026466  023737  001160  001104    CMP      $TIMES,$ICNT           ;;CHECK THE NUMBER OF ITERATIONS MADE
 (1)  026474  002024                    BGE      $OVER                  ;;BR IF MORE ITERATION REQUIRED
 (1)  026 76  012737  000001  001104  1$: MOV    #1,$ICNT               ;;REINITIALIZE THE ITERATION COUNTER
 (1)  026504  013737  026562  001160    MOV      $MXCNT,$TIMES          ;;SET NUMBER OF ITERATIONS TO DO
```

```
  (1)  026512  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
  (1)  026516  113737  001102  001174          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
  (1)  026524  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
  (1)  026530  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
  (1)  026534  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
  (1)  026540  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
  (1)  026546  013777  001102  152366  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
  (1)  026554  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
  (1)  026560  000002                          RTI                     ;;FIXES PS
  (1)  026562  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
 2407  026564  053737  001512  001510  WHICHV: BIS     MASKNM,BADUNT   ;SET CURRENT UNIT INTO BAD FIELD
 2408  026572  013737  001512  026622  WHICHU: MOV     MASKNM,11$      ;GET CURRENT UNIT
 2409  026600  012737  000000  001514          MOV     #0,UNITBD       ;PRIME THE VALUE
 2410  026606  006237  026622          10$:    ASR     11$             ;CONVERT
 2411  026612  001404                          BEQ     12$             ;BR WHEN DONE
 2412  026614  005237  001514                  INC     UNITBD          ;BUMP POINTER
 2413  026620  000772                          BR      10$
 2414  026622  000000          11$:    0
 2415  026624  000207          12$:    RTS     PC                      ;EXIT
 2419                          .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)                          ;;*SW09=1         LOOP ON ERROR
  (1)                          ;;*CALL
  (1)                          ;;*      ERROR    N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
  (1)
  (1)  026626                  $ERROR:
  (1)  026626  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
  (3)  026630  004737  026564                  JSR     PC,WHICHV       ;INDICATE BAD UNIT
  (1)  026634  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
  (1)  026640  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
  (1)  026642  013777  001102  152272          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
  (1)  026650  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
  (1)  026654  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
  (1)  026660  162737  000002  001116          SUB     #2,$ERRPC
  (1)  026666  117737  152224  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
  (1)  026674  032777  020000  152236          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
  (1)  026702  001004                          BNE     20$             ;;SKIP TYPEOUTS
  (1)  026704  004737  027016                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
  (1)  026710  104401  001165                  TYPE    ,$CRLF
  (1)  026714                          20$:
  (1)  026714  122737  000001  001210          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1)  026722  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
  (1)  026724  113737  001114  026736          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
  (1)  026732  004737  030134                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
  (1)  026736     000              21$:    .BYTE   0
  (1)  026737     000                      .BYTE   0
  (1)  026740  000777          22$:    BR      22$             ;;APT ERROR LOOP
```

```
    (1)   026742  005777  152172      2$:     TST     @SWR                ;;HALT ON ERROR
    (1)   026746  100002                      BPL     3$                  ;;SKIP IF CONTINUE
    (1)   026750  000000                      HALT                        ;;HALT ON ERROR!
    (1)   026752  104410                      CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
    (1)   026754  032777  001000  152156  3$: BIT     #BIT09,@SWR         ;;LOOP ON ERROR SWITCH SET?
    (1)   026762  001402                      BEQ     4$                  ;;BR IF NO
    (1)   026764  013716  001110              MOV     $LPERR,(SP)         ;;FUDGE RETURN FOR LOOPING
    (1)   026770  005737  001162      4$:     TST     $ESCAPE             ;;CHECK FOR AN ESCAPE ADDRESS
    (1)   026774  001402                      BEQ     5$                  ;;BR IF NONE
    (1)   026776  013716  001162              MOV     $ESCAPE,(SP)        ;;FUDGE RETURN ADDRESS FOR ESCAPE
    (1)   027002                      5$:
    (1)   027002  022737  016500  000042      CMP     #$ENDAD,@#42        ;;ACT-11 AUTO-ACCEPT?
    (1)   027010  001001                      BNE     6$                  ;;BRANCH IF NO
    (1)   027012  000000                      HALT                        ;;YES
    (1)   027014                      6$:
    (1)   027014  000002                      RTI                         ;;RETURN
  2420                                 .SBTTL   ERROR MESSAGE TYPEOUT ROUTINE
    (1)
    (2)                                ;;****************************************************************
    (1)                                ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($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)   027016                      $ERRTYP:
    (1)   027016  104401  001165              TYPE    ,$CRLF              ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   027022  010046                      MOV     R0,-(SP)            ;;SAVE R0
    (1)   027024  005000                      CLR     R0                  ;;PICKUP THE ITEM INDEX
    (1)   027026  153700  001114              BISB    @#$ITEMB,R0
    (1)   027032  001004                      BNE     1$                  ;;IF ITEM NUMBER IS ZERO, JUST
    (1)                                                                   ;;TYPE THE PC OF THE ERROR
    (2)   027034  013746  001116              MOV     $ERRPC,-(SP)        ;;SAVE $ERRPC FOR TYPEOUT
    (2)                                                                   ;;ERROR ADDRESS
    (2)   027040  104402                      TYPOC                       ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)   027042  000445                      BR      10$                 ;;GET OUT
    (1)   027044  005300              1$:     DEC     R0                  ;;ADJUST THE INDEX SO THAT IT WILL
    (1)   027046  006300                      ASL     R0                  ;;        WORK FOR THE ERROR TABLE
    (1)   027050  006300                      ASL     R0
    (1)   027052  006300                      ASL     R0
    (1)   027054  062700  001252              ADD     #$ERRTB,R0          ;;FORM TABLE POINTER
    (1)   027060  012037  027070              MOV     (R0)+,2$            ;;PICKUP 'ERROR MESSAGE'' POINTER
    (1)   027064  001404                      BEQ     3$                  ;;SKIP TYPEOUT IF NO POINTER
    (1)   027066  104401                      TYPE                        ;;TYPE THE 'ERROR MESSAGE''
    (1)   027070  000000              2$:     .WORD   0                   ;;'ERROR MESSAGE'' POINTER GOES HERE
    (1)   027072  104401  001165              TYPE    ,$CRLF              ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   027076  012037  027106      3$:     MOV     (R0)+,4$            ;;PICKUP 'DATA HEADER'' POINTER
    (1)   027102  001404                      BEQ     5$                  ;;SKIP TYPEOUT IF 0
    (1)   027104  104401                      TYPE                        ;;TYPE THE 'DATA HEADER''
    (1)   027106  000000              4$:     .WORD   0                   ;;'DATA HEADER'' POINTER GOES HERE
    (1)   027110  104401  001165              TYPE    ,$CRLF              ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   027114  010146              5$:     MOV     R1,-(SP)            ;;SAVE R1
    (1)   027116  012001                      MOV     (R0)+,R1            ;;PICKUP 'DATA TABLE'' POINTER
    (1)   027120  001415                      BEQ     9$                  ;;BR IF NO DATA TO BE TYPED
    (1)   027122  012000                      MOV     (R0)+,R0            ;;PICKUP 'DATA FORMAT'' POINTER
    (1)   027124  105720              6$:     TSTB    (R0)+               ;;''OCTAL'' OR ''DECIMAL''
```

```
(1)   027126  001003                    BNE    7$              ;;BR IF DECIMAL
(2)   027130  013146                    MOV    @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)   027132  104402                    TYPOC                  ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)   027134  000402                    BR     8$
(2)   027136                    7$:
(2)   027136  013146                    MOV    @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)   027140  104405                    TYPDS                  ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)   027142  005711            8$:     TST    (R1)            ;;IS THERE ANOTHER NUMBER?
(1)   027144  001403                    BEQ    9$              ;;BR IF NO
(1)   027146  104401  027166            TYPE   ,11$            ;;TYPE TWO(2) SPACES
(1)   027152  000764                    BR     6$              ;;LOOP
(1)
(1)   027154  012601            9$:     MOV    (SP)+,R1        ;;RESTORE R1
(1)   027156  012600            10$:    MOV    (SP)+,R0        ;;RESTORE R0
(1)   027160  104401  001165            TYPE   ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
(1)   027164  000207                    RTS    PC              ;;RETURN
(1)   027166  020040    000     11$:    .ASCIZ / /            ;;TWO(2) SPACES
(1)           027172                    .EVEN
2421                            .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                             ;;****************************************************************
(1)                             ;POWER DOWN ROUTINE
(1)   027172  012737  027336  000024  $PWRDN: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)   027200  012737  000340  000026          MOV    #340,@#PWRVEC+2 ;;PRIO:7
(3)   027206  010046                    MOV    R0,-(SP)        ;;PUSH R0 ON STACK
(3)   027210  010146                    MOV    R1,-(SP)        ;;PUSH R1 ON STACK
(3)   027212  010246                    MOV    R2,-(SP)        ;;PUSH R2 ON STACK
(3)   027214  010346                    MOV    R3,-(SP)        ;;PUSH R3 ON STACK
(3)   027216  010446                    MOV    R4,-(SP)        ;;PUSH R4 ON STACK
(3)   027220  010546                    MOV    R5,-(SP)        ;;PUSH R5 ON STACK
(3)   027222  017746  151712            MOV    @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)   027226  010637  027342            MOV    SP,$SAVR6       ;;SAVE SP
(1)   027232  012737  027244  000024    MOV    #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)   027240  000000                    HALT
(1)   027242  000776                    BR     .-2             ;;HANG UP
(1)
(2)                             ;;****************************************************************
(1)                             ;POWER UP ROUTINE
(1)   027244  012737  027336  000024  $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)   027252  013706  027342            MOV    $SAVR6,SP       ;;GET SP
(1)   027256  005037  027342            CLR    $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)   027262  005237  027342    1$:     INC    $SAVR6          ;;WAIT FOR THE INC
(1)   027266  001375                    BNE    1$              ;;OF  WORD
(3)   027270  012677  151644            MOV    (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)   027274  012605                    MOV    (SP)+,R5        ;;POP STACK INTO R5
(3)   027276  012604                    MOV    (SP)+,R4        ;;POP STACK INTO R4
(3)   027300  012603                    MOV    (SP)+,R3        ;;POP STACK INTO R3
(3)   027302  012602                    MOV    (SP)+,R2        ;;POP STACK INTO R2
(3)   027304  012601                    MOV    (SP)+,R1        ;;POP STACK INTO R1
(3)   027306  012600                    MOV    (SP)+,R0        ;;POP STACK INTO R0
(1)   027310  012737  027172  000024    MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)   027316  012737  000340  000026    MOV    #340,@#PWRVEC+2 ;;PRIO:7
(1)   027324  104401                    TYPE                   ;;REPORT THE POWER FAILURE
(1)   027326  027344            $PWRMG: .WORD  PWRMSG          ;;POWER FAIL MESSAGE POINTER
```

```
     (1)   027330  012716                    MOV      (PC)+,(SP)       ;;RESTART AT BEGIN
     (1)   027332  001556          $PWRAD:  .WORD    BEGIN            ;;RESTART ADDRESS
     (1)   027334  000002                    RTI
     (1)   027336  000000          $ILLUP:  HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
     (1)   027340  000776                    BR       .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
     (1)   027342  000000          $SAVR6:  0                         ;;PUT THE SP HERE
    2422   027344  051200  051505  040524  PWRMSG:  .ASCIZ   <200>/RESTARTING AFTER A POWER FAILURE  /
           027352  052122  047111  020107
           027360  043101  042524  020122
           027366  020101  047520  042527
           027374  020122  040506  046111
           027402  051125  020105  000040
```

```
 2424                                       .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)  027410  105737  001157    $TYPE:  TSTB     $TPFLG          ;;IS THERE A TERMINAL?
 (1)  027414  100002                    BPL      1$              ;;BR IF YES
 (1)  027416  000000                    HALT                     ;;HALT HERE IF NO TERMINAL
 (1)  027420  000430                    BR       3$              ;;LEAVE
 (1)  027422  010046            1$:     MOV      R0,-(SP)        ;;SAVE R0
 (1)  027424  017600  000002            MOV      @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 (1)  027430  122737  000001  001210    CMPB     #APTENV,$ENV    ;;RUNNING IN APT MODE
 (1)  027436  001011                    BNE      62$             ;;NO,GO CHECK FOR APT CONSOLE
 (1)  027440  132737  000100  001211    BITB     #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 (1)  027446  001405                    BEQ      62$             ;;NO,GO CHECK FOR CONSOLE
 (1)  027450  010037  027460            MOV      R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 (1)  027454  004737  030124            JSR      PC,$ATY3        ;;SPOOL MESSAGE TO APT
 (1)  027460  000000            61$:    .WORD    0               ;;MESSAGE ADDRESS
 (1)  027462  132737  000040  001211  62$:  BITB   #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
 (1)  027470  001003                    BNE      60$             ;;YES,SKIP TYPE OUT
 (1)  027472  112046            2$:     MOVB     (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 (1)  027474  001005                    BNE      4$              ;;BR IF IT ISN'T THE TERMINATOR
 (1)  027476  005726                    TST      (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 (1)  027500  012600            60$:    MOV      (SP)+,R0        ;;RESTORE R0
 (1)  027502  062716  000002    3$:     ADD      #2,(SP)         ;;ADJUST RETURN PC
 (1)  027506  000002                    RTI                      ;;RETURN
 (1)  027510  122716  000011    4$:     CMPB     #HT,(SP)        ;;BRANCH IF <HT>
 (1)  027514  001430                    BEQ      8$
 (1)  027516  122716  000200            CMPB     #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 (1)  027522  001006                    BNE      5$
 (1)  027524  005726                    TST      (SP)+           ;;POP  <CR><LF> EQUIV
 (1)  027526  104401                    TYPE                     ;;TYPE A CR AND LF
 (1)  027530  001165                    $CRLF
 (1)  027532  105037  027666            CLRB     $CHARCNT        ;;CLEAR CHARACTER COUNT
 (1)  027536  000755                    BR       2$              ;;GET NEXT CHARACTER
 (1)  027540  004737  027622    5$:     JSR      PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 (1)  027544  123726  001156    6$:     CMPB     $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 (1)  027550  001350                    BNE      2$              ;;IF NO GO GET NEXT CHAR.
 (1)  027552  013746  001154            MOV      $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 (1)                                                             ;;AND THE NULL CHAR.
 (1)  027556  105366  000001    7$:     DECB     1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 (1)  027562  002770                    BLT      6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
 (1)   027564  004737  027622              JSR     PC,$TYPEC       ;;GO TYPE A NULL
 (1)   027570  105337  027666              DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 (1)   027574  000770                      BR      7$              ;;LOOP
 (1)
 (1)                                  ;HORIZONTAL TAB PROCESSOR
 (1)
 (1)   027576  112716  000040     8$:      MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 (1)   027602  004737  027622     9$:      JSR     PC,$TYPEC       ;;TYPE A SPACE
 (1)   027606  132737  000007  027666      BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 (1)   027614  001372                      BNE     9$              ;;TAB STOP
 (1)   027616  005726                      TST     (SP)+           ;;POP SPACE OFF STACK
 (1)   027620  000724                      BR      2$              ;;GET NEXT CHARACTER
 (1)   027622  105777  151322     $TYPEC:  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 (1)   027626  100375                      BPL     $TYPEC
 (1)   027630  116677  000002  151314      MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 (1)   027636  122766  000015  000002      CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 (1)   027644  001003                      BNE     1$              ;;BRANCH IF NO
 (1)   027646  105037  027666              CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 (1)   027652  000406                      BR      $TYPEX          ;;EXIT
 (1)   027654  122766  000012  000002 1$:  CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 (1)   027662  001402                      BEQ     $TYPEX          ;;BRANCH IF YES
 (1)   027664  105227                      INCB    (PC)+           ;;COUNT THE CHARACTER
 (1)   027666  000000            $CHARCNT:.WORD    0               ;;CHARACTER COUNT STORAGE
 (1)   027670  000207            $TYPEX:   RTS     PC
 (1)
2425                                 .SBTTL   CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 (1)
 (2)                                 ;;******************************************************************
 (1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 (1)                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 (1)                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 (1)                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 (1)                                 ;*REPLACED WITH SPACES.
 (1)                                 ;*CALL:
 (1)                                 ;*       MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 (1)                                 ;*       TYPDS                   ;;GO TO THE ROUTINE
 (1)
 (2)   027672                  $TYPDS:
 (3)   027672  010046                       MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 (3)   027674  010146                       MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (3)   027676  010246                       MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 (3)   027700  010346                       MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 (3)   027702  010546                       MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 (1)   027704  012746  020200               MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 (1)   027710  016605  000020               MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 (1)   027714  100004                       BPL     1$              ;;BR IF INPUT IS POS.
 (1)   027716  005405                       NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 (1)   027720  112766  000055  000001       MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 (1)   027726  005000            1$:        CLR     R0              ;;ZERO THE CONSTANTS INDEX
 (1)   027730  012703  030106               MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 (1)   027734  112723  000040               MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 (1)   027740  005002            2$:        CLR     R2              ;;CLEAR THE BCD NUMBER
 (1)   027742  016001  030076               MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 (1)   027746  160105            3$:        SUB     R1,R5           ;;FORM THIS BCD DIGIT
```

```
 (1)   027750  002402                       BLT     4$          ;;BR IF DONE
 (1)   027752  005202                       INC     R2          ;;INCREASE THE BCD DIGIT BY 1
 (1)   027754  000774                       BR      3$
 (1)   027756  060105              4$:      ADD     R1,R5       ;;ADD BACK THE CONSTANT
 (1)   027760  005702                       TST     R2          ;;CHECK IF BCD DIGIT=0
 (1)   027762  001002                       BNE     5$          ;;FALL THROUGH IF 0
 (1)   027764  105716                       TSTB    (SP)        ;;STILL DOING LEADING 0'S?
 (1)   027766  100407                       BMI     7$          ;;BR IF YES
 (1)   027770  106316              5$:      ASLB    (SP)        ;;MSD?
 (1)   027772  103003                       BCC     6$          ;;BR IF NO
 (1)   027774  116663  000001  177777       MOVB    1(SP),-1(R3) ;;YES--SET THE SIGN
 (1)   030002  052702  000060      6$:      BIS     #'0,R2      ;;MAKE THE BCD DIGIT ASCII
 (1)   030006  052702  000040      7$:      BIS     #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 (1)   030012  110223                       MOVB    R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 (1)   030014  005720                       TST     (R0)+       ;;JUST INCREMENTING
 (1)   030016  020027  000010               CMP     R0,#10      ;;CHECK THE TABLE INDEX
 (1)   030022  002746                       BLT     2$          ;;GO DO THE NEXT DIGIT
 (1)   030024  003002                       BGT     8$          ;;GO TO EXIT
 (1)   030026  010502                       MOV     R5,R2       ;;GET THE LSD
 (1)   030030  000764                       BR      6$          ;;GO CHANGE TO ASCII
 (1)   030032  105726              8$:      TSTB    (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?
 (1)   030034  100003                       BPL     9$          ;;BR IF NO
 (1)   030036  116663  177777  177776       MOVB    -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
 (1)   030044  105013              9$:      CLRB    (R3)        ;;SET THE TERMINATOR
 (3)   030046  012605                       MOV     (SP)+,R5    ;;POP STACK INTO R5
 (3)   030050  012603                       MOV     (SP)+,R3    ;;POP STACK INTO R3
 (3)   030052  012602                       MOV     (SP)+,R2    ;;POP STACK INTO R2
 (3)   030054  012601                       MOV     (SP)+,R1    ;;POP STACK INTO R1
 (3)   030056  012600                       MOV     (SP)+,R0    ;;POP STACK INTO R0
 (1)   030060  104401  030106               TYPE    ,$DBLK      ;;NOW TYPE THE NUMBER
 (1)   030064  016666  000002  000004       MOV     2(SP),4(SP) ;;ADJUST THE STACK
 (1)   030072  012616                       MOV     (SP)+,(SP)
 (1)   030074  000002                       RTI                 ;;RETURN TO USER
 (1)   030076  023420              $DTBL:   10000.
 (1)   030100  001750                       1000.
 (1)   030102  000144                       100.
 (1)   030104  000012                       10.
 (1)   030106  000004              $DBLK:   .BLKW   4
2426                                .SBTTL  APT COMMUNICATIONS ROUTINE
 (1)
 (2)                                ;;***********************************************************
 (1)   030116  112737  000001  030362  $ATY1:  MOVB    #1,$FFLG    ;;TO REPORT FATAL ERROR
 (1)   030124  112737  000001  030360  $ATY3:  MOVB    #1,$MFLG    ;;TO TYPE A MESSAGE
 (1)   030132  000403                       BR      $ATYC
 (1)   030134  112737  000001  030362  $ATY4:  MOVB    #1,$FFLG    ;;TO ONLY REPORT FATAL ERROR
 (2)   030142                      $ATYC:
 (3)   030142  010046                       MOV     R0,-(SP)    ;;PUSH R0 ON STACK
 (3)   030144  010146                       MOV     R1,-(SP)    ;;PUSH R1 ON STACK
 (1)   030146  105737  030360               TSTB    $MFLG       ;;SHOULD TYPE A MESSAGE?
 (1)   030152  001450                       BEQ     5$          ;;IF NOT:  BR
 (1)   030154  122737  000001  001210       CMPB    #APTENV,$ENV  ;;OPERATING UNDER APT?
 (1)   030162  001031                       BNE     3$          ;;IF NOT:  BR
 (1)   030164  132737  000100  001211       BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 (1)   030172  001425                       BEQ     3$          ;;IF NOT:  BR
```

```
(1)  030174  017600  000004              MOV    a4(SP),R0       ;:GET MESSAGE ADDR.
(1)  030200  062766  000002  000004      ADD    #2,4(SP)              ;:BUMP RETURN ADDR.
(1)  030206  005737  001170      1$:     TST    $MSGTYPE        ;:SEE IF DONE W/ LAST XMISSION?
(1)  030212  001375                      BNE    1$              ;:IF NOT:  WAIT
(1)  030214  010037  001204              MOV    R0,$MSGAD       ;:PUT ADDR IN MAILBOX
(1)  030220  105720              2$:     TSTB   (R0)+           ;:FIND END OF MESSAGE
(1)  030222  001376                      BNE    2$
(1)  030224  163700  001204              SUB    $MSGAD,R0       ;:SUB START OF MESSAGE
(1)  030230  006200                      ASR    R0              ;:GET MESSAGE LNGTH IN WORDS
(1)  030232  010037  001206              MOV    R0,$MSGLGT      ;:PUT LENGTH IN MAILBOX
(1)  030236  012737  000004  001170      MOV    #4,$MSGTYPE     ;:TELL APT TO TAKE MSG.
(1)  030244  000413                      BR     5$
(1)  030246  017637  000004  030272 3$:  MOV    a4(SP),4$       ;:PUT MSG ADDR IN JSR LINKAGE
(1)  030254  062766  000002  000004      ADD    #2,4(SP)              ;:BUMP RETURN ADDRESS
(3)  030262  013746  177776              MOV    177776,-(SP)    ;:PUSH 177776 ON STACK
(1)  030266  004737  027410              JSR    PC,$TYPE        ;:CALL TYPE MACRO
(1)  030272  000000              4$:     .WORD  0
(1)  030274                      5$:
(1)  030274  105737  030362      10$:    TSTB   $FFLG           ;:SHOULD REPORT FATAL ERROR?
(1)  030300  001416                      BEQ    12$             ;:IF NOT:  BR
(1)  030302  005737  001210              TST    $ENV            ;:RUNNING UNDER APT?
(1)  030306  001413                      BEQ    12$             ;:IF NOT:  BR
(1)  030310  005737  001170      11$:    TST    $MSGTYPE        ;:FINISHED LAST MESSAGE?
(1)  030314  001375                      BNE    11$             ;:IF NOT:  WAIT
(1)  030316  017637  000004  001172      MOV    a4(SP),$FATAL   ;:GET ERROR #
(1)  030324  062766  000002  000004      ADD    #2,4(SP)              ;:BUMP RETURN ADDR.
(1)  030332  005237  001170              INC    $MSGTYPE        ;:TELL APT TO TAKE ERROR
(1)  030336  105037  030362      12$:    CLRB   $FFLG           ;:CLEAR FATAL FLAG
(1)  030342  105037  030361              CLRB   $LFLG           ;:CLEAR LOG FLAG
(1)  030346  105037  030360              CLRB   $MFLG           ;:CLEAR MESSAGE FLAG
(3)  030352  012601                      MOV    (SP)+,R1        ;:POP STACK INTO R1
(3)  030354  012600                      MOV    (SP)+,R0        ;:POP STACK INTO R0
(1)  030356  000207                      RTS    PC              ;:RETURN
(1)  030360     000          $MFLG:      .BYTE  0               ;:MESSG. FLAG
(1)  030361     000          $LFLG:      .BYTE  0               ;:LOG FLAG
(1)  030362     000          $FFLG:      .BYTE  0               ;:FATAL FLAG
(1)          030364                      .EVEN
(1)          000200          APTSIZE-200
(1)          000001          APTENV=001
(1)          000100          APTSPOOL=100
(1)          000040          APTCSUP=040
```

I 8

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 46
CVMNAA.P11   BINARY TO OCTAL (ASCII) AND TYPE   SEQ 0099

```
 2428                                    .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)  030364  017646  000000   $TYPOS: MOV     @(SP),-(SP)      ;;PICKUP THE MODE
  (1)  030370  116637  000001  030607   MOVB    1(SP),$OFILL     ;;LOAD ZERO FILL SWITCH
  (1)  030376  112637  030611           MOVB    (SP)+,$OMODE+1   ;;NUMBER OF DIGITS TO TYPE
  (1)  030402  062716  000002           ADD     #2,(SP)          ;;ADJUST RETURN ADDRESS
  (1)  030406  000406                    BR      $TYPON
  (1)  030410  112737  000001  030607   $TYPOC: MOVB    #1,$OFILL        ;;SET THE ZERO FILL SWITCH
  (1)  030416  112737  000006  030611   MOVB    #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
  (1)  030424  112737  000005  030606   $TYPON: MOVB    #5,$OCNT         ;;SET THE ITERATION COUNT
  (1)  030432  010346                    MOV     R3,-(SP)         ;;SAVE R3
  (1)  030434  010446                    MOV     R4,-(SP)         ;;SAVE R4
  (1)  030436  010546                    MOV     R5,-(SP)         ;;SAVE R5
  (1)  030440  113704  030611            MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
  (1)  030444  005404                    NEG     R4
  (1)  030446  062704  000006            ADD     #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
  (1)  030452  110437  030610            MOVB    R4,$OMODE        ;;SAVE IT FOR USE
  (1)  030456  113704  030607            MOVB    $OFILL,R4        ;;GET THE ZERO FILL SWITCH
  (1)  030462  016605  000012            MOV     12(SP),R5        ;;PICKUP THE INPUT NUMBER
  (1)  030466  005003                    CLR     R3               ;;CLEAR THE OUTPUT WORD
  (1)  030470  006105            1$:     ROL     R5               ;;ROTATE MSB INTO 'C'
  (1)  030472  000404                    BR      3$               ;;GO DO MSB
  (1)  030474  006105            2$:     ROL     R5               ;;FORM THIS DIGIT
  (1)  030476  006105                    ROL     R5
  (1)  030500  006105                    ROL     R5
  (1)  030502  010503                    MOV     R5,R3
  (1)  030504  006103            3$:     ROL     R3               ;;GET LSB OF THIS DIGIT
  (1)  030506  105337  030610            DECB    $OMODE           ;;TYPE THIS DIGIT?
  (1)  030512  100016                    BPL     7$               ;;BR IF NO
  (1)  030514  042703  177770            BIC     #177770,R3       ;;GET RID OF JUNK
  (1)  030520  001002                    BNE     4$               ;;TEST FOR 0
```

J 8

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   MACY11 27(654) 19-SEP-78 08:51 PAGE 46-1
CVMNAA.P11   BINARY TO OCTAL (ASCII) AND TYPE   SEQ 0100

```
  (1)  030522  005704                        TST    R4            ;;SUPPRESS THIS 0?
  (1)  030524  001403                        BEQ    5$            ;;BR IF YES
  (1)  030526  005204              4$:        INC    R4            ;;DON'T SUPPRESS ANYMORE 0'S
  (1)  030530  052703  000060                 BIS    #'0,R3        ;;MAKE THIS DIGIT ASCII
  (1)  030534  052703  000040      5$:        BIS    #' ,R3        ;;MAKE ASCII IF NOT ALREADY
  (1)  030540  110337  030604                 MOVB   R3,8$         ;;SAVE FOR TYPING
  (1)  030544  104401  030604                 TYPE   ,8$          ;;GO TYPE THIS DIGIT
  (1)  030550  105337  030606      7$:        DECB   $OCNT         ;;COUNT BY 1
  (1)  030554  003347                        BGT    2$            ;;BR IF MORE TO DO
  (1)  030556  002402                        BLT    6$            ;;BR IF DONE
  (1)  030560  005204                        INC    R4            ;;INSURE LAST DIGIT ISN'T A BLANK
  (1)  030562  000744                        BR     2$            ;;GO DO THE LAST DIGIT
  (1)  030564  012605              6$:        MOV    (SP)+,R5      ;;RESTORE R5
  (1)  030566  012604                        MOV    (SP)+,R4      ;;RESTORE R4
  (1)  030570  012603                        MOV    (SP)+,R3      ;;RESTORE R3
  (1)  030572  016666  000002 000004          MOV    2(SP),4(SP)   ;;SET THE STACK FOR RETURNING
  (1)  030600  012616                        MOV    (SP)+,(SP)
  (1)  030602  000002                        RTI                  ;;RETURN
  (1)  030604     000              8$:        .BYTE  0             ;;STORAGE FOR ASCII DIGIT
  (1)  030605     000                         .BYTE  0             ;;TERMINATOR FOR TYPE ROUTINE
  (1)  030606     000              $OCNT:     .BYTE  0             ;;OCTAL DIGIT COUNTER
  (1)  030607     000              $OFILL:    .BYTE  0             ;;ZERO FILL SWITCH
  (1)  030610  000000              $OMODE:    .WORD  0             ;;NUMBER OF DIGITS TO TYPE
                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
 2429
  (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)  030612  010146              $TYPBN: MOV    R1,-(SP)      ;;SAVE R1 ON THE STACK
  (1)  030614  016601  000006              MOV    6(SP),R1      ;;GET THE INPUT NUMBER
  (1)  030620  000261                      SEC                  ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
  (1)  030622  112737  000060 030664  1$:  MOVB   #'0,$BIN      ;;SET CHARACTER TO AN ASCII '0'.
  (1)  030630  006101                      ROL    R1            ;;GET THIS BIT
  (1)  030632  001406                      BEQ    2$            ;;DONE?
  (1)  030634  105537  030664              ADCB   $BIN          ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
  (1)  030640  104401  030664              TYPE   ,$BIN         ;;GO TYPE THIS BIT
  (1)  030644  000241                      CLC                  ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
  (1)  030646  000765                      BR     1$            ;;GO DO THE NEXT BIT
  (1)  030650  012601              2$:      MOV    (SP)+,R1      ;;POP THE STACK INTO R1
  (1)  030652  016666  000002 000004        MOV    2(SP),4(SP)   ;;ADJUST THE STACK
  (1)  030660  012616                      MOV    (SP)+,(SP)
  (1)  030662  000002                      RTI                  ;;RETURN TO USER
  (1)  030664     000     000      $BIN:    .BYTE  0,0           ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
```

```
2431                                        .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)   030666 010046           $TRAP:   MOV      R0,-(SP)          ;;SAVE R0
  (1)   030670 016600  000002            MOV      2(SP),R0          ;;GET TRAP ADDRESS
  (1)   030674 005740                    TST      -(R0)             ;;BACKUP BY 2
  (1)   030676 111000                    MOVB     (R0),R0           ;;GET RIGHT BYTE OF TRAP
  (1)   030700 006300                    ASL      R0                ;;POSITION FOR INDEXING
  (1)   030702 016000  030722            MOV      $TRPAD(R0),R0     ;;INDEX TO TABLE
  (1)   030706 000200                    RTS      R0                ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                 ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
  (1)
  (1)   030710 011646           $TRAP2:  MOV      (SP),-(SP)        ;;MOVE THE PC DOWN
  (1)   030712 016666  000004 000002    MOV      4(SP),2(SP)       ;;MOVE THE PSW DOWN
  (1)   030720 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)   030722 030710           $TRPAD:  .WORD    $TRAP2
  (3)   030724 027410                    $TYPE    ;;CALL=TYPE    TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)   030726 030410                    $TYPOC   ;;CALL=TYPOC   TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)   030730 030364                    $TYPOS   ;;CALL=TYPOS   TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)   030732 030424                    $TYPON   ;;CALL=TYPON   TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)   030734 027672                    $TYPDS   ;;CALL=TYPDS   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (3)   030736 030612                    $TYPBN   ;;CALL=TYPBN   TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
  (1)
  (3)   030740 025276                    $GTSWR   ;;CALL=GTSWR   TRAP+7(104407)  GET SOFT-SWR SETTING
  (1)
  (3)   030742 025206                    $CKSWR   ;;CALL=CKSWR   TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
  (3)   030744 025550                    $RDCHR   ;;CALL=RDCHR   TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
  (3)   030746 025640                    $RDLIN   ;;CALL=RDLIN   TRAP+12(104412) TTY TYPEIN STRING ROUTINE
  (3)   030750 026202                    $RDOCT   ;;CALL=RDOCT   TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
2432   030752 005362                    TEST     ;;CALL=CHECK   TRAP+14(104414)
2433   030754 005354                    TESTIT   ;;CALL=CHKIT   TRAP+15(104415)
2434   030756 016052                    DECTYP   ;;CALL=TYPDC   TRAP+16(104416)
2435   030760 010532                    TPRMP    ;;CALL=TESTID  TRAP+17(104417)
2436   030762 013416                    DELAY4   ;;CALL=DELY    TRAP+20(104420)
2437   030764 000310           DIST:    .BLKW    200.             ;STATE-WIDTH DISTRIBUTION
2438   031604 010000           BUFFER:  .BLKW    4096.            ;BUFFER AREA
2439
2440          000001           .END
```

```
ABASE = 171000          23#     59     104     106     108     110     114     115     116
ACDW1 = 000000          59
ACDW2 = 000000          59
ACPUOP= 000000          59
ADBUFF  001356          116#    456     472     483     493     511     522     537     542     562     575     583     602
                        631     774*    778*    816     818     819     821    1145    1171    1310    1359    1377   1446*
                       1454*   1458*   1487*   1553*   1559    1565    1626    1654    1682    1925*   1939    2235
ADDW0 = 000000          59
ADDW1 = 000000          59
ADDW10= 000000          59
ADDW11= 000000          59
ADDW12= 000000          59
ADDW13= 000000          59
ADDW14= 000000          59
ADDW15= 000000          59
ADDW2 = 000000          59
ADDW3 = 000000          59
ADDW4 = 000000          59
ADDW5 = 000000          59
ADDW6 = 000000          59
ADDW7 = 000000          59
ADDW8 = 000000          59
ADDW9 = 000000          59
ADEVCT= 000000          59
ADEVM = 000000          59
ADST1   001354          115#   1445*   1453*   1457*   1555*
AENV  = 000000          59
AENVM = 000000          59
AFATAL= 000000          59
AGAIN   013600         1615#   1701
AGTST   016610         1188*   1193*   1208*   1219*   1238*   2144#
AJGAIN  010246         1064    1083#
AJOFF   010174         1062    1069#
AMADR1= 000000          59
AMADR2= 000000          59
AMADR3= 000000          59
AMADR4= 000000          59
AMAMS1= 000000          59
AMAMS2= 000000          59
AMAMS3= 000000          59
AMAMS4= 000000          59
AMSGAD= 000000          59
AMSGLG= 000000          59
AMSGTY= 000000          59
AMTYP1= 000000          59
AMTYP2= 000000          59
AMTYP3= 000000          59
AMTYP4= 000000          59
APASS = 000000          59
APRIOR= 000000          59
APTCSU= 000040         2424    2426#
APTENV= 000001         2419    2424    2426#
APTSIZ= 000200          191    2426#
APTSPO= 000100         2424    2426#
```

AROUND	014150	1643	1671	1700#								
ASKC	012210	1345	1380	1393#	1408							
ASWREG=	000000	59										
ATESTN=	000000	59										
ATMSG	021056	1481	1485	2289#								
AUNIT =	000000	59										
AUSWR =	000000	59										
AVECT1=	000400	24#	59	105	107	109	111	117	118	119	120	
AVECT2=	000000	59										
BADID	020451	1262	2273#									
BADUNT	001510	161#	337*	2140	2407*							
BASECH	001370	121#	653	706	740	850	948	985	1227*	1232	1246*	1283*	1364*	1385
BASEND	001372	122#	711	731	744	848	924	928	946	1230*	1232*	1250*	1256*	1284*
		1383*	1387									
BASEXC	012306	271	1414#									
BEGIN	001556	52	177#	181	2144	2421						
BEGINA	011040	256	1197#									
BEGINC	010122	259	1054#									
BEGINL	003266	343#	1190	1199								
BEGINM	010376	287	1108#									
BEGINN	011156	284	1223#									
BEGINP	010614	262	1153#									
BEGINW	011116	268	281	1212#								
BEGIN2	001626	54	185	188#								
BEGL	010776	224	265	1187#								
BEG2	001564	179#	1079	1103	1136	1277	2402					
BITPNT	001472	154#	1549*	1552	1572	1573*						
BIT0 =	000001	21#	506	550	555	566	567	568				
BIT00 =	000001	21#										
BIT01 =	000002	21#										
BIT02 =	000004	21#										
BIT03 =	000010	21#										
BIT04 =	000020	21#										
BIT05 =	000040	21#										
BIT06 =	000100	21#										
BIT07 =	000200	21#										
BIT08 =	000400	21#	2406									
BIT09 =	001000	21#	2406	2419								
BIT1 =	000002	21#	393	1248	1337	1354	1372					
BIT10 =	002000	21#	1899	1996								
BIT11 =	004000	21#	588	1989	2406							
BIT12 =	010000	21#	1115	1138								
BIT13 =	020000	21#	1115	1160	1172	2419						
BIT14 =	040000	21#	398	401	533	535	1325	2406				
BIT15 =	100000	21#	188	327	429	442	453	533	535	554	565	628	1325	1429
		2113										
BIT2 =	000004	21#	419									
BIT3 =	000010	21#	424	579	1112	1335	1352	1370				
BIT4 =	000020	21#	415	608								
BIT5 =	000040	21#	411	593								
BIT6 =	000100	21#	243	407	463	499	506	512	526	551	1166	2131	
BIT7 =	000200	21#	450	512	554	593	608	727				
BIT8 =	000400	21#	390	587	613	614						
BIT9 =	001000	21#	448	556	595	612						

```
BPTVEC= 000014        21#
BUFFER  031604      1593    1631*   1633*   1659*   1661*   1687*   1689*   1707    1735    1811    1823    1825    1847
                    1859    1860    1861    1883    1886    2438#
BUFF1   023027      1794    2333#
BUFF2   023055      1882    2343#
BUMPAD  012410      1191    1200    1206    1217    1433#
BYPASS  012562      1448    1457#
B10     016002      2022#   2023
B11     016036      2032#   2033
CCHAN   021143      1054    1108    2295#
CH      021011      1162    2285#
CHAN    021113      1918    2292#
CHANL   001444       143#    478*    488*    706*    708*    711     718*    749*    750     757*    775*    791*    847*
                     850*    856     885     910*    913     915     917*    920     924     930     961*   1056*   1057
                    1560    1919    1924*   1926    1964*   2396
CHECK = 104414       437     445     454     464     473     559     600     629    2432#
CHKIT = 104415       402     408     412     416     420     425     430    2433#
CH1     001432       138#    951*    955*    975     977     978*   1484    1486    1507
CH2     001434       139#    953*    956*    961     975     978     979*   1480    1502
CKSWR = 104410      2406    2419    2431#
CLEAR1  013474      1595#   1597
CLEAR2  013540      1606#   1608
CLKBPR  001406       128#    596*
CLKCSR  001404       127#    597*
COMPAR  015502       360     377     657     664     672     682     690     698     714     753     781     796     825
                     830    1045    1949#
CONVCD  015366       776     779    1926#
CONVRT  015352       655     662     670     680     688     696    1488    1923#
CONVR1  013624      1621    1623#
CONVR2  013740      1649    1651#
CONVR3  014054      1677    1679#
CONVTC  015360       713     752     857     886     962    1925#
CONV15  015664       358     375    1989#
CR    = 000015        21#   2424
CRLF  = 000200        21#    203    2424
CRWR    021463       625     734     767    1084    1139    1295    2301#
C2      022765      1803    1888    2327#
C3      022773      1885    2328#
DAC     001464       151#    862     866     891     895     997    1001    1010    1012    1093    1097    1506    1511
                    1550*   1552*   1553    1572*
DASH    020765      1738    2283#
DDISP = 177570        21#     59     191
DECPNT  024545      1745*   1778*   2390#
DECTYP  016052      2039#   2434
DELAY   001466       152#   1036*   1040*   1613*   1700*
DELAY1  013622       208*    213*   1622#
DELAY2  013736       209*    214*   1650#
DELAY3  014052       210*    215*   1678#
DELAY4  013416      1579#   1580    2436
DELCLR  015254      1792    1804    1889    1899#
DELY  = 104420       213     214     215    2436#
DF1     024642        76      83      89      95      98     101    2400#
DH1     024312        74    2383#
DH2     024356        93    2384#
```

DH3	024442	81	87	2385#								
DH5	024476	98	2386#									
DH6	024522	101	2387#									
DIFLIN	013424	989	1584#									
DIFM	020235	733	2268#									
DISPLA	001142	59#	191*	1174*	2406*	2419*						
DISPRE	000174	49#	191									
DIST	030764	1598	1718*	1795	2437#							
DOT	023643	250	2375#									
DRVCSR	001410	129#										
DRVDIR	001414	131#	1252	1258								
DRVDOR	001412	130#	613*	614*	727*							
DSWR =	177570	21#	59	191								
DT1	024552	75	2395#									
DT2	024566	94	2396#									
DT3	024606	82	88	2397#								
DT5	024620	98	2398#									
DT6	024632	101	2399#									
DUMMY	001442	142#	792*	856*	885*	1057*	1502*	1507*	1555			
DWRF	017267	233	2246#									
ECHAN	017126	1228	2242#									
EDGE	001470	153#	858*	887*	963*	994*	998*	1090*	1094*	1516	1565	
EDGFLG	013164	86/*	896*	973*	1518	1524#						
EMTVEC=	000030	21#	191*									
EM1	023733	73	2377#									
EM2	023771	80	2378#									
EM3	024031	86	2379#									
EM4	024072	92	2380#									
EM5	024133	98	2381#									
EM6	024214	101	2382#									
ENCOM	017517	1416	1424	2255#								
ERMSG	021606	366	383	801	877	906	1080	1104	1495	1751	1768	1782	1840	2305#
ERR	013020	1492	1495#									
ERRADR	007770	851*	880*	1017#	1501*							
ERRTOT	022721	2134	2325#									
ERRVEC=	000004	21#	191*	293*	312*	313*	1450*	2406*				
EVER	001506	160#	249*	314	326*	327*	329	331	334			
EXTMSG	016534	2130	2131#									
EXTST	020330	622	2269#									
FIRST	001422	134#	1604*	1730	1732*							
FIXADR	012504	1434	1449#									
FIXONE	012510	217	336	1223	1450#							
FLAG	001460	149#	280*	1790	1845							
FOUND1	022074	318	2312#									
FOUND2	022117	321	2313#									
GADBUF	001400	125#	1976									
GAIN5	020673	1119	2279#									
GAIN5M	020734	1125	2281#									
GAIN50	020713	1122	2280#									
GCONVT	015550	355	372	1963#								
GETDAT	015032	1850#	1858									
GHLF	020656	1116	2278#									
GNS =	****** U	203	2431	2432	2433	2434	2435	2436				
GSTREG	001376	124#	1967*	1974*								

```
GTSWR = 104407         203    276    2431#
GVECT   001402         126#   1971*  1972*
HAFMSG  021776         1779   2310#
HALF    014476         1771   1773#
HEAD5   022642         1154   2323#
HIDLY1  013704         1629   1638   1639#
HIDLY2  014020         1657   1666   1667#
HIDLY3  014134         1685   1694   1695#
HT    = 000011          21#   2424
HUNS    024544         2048*  2061*  2062   2064*  2068*  2075   2389#
IGND    021424         1069   2300#
INITVT  023075          202   2349#
INRNGE  014214         1714   1717#
IOTRD   016612           47    312   1450   2166#
IOTVEC= 000020          21#    191*
IVOLT   021523         1083   2302#
KBVECT  001374          123#
LAST    014330         1728   1741#
LEND    007666          986    988    990#
LESS    015770         2017   2019#
LF    = 000012          21#   2424
LINEA   022621         1833   2322#
LOAD    015232         1884   1887   1891#
LOADY   015750         1798   1800   1894   2013#
LOAD0   015236         1892#  1896
LODLY1  013676         1627   1637#
LODLY2  014012         1655   1665#
LODLY3  014126         1683   1693#
LO2     015230         1846   1890#
LSB     021021         1479   2287#
LSBMSG  020756         1740   2282#
MADR    017425         1414   2253#
MASKNM  001512          162#  1435*  1437   1451*  2407   2408
MAX     001500          157#  1866*  1871   1873*  1877
MAXTST  015130         1869   1871#
MDASH   017347         1312   1386   1395   2249#
MDIF    017373         1327   2251#
MEND    022045         1202   2311#
MESGD   022750         2139   2326#
MIN     001474          155#  1865*  1868   1870*  1876
MINUS   017343         1521   1992   2041   2247#
MLSB    023010         1025   2330#
MLSBAT  023016         1916   2331#
MNCAD0  001332          104#  1442   1444
MOFSET  022775          794   2329#
MPRMP   017413         1329   2252#
MSE     017353         1323   2250#
MSG16   022247         1793   2317#
MSG18   022143         1881   2314#
MSG20   022203         1584   2316#
MSG21   022574         1810   2321#
MTEST   002470          231#   273
MTEST0  002532          228    239    241#
MTEST1  002536          226    242#   277    289    1430
```

```
MVCT    017461      1421    2254#
MOLSB   021336      1088    2298#
NAR     014404      1754    1756#
NARMSG  021645      1758    2307#
NARROW  001420       133#   1603*   1722*   1756    1773
NBEXT   001436       140#   1433    1447*   1456*
NEXT1   013604      1616#   1699
NMBEXT  001440       141#    334*    335*   1224*   1456    2137
NOIJMP  007366       838     934#
NOIMSG  017154       839    2243#
NOITST  006612       837     839#   1237
NOTNAR  014250      1721    1727#
NOTNEW  014714      1820    1825#
NOTOK1  013670      1635#   1641
NOTOK2  014004      1663#   1669
NOTOK3  014120      1691#   1697
NXTCMP  015114      1867#   1875
NXTCVT  013304      1555#   1569
NXTSTA  014656      1812#   1826
NXTY1   014612      1797#   1802
NXT8    015076      1863#   1880
OFFERR  006416       799     801#
OFFOK   006434       800     805#
OFFSET  007670       793     994#   1074
OFSET   017237       786    2245#
OKAYD   016076      2044    2046#
OKAY1   013714      1632    1634    1636    1642#
OKAY2   014030      1660    1662    1664    1670#
OKAY3   014144      1688    1690    1692    1698#
OKMSG   021132       364     381     805     873     902    1493    1755    1772    1786    1844    2294#
ONES    024547      2046*   2053*   2054    2056*   2070*   2392#
OUT     001504       159#   1601*   1715*   1760    1763    1766    1775
OUTMSG  021743      1765    2309#
PC     -%000007       21#    197*    217*    221*    336*    341*    358*    367*    375*    384*    503*    530*    623*
                     637*    787*    793*    795*    802*    837*    840*    861*    865*    870*    878*    890*    894*
                     899*    907*    933*    938*    965*    967*    974*    989*    990*   1003*   1014*   1016    1026*
                    1051*   1074*   1075*   1076*   1099*   1100*   1187*   1189*   1190*   1191*   1197*   1198*   1199*
                    1200*   1203*   1205*   1206*   1212*   1213*   1214*   1216*   1217*   1223*   1237*   1273*   1299*
                    1345*   1380*   1389*   1410*   1475*   1476*   1482*   1494*   1496*   1498*   1505*   1510*   1513*
                    1523*   1536*   1585*   1752*   1769*   178?    1792*   1798*   1800*   1804*   1841*   1884*   1887*
                    1889*   1890*   1894*   1897*   1910*   191/    1920*   2008*   2035*   2089*   2130*   2143    2214*
                    2402*   2415*   2419*   2420*   2421    2424*   2426*
PEAK    001456       148#    891*    895*    898     900     904*
PERCNT  001502       158#   1544*   1545*   1546*   1547*   1548*   1570
PIRQ  = 177772        21#
PIRQVE= 000240        21#
PKNOI   021077       897    2291#
PLUS    014676      1817    1819#
PLUSR2  015756      2014    2016#
POS     016064      2040    2043#
POSITV  021604      1023    2304#
POSNOI  015322      1913    1915#
PRIME1  023127       241    2365#
PRO   = 000000        21#
```

```
PR1   = 000040      21#
PR2   = 000100      21#
PR3   = 000140      21#
PR4   = 000200      21#
PR5   = 000240      21#
PR6   = 000300      21#
PR7   = 000340      21#
PS    = 177776      21#
PSW   = 177776      21#
PWRMSG  027344      2421    2422#
PWRVEC= 000024      21#     191*    2421*
QUEST   017345      288     1399    2248#
RBEG    001634      178     189#
RBEG1   001640      187     190#
RBEG2   001620      53      183     186#
RDCHR = 104411      2402    2431#
RDLIN = 104412      234     251     626    735    768    1059   1070   1085   1140   1264   1296   1400   2404
                    2431#
RDOCT = 104413      1055    1109    1226   1229   1417   1425   2431#
READ    014166      1708#   1742
RELACC  014642      1791    1808#
RESVEC= 000010      21#     204     205*   216*
RET     013162      1519    1523#
RETERR  005404      645     647#
RETURN  001540      170#    1164    1556   1934   1971
RET1    001550      172#    1614
RET2    001554      171     173#
RMS     001454      147#    862*    866*   869    871    875*
RMSNOI  021063      868     2290#
RNA     001446      144#    218*    1590
RNB     001450      145#    219*    1591
RNC     001452      146#    220*    1592
R0     %000000      21#     179*    180*   181*   182*   183*   184*   185*   206*   207*   235*   237*   238
                    252*    253*    254    257    260    263    266    269    272    274    278    282    28.
                    448*    451*    456*   472*   556*   557*   562*   575*   595*   598*   612*   615*   740*
                    741     743*    745    747    749    945*   948*   950*   951    952*   953    1060*  1061
                    1063    1110*   1111*  1112*  1113   1158*  1159*  1163   1165*  1166*  1167   1171*  1174
                    1177    1252*   1253*  1254   1258*  1259*  1260   1269*  1270*  1271   1274   1305*  1310*
                    1314*   1321    1325   1333*  1334*  1335*  1336   1348*  1349   1351*  1352*  1353   1359*
                    1360*   1366*   1367   1369*  1370*  1371   1377*  1378*  1396*  1397*  1398   1401*  1402*
                    1403    1405    1437*  1438*  1467*  1469   1471*  1472   1473   1526*  1528   1530*  1531
                    1532    1551*   1567*  1570   1579*  1593*  1595*  1598*  1606*  1609   1610*  1611*  1612
                    1619*   1620*   1622*  1626*  1628   1630*  1631*  1633*  1637   1647*  1648*  1650*  1654*
                    1656    1658*   1659*  1661*  1665   1675*  1676*  1678*  1682*  1684   1686*  1687*  1689*
                    1693    1706*   1741*  1776*  1780   1795*  1797   1811*  1812   1815*  1822   1825   1847*
                    1850    1856*   1859*  1864   1867   1883*  1886*  1892   1893   1903*  1905*  1923*  1924
                    1926*   1927*   1928*  1929   1930*  1931*  1936*  1940*  1953*  1954*  1956*  1957   1963*
                    1964    1965*   1967   1968*  1969*  1973*  1977*  1996*  1997   2001*  2130*  2404*  2420*
                    2421*   2424*   2425*  2426*  2431*
R1    -%000001      21#     1178*   1179*  1304*  1311   1313*  1316*  1317   1319*  1320   1330*  1331   1333
                    1344*   1347*   1348   1362*  1364   1365*  1366   1382*  1383   1394   1396   1422*  1423
                    1436*   1440*   1442   1443*  1444   1468*  1469*  1470*  1471   1527*  1528   1529*  1530
                    1554*   1568*   1594*  1596*  1599*  1607*  1615*  1639*  1642*  1667*  1670*  1695*  1698*
                    1707*   1708    1734   1796*  1801*  1808*  1814*  1815   1816   1848*  1857*  1862*  1879*
```

```
                              1891*  1895*  1904*  1907*  1952*  1954   1995*  1999   2000*  2080*  2086*  2087   2404*
                              2420*  2421*  2425*  2426*  2429*
R2      -%000002              21#    812*   822*   869*   898*   964*   969    971*   972    977*   979    981*   1021*
                              1168*  1181*  1506*  1511*  1590*  1616*  1617*  1618*  1619   1644   1672   1708*  1709*
                              1710*  1711*  1712*  1713   1717*  1718*  1719*  1720   1723   1727   1746*  1756*  1759*
                              1760*  1763*  1773*  1774*  1775*  1776   1797*  1799*  1812*  1813*  1814   1831*  1850*
                              1851*  1852*  1853*  1854*  1855*  1856   1863*  1874*  1892*  1912   1914*  1994*  1999*
                              2003*  2004*  2005*  2013   2015*  2016   2018*  2019   2020*  2021*  2024   2039   2042*
                              2043   2045*  2050   2052*  2083*  2085*  2086   2404*  2421*  2425*
R3      -%000003              21#    357*   374*   809*   819    821*   824    1030*  1032*  1073*  1089*  1516*  1517
                              1520*  1522   1600*  1640*  1668*  1696*  1734*  1735*  1736*  1737*  1809*  1819   1821*
                              1827*  1828*  1829*  1830*  1831   1838   1864*  1865   1866   1867*  1868   1870   1871
                              1873   1989   1991*  1993*  1997   2019   2025*  2026*  2027*  2028*  2029*  2030*  2031*
                              2034   2402*  2421*  2425*  2428*
R4      -%000004              21#    777*   780    810*   816    818*   829    972*   1491   1559*  1560*  1561*  1562*
                              1563   1591*  1616   1644*  1645*  1646*  1647   1673   1816*  1818*  1819   1821   1860*
                              1876*  1878   2421*  2428*
R5      %000005               21#    355*   360*   372*   377*   655*   657*   662*   664*   670*   672*   680*   682*
                              688*   690*   696*   698*   709*   710    713*   714*   719    744*   745*   746*   747*
                              750    752*   753*   776*   779*   781*   796*   825*   830*   852*   853*   854    857*
                              859*   863*   881*   882*   883    886*   888*   892*   928*   929*   930    957*   958*
                              959    962*   995*   999*   1045*  1091*  1095*  1488*  1503*  1508*  1544   1575*  1592*
                              1617   1645   1672*  1673*  1674*  1675   1822*  1823*  1824*  1834   1836*  1837   1861*
                              1877*  1878   1923   1946*  1949   1950   1959   1960*  1963   1983*  2421*  2425*  2428*
R6      -%000006              21#    191*
R7      =%000007              21#
SADTST  020044               1294   2265#
SARSUB  013224               859    863    888    892    995    999    1091   1095   1503   1508   1544#
SCHAN   017076               1225   2241#
SCM     020551               1114   2275#
SDDIF   020171               1291   2267#
SDSE    020125               766    1293   2266#
SEL     021231               1058   2296#
SETCH   021034               1483   2288#
SETINT  013166               503    530    1526#
SETMSG  017204               937    2244#
SET1A   013036               965    967    1501#
SHIFT   013406               1571   1573#
SKIPST  001424               135#   1605*  1725*  1746   1749
SKPMSG  021623               1748   2306#
SLASH   021130               1835   2293#
SP      =%000006             21#    170    172*   191*   204*   212    216    235    244*   245*   252    304    310
                              319*   324    338*   339*   351*   352*   500*   501*   515    518*   519*   527*   528*
                              540    545*   546*   624*   627    646*   736    769    788*   841*   939*   1015   1049*
                              1050*  1056   1060   1071   1086   1110   1141   1149*  1155*  1156*  1163*  1177*  1227
                              1230   1266*  1267*  1269   1297   1300*  1311*  1320*  1385*  1387*  1394*  1398*  1401
                              1409*  1415*  1418   1420   1423*  1426   1428   1449*  1480*  1484*  1490*  1512*  1517*
                              1522*  1586*  1737*  1834*  1837*  1919*  2130*  2135*  2140*  2166   2173   2209   2210
                              2215*  2218*  2220*  2402*  2404*  2406*  2419*  2420*  2421*  2424*  2425*  2426*  2428*
                              2429*  2431*
SPACE   021014               359    376    1035   1176   2286#
SPREAD  001462               150#   479*   489*   1950*  1957   2396
SRM     020577               1130   2276#
STACK = 001100               21#    191
```

```
STATE    020771        1733   2284#
STD      006240         679    771#
STKLMT=  177774          21#
STREG    001352         114#   391*   392    434*   442*   449*   453*   459*   460    469*   470    480*   481
                        490*   491    506*   507    509    510*   521*   533*   534    536*   541*   548*   550*
                        552    555*   566*   567*   568*   569    576*   579*   580*   581    587*   588    594*
                        603*   609*   628*   632*   642*   643    811*   813*   814    1113*  1142*  1143   1153*
                        1167*  1169*  1247*  1248   1307*  1308   1336*  1337   1353*  1354   1356*  1357   1371*
                        1372   1374*  1375   1442*  1445   1446   1452*  1557*  1563*  1612*  1623*  1651*  1679*
                        1929*  1937*  2234*  2395   2396   2397   2399
SVM      020630        1134   2277#
SWDIST   014560        1785   1790#
SWR      001140          59#   191*   203    299    1158   1160   1172   1899   2402*  2406   2419   2421*
SWREG    000176          50#   191    203    2402
SW0   =  000001          21#
SW00  =  000001          21#
SW01  =  000002          21#
SW02  =  000004          21#
SW03  =  000010          21#
SW04  =  000020          21#
SW05  =  000040          21#
SW06  =  000100          21#
SW07  =  000200          21#
SW08  =  000400          21#
SW09  =  001000          21#
SW1   =  000002          21#
SW10  -  002000          21#
SW11  =  004000          21#
SW12  =  010000          21#   299
SW13  =  020000          21#
SW14  =  040000          21#
SW15  =  100000          21#
SW2   =  000004          21#
SW3   =  000010          21#
SW4   =  000020          21#
SW5   =  000040          21#
SW6   =  000100          21#
SW7   =  000200          21#
SW8   =  000400          21#
SW9   =  001000          21#
TBITVE=  000014          21#
TCHAN    017721        1384   1393   2261#
TCHANL   011446        1203   1214   1283#
TCHK     010014        1030#  1076   1100
TEMP     001426         136#   331*   357    374    777    780*   824*   829*   858    887    963    997*   1001*
                        1002*  1021   1093*  1097*  1098*  1490   1635*  1640*  1663*  1668*  1691*  1696*  1933*
                        1939*  1942*  1943*  1944*  1945*  1951   1966*  1976*  1979*  1980*  1981*  1982*  2398
TEMP1    001430         137#   186*   189*   198    225
TENS     024546        2047*  2057*  2058   2060*  2069*  2391#
TEST     005362         643#   2432
TESTAD   002772         291#   1189   1197   1212
TESTID=  104417        1117   1120   1123   1126   2435#
TESTIT   005354         642#   2433
TESTR2   016116        2050#  2055   2059   2063   2066
```

```
THOUS    024543          2049*   2065*   2067*   2071    2073    2388#
TKVEC = 000060            21#    2402*
TM15     020433          371     2271#
TOFF     007772          795     1021#   1075    1099
TPRMP    010532          1138#   2435
TPVEC = 000064            21#
TP15     020424          354     2270#
TRAPVE= 000034            21#     191*
TRFRO    017074          2173*   2238#
TRTO     017072          2166*   2167*   2168    2220    2226    2237#
TRTVEC= 000014            21#
TRY      013262          1551#   1574
TSTAD    020000          1251    2263#
TSTADM   020022          1257    2264#
TSTDAC   007744          861     865     890     894     1010#   1505    1510
TST1     003266          344#    346
TST10    003700          418#
TST11    003714          423#
TST12    003730          428#
TST13    003744          432#
TST14    004002          440#
TST15    004034          447#
TST16    004104          457#
TST17    004146          467#
TST2     003424          365     370#
TST20    004176          476#
TST21    004242          484     486#
TST22    004320          495     497#
TST23    004500          524#
TST24    004654          549#
TST25    004734          564#
TST26    005024          577#
TST27    005110          590#
TST3     003516          348     350     382     388#
TST30    005174          605#
TST31    005406          651#    652
TST32    005464          661#
TST33    005514          669#
TST34    005544          676#
TST35    005606          687#
TST36    005636          695#
TST37    005666          704#
TST4     003604          400#
TST40    005774          712     723#
TST41    006240          730     732     765     772#
TST42    006326          785#
TST43    006440          804     807#
TST44    006566          835#
TST45    007366          936#
TST46    007636          976     984#
TST5     003620          404#
TST6     003650          410#
TST7     003664          414#
TST8     015142          1872    1874#
```

```
TYPBAD  014262      1716    1726    1730#
TYPBN = 104406      2141    2431#
TYPDC = 104416      1024    1478    1739    1747    1757    1761    1764    1777    1832    1915    2006    2434#
TYPDS = 104405       320    2130    2136    2216    2420    2431#
TYPE  = 104401       202     203     231     232     233     241     250     288     318     321     354     359     364
                     366     371     376     381     383     622     625     733     734     766     767     786     794
                     801     805     839     844     868     873     877     897     902     906     909     937     942
                    1023    1025    1033    1035    1054    1058    1065    1069    1072    1080    1083    1084    1087
                    1088    1104    1108    1114    1116    1119    1122    1125    1130    1134    1139    1154    1162
                    1176    1183    1202    1225    1228    1251    1257    1262    1263    1285    1291    1293    1294
                    1295    1298    1303    1312    1323    1327    1329    1339    1384    1386    1388    1393    1395
                    1399    1407    1414    1416    1421    1424    1479    1481    1483    1485    1493    1495    1521
                    1584    1589    1733    1738    1740    1748    1751    1755    1758    1762    1765    1768    1772
                    1779    1782    1786    1793    1794    1803    1810    1833    1835    1840    1844    1881    1882
                    1885    1888    1909    1916    1918    1992    2007    2041    2073    2075    2130    2134    2139
                    2142    2213    2217    2219    2221    2402    2419    2420    2421    2424    2425    2428    2429
                    2431#
TYPEDG  013122      1482    1516#   1917
TYPOC = 104402      1415    2402    2420    2431#
TYPON = 104404      2431#
TYPOS = 104403       624     789     842     940    1163    1177    1301    1311    1320    1385    1387    1394    1398
                    1423    1480    1484    1490    1517    1522    1587    1737    1834    1837    1919    2218    2220
                    2431#
TYPOUT  016212      2051    2067#
TYPRP   015314       870     899    1912#
TYPSET  012704       974    1478#
UNEXP   001516      165#    405
UNITBD  001514      163#     624     788     841     939    1300    1586    2395    2396    2397    2398    2399    2409*
                    2412*
VARLT1  016372      2083    2117#
VARLT2  016402      2085    2124#
VECTOR  001360      117#    405*    504*    516*   1164*   1444*   1455*   1459*   1460    1462    1464    1556*   1614*
                    1934*   2218    2222*   2226*   2227*   2228    2230
VECTR1  001362      118#    406*    505*    516     517*   1460*   1461*   1935*   2222    2224*   2228*   2229*
VECTR2  001364      119#    531*    543*   1462*   1463*   2223*   2230*   2231*   2232
VECTR3  001366      120#    532*    543     544*   1464*   1465*   2223    2225*   2232*   2233*
VLIN    016366      1838    2112#
VNP     016362       900    2110#
VNR     016360       871    2080    2109#
VOLTS   020442      2007    2272#
VSET    016364      1491    2111#
VTABLE  016340       709     719    2100#
VTINIT  023114      1909    2356#
VTMSG   017524      1298    2213    2256#
VTMSG1  017601      2219    2258#
VTMSG2  017631      2221    2259#
VTMSG3  017550      2217    2257#
V0      016322       827     832    2091#
V10     016326      1047    2093#
V100D   016334       362     379    2096#
V12     016330       659     684    2094#
V2      016324       783    2092#
V326    016336       666     674     692     700     716     755    2097#
V50D    016332       798    2095#
```

```
WFADJ    016266           221    2080#
WFCHK    011254          1187    1198    1213    1244#
WFTEST   001476           156#    177*    188*    227     236*    240*    316     347     591     606     610     677     724
                          761     845     918     943    1244    1288    1342    2081
WHICHU   026572           623     787     840     938    1299    1475    1585    2214    2408#
WHICHV   026564           367     384     802     878     907    1496    1752    1769    1783    1841    2407#   2419
WIDE     001416           132#    836*    911    1236*   1602*   1729*   1759    1774
WIDMSG   021704          1762    2308#
WRAP     005406           650#   1205    1216
WRAPX    005666           654     703#
XADJ     021322          1072    2297#
YADJ     021567          1087    2303#
YESNO    017745           232    1407    2262#
YORNO    020476          1263    2274#
$APTHD   001000            58#
$ASTAT=  ******  U        2426
$ATYC    030142          2426#
$ATY1    030116          2426#
$ATY3    030124          2424    2426#
$ATY4    030134          2419    2426#
$AUTOB   001134            59#    203*    222     618    2402
$BASE    001244            59#    291    1415    1420*   1452    1453    1454
$BDADR   001122            59#
$BDDAT   001126            59#    291*    294     295*    392*    393*    394     483*    493*    494     509*    534*    569*
                          570*    571     583*    584*    643*    644    1145*   1146*   1147    1951*   1952    2395    2396
                         2397
$BELL    026134          2402#
$BIN     030664          2429#*
$CDW1    001250            59#
$CHARC   027666          2424#*
$CKSWR   025206          2402#   2431
$CMTAG   001100            59#    191
$CM3 =   000000            59#
$CNTLC   026140          2402#
$CNTLG   026152          2402#
$CNTLU   026145          2402#
$CPUOP   001216            59#
$CRLF    001165            59#    231     844     909     942    1033    1183    1285    1303    1339    1388    1589    2402
                         2419    2420    2424
$DBLK    030106          2425#
$DEVCT   001200            59#
$DEVM    001246            59#
$DOAGN   016510          2130#
$DTBL    030076          2425#
$ENDAD   016500            56     203    2130#   2419
$ENDCT   016446           191    2130#
$ENDMG   016517          2130#
$ENULL   016514          2130#   2142
$ENV     001210            59#    203     297    2419    2424    2426
$ENVM    001211            59#    191    2424    2426
$EOP     016412           311     325    1194    1209    1220    1239    2130#
$EOPCT   016440           191*   2130#
$ERFLG   001103            59#    166*   2406*   2419*
$ERMAX   001115            59#    191*   2406*
```

$ERROR	026626	191	2419#											
$ERRPC	001116	59#	2395	2396	2397	2398	2399	2419*	2420					
$ERRTB	001252	59#	2420											
$ERRTY	027016	2419	2420#											
$ERTTL	001112	59#	248*	368*	385*	803*	879*	908*	1497*	1753*	1770*	1784*	1842*	2132
		2135	2419*											
$ESCAP	001162	59#	165*	168*	191*	2406*	2419							
$ETABL	001210	59#												
$ETEND	001252	58	59#											
$FATAL	001172	59#	2426*											
$FFLG	030362	2426#*												
$FILLC	001156	59#	2424											
$FILLS	001155	59#	2424											
$GDADR	001120	59#												
$GDDAT	001124	59#	390*	391	394	397*	398	401*	407*	411*	415*	419*	424*	429*
		433*	450*	458*	468*	477*	487*	494	511*	512*	535*	554*	565*	571
		578*	593*	594	608*	609	642	644	1115*	1128	1132	1138*	1147	1949*
		1953	2395	2396										
$GET42	016470	2130#												
$GTSWR	025276	2402#	2431											
$HD =	000000	20												
$HIBTS	001000	58#												
$HIOCT	026302	2404#*												
$ICNT	001104	59#	2406*											
$ILLUP	027336	2421#												
$INTAG	001135	59#	2402*											
$ITEMB	001114	59#	2419*	2420										
$LF	001166	59#	2402	2419	2424									
$LFLG	030361	2426#*												
$LPADR	001106	59#	191*	344*	498*	525*	652*	737*	2236	2406*				
$LPERR	001110	59#	191*	346*	652*	738*	2406*	2419						
$MADR1	001222	59#												
$MADR2	001226	59#												
$MADR3	001232	59#												
$MADR4	001236	59#												
$MAIL	001170	58	59#	191	203	2406	2419	2424						
$MAMS1	001220	59#												
$MAMS2	001224	59#												
$MAMS3	001230	59#												
$MAMS4	001234	59#												
$MBADR	001002	58#												
$MFLG	030360	2426#*												
$MNEW	026170	2402#												
$MSGAD	001204	59#	2426*											
$MSGLG	001206	59#	2426*											
$MSGTY	001170	59#	2426*											
$MSWR	026157	2402#												
$MTYP1	001221	59#												
$MTYP2	001225	59#												
$MTYP3	001231	59#												
$MTYP4	001235	59#												
$MXCNT	026562	2406#												
$NULL	001154	59#	2424											
$NWTST=	000001	344#	370#	388#	400#	404#	410#	414#	418#	423#	428#	432#	440#	447#

```
                        457#   467#   476#   486#   497#   524#   549#   564#   577#   590#   605#   651#   661#
                        669#   676#   687#   695#   704#   723#   772#   785#   807#   835#   936#   984#
$OCNT    030606         2428#*
$OMODE   030610         2428#*
$OVER    026546         2406#
$PASS    001176          59#   191*   247*   349    620    729    987   1286   2130*  2211   2406
$PASTM   001006          58#
$PWRAD   027332         2421#
$PWRDN   027172         191    2421#
$PWRMG   027326         2421#
$PWRUP   027244         2421#
$QUES    001164          59#   1065   2402   2419   2424
$RDCHR   025550         2402#  2431
$RDDEC=  ****** U       2431
$RDLIN   025640         2402#  2431
$RDOCT   026202         2404#  2431
$RDSZ =  000040         2402#
$RTNAD   016512         2130#
$R2A  =  ****** U       2431
$SAVRE=  ****** U       2431
$SAVR6   027342         2421#*
$SCOPE   026304         191    2406#
$SETUP=  000137          60#   191    203    2130   2402   2406   2419
$STUP =  177777          60#
$SVLAD   026512         2406#
$SVPC =  000214          56#
$SWR  =  165400           5#    20     22     59    191    344    370    388    400    404    410    414    418
                        423    428    432    440    447    457    467    476    486    497    524    549    564
                        577    590    605    651    661    669    676    687    695    704    723    772    785
                        807    835    936    984    2130   2406   2419   2421
$SWREG   001212          59#   191
$SWRMK=  000000          22     2406
$TESTN   001174          59#   2406*
$TIMES   001160          59#   191*   344*   370*   432*   441*   447*   457*   497*   524*   605*   651*   661*
                        669#   676*   687*   695*   704*   723*   726*   772*   785*   808*   835*   936*   984*
                        2130*  2406*
$TKB     001146          59#   1043   2402
$TKCNT   024652         2402#*
$TKINT   024720         197    2402#
$TKQEN=  024720         2402#
$TKQIN   024654         2402#*
$TKQOU   024656         2402#*
$TKQSR   024660         2402#
$TKS     001144          59#   243*   436*   444*   463*   499*   526*   551*   764*   1037*  1038   1044*  1265*
                        2131*  2402*
$TKSRV   024770         2402#
$TN   =  000047           6#    20     344#   345    365    370#   382    388#   389    400#   404#   410#   414#
                        418#   423#   428#   432#   440#   447#   457#   467#   476#   484    486#   495    497#
                        524#   549#   564#   577#   590#   605#   651#   652    661#   669#   676#   687#   695#
                        704#   705    712    723#   730    732    765    772#   773    785#   804    807#   835#
                        936#   976    984#
$TPB     001152          59#   2024*  2034*  2424*
$TPFLG   001157          59#   2424
$TPS     001150          59#   2022   2032   2424
```

CVMNA-A MNCAD/MNCAM   DIAGNOSTIC   M 9
MACY11 27(654) 19-SEP-78 08:51 PAGE 47-15
CVMNAA.P11   CROSS REFERENCE TABLE   SEQ 0116

```
$TRAP     030666        191     2431#
$TRAP2    030710       2431#
$TRP    = 000021       2431#    2432#   2433#   2434#   2435#   2436#
$TRPAD    030722       2431#
$TSTM     001004         58#
$TSTNM    001102         59#     345*    389*    651*    705*    773*    2130*   2174    2406*   2419
$TTYIN    026074       2402#
$TYPBN    030612       2429#    2431
$TYPDS    027672       2425#    2431
$TYPE     027410        193*    194*    195*    196*    2424#   2426    2431
$TYPEC    027622       2402     2424#
$TYPEX    027670       2424#
$TYPOC    030410       2428#    2431
$TYPON    030424       2428#    2431
$TYPOS    030364       2428#    2431
$UNIT     001202         59#     292*    296*    301     305     319     322     326     329     333*    2215    2398
$UNITM    001010         58#
$USWR     001214         59#
$VECT1    001240         59#    1422    1428*   1429*   1455
$VECT2    001242         59#
$XTSTR    026316       2406#
$$GET4=   000000       2130#
$OFILL    030607       2428#*
$4OCAT= ****** U       2406     2419
.       = 051604         26#      31#     42      43#     46#     48#     56#     57#     58#     59#     191     203#    207
                       2130     2393#   2402#   2406    2419    2420#   2421    2424    2425#   2426#   2437#   2438#
.$ASTA= ****** U       2426
.$X     = 001000         58#
```

ASCIZ	8#													
BERROR	2416#	2419												
COMMEN	21#													
DUMWRN	64#													
ENDCOM	21#													
ERROR	21#	167	309	332	396	403	409	413	417	421	426	431	438	446	455
	465	474	485	496	513	538	560	573	586	601	630	660	667	675	685
	693	701	717	756	784	828	833	1118	1121	1124	1127	2177			
ESCAPE	21#	165												
GETPRI	21#													
GETSWR	21#	203#												
MULT	21#													
NEWTST	21#	344	370	388	400	404	410	414	418	423	428	432	440	447	457
	467	476	486	497	524	549	564	577	590	605	651	661	669	676	687
	695	704	723	772	785	807	835	936	984						
POP	21#	2404	2421	2425	2426									
PUSH	21#	2404	2421	2425	2426									
REPORT	21#													
SCOPE	21#	370	388	400	404	410	414	418	423	428	432	440	447	457	467
	476	486	497	524	549	564	577	590	605	636	661	669	676	687	695
	704	723	772	785	807	835	936	984							
SET	12#	652												
SETPRI	21#	2402												
SETTRA	2431#	2432	2433	2434	2435	2436								
SETUP	21#	191												
SKIP	21#	255	258	261	264	267	270	273	275	279	283	286	348	363	365
	380	382	399	484	495	514	572	585	589	592	607	611	617	619	621
	645	678	707	712	720	721	725	728	730	732	742	751	762	765	804
	846	849	855	874	876	884	903	905	914	916	919	921	923	925	931
	944	947	949	954	960	976	980	982	986	1011	1013	1022	1031	1034	1039
	1041	1042	1048	1062	1064	1066	1077	1078	1101	1102	1129	1133	1148	1161	1245
	1249	1255	1261	1272	1275	1289	1290	1292	1306	1315	1318	1322	1324	1326	1328
	1332	1338	1343	1346	1350	1355	1358	1361	1363	1368	1373	1376	1379	1381	1404
	1406	1408	1900	1902	1990	1998	2002	2063	2072	2082	2084	2088	2212		
SLASH	21#	2176	2207											
SPACE	21#													
STARS	21#	56	58	59	344	370	388	400	404	410	414	418	423	428	432
	440	447	457	467	476	486	497	524	549	564	577	590	605	651	661
	669	676	687	695	704	723	772	785	807	835	936	984	2130	2402	2404
	2406	2419	2420	2421	2424	2425	2426	2428	2429	2431					
SWRSU	21#	191#												
TRMTRP	2431#													
TYPBIN	21#													
TYPDEC	21#	2130	2420											
TYPNAM	21#	203												
TYPNUM	21#													
TYPOCS	21#	624	1163	1177	1311	1320	1385	1387	1394	1398	1423	1480	1484	1490	1517
	1522	1737	1834	1837	1919	2218	2220								
TYPOCT	21#	1415	2402	2420										
TYPTXT	21#													
$$CMRE	59#													
$$CMTM	59#													
$$ESCA	21#													
$$NEWT	21#	344	370	388	400	404	410	414	418	423	428	432	440	447	457

	467	476	486	497	524	549	564	577	590	605	651	661	669	676	687
	695	704	723	772	785	807	835	936	984						
$$SET	2431#	2432	2433	2434	2435	2436									
$$SETM	191#														
$$SKIP	21#	365	382	484	495	712	730	732	765	804	976				
.EQUAT	7#	21													
.HEADE	7#	20													
.SETUP	9#	60													
.SWRHI	9#	22													
.SWRLO	22#														
.$ACT1	10#	56													
.$APTB	10#	59#													
.$APTH	10#	58													
.$APTY	10#	2426													
.$CATC	7#														
.$CMTA	7#	59													
.$EOP	7#	2130													
.$ERRO	7#	2419													
.$ERRT	9#	2420													
.$PARM	8#														
.$POWE	8#	2421													
.$RAND	10#														
.$RDDE	7#														
.$RDOC	10#	2404													
.$READ	8#	2402													
.$SAVE	8#														
.$SCOP	8#	2406													
.$SPAC	9#														
.$SWDO	9#														
.$TRAP	9#	2431													
.$TYPB	8#	2429													
.$TYPD	10#	2425													
.$TYPE	9#	2424													
.$TYPO	8#	2428													

ADC	1618	1646	1674	1712	1830	1854	1945	1982	2005					
ADCB	2429													
ADD	172	295	587	646	747	950	1001	1049	1050	1097	1138	1149	1313	1316	1347
	1362	1365	1397	1409	1440	1443	1449	1458	1461	1463	1465	1506	1512	1520	1552
	1616	1617	1644	1645	1672	1673	1760	1774	1775	1814	1855	1939	1976	1999	2229
	2231	2233	2402	2404	2420	2424	2425	2426	2428					
ASL	397	1435	1545	1546	1547	1548	1630	1658	1686	1717	2402	2404	2420	2431
ASLB	2425													
ASR	746	1438	1573	1709	1710	1711	1719	1736	1824	1827	1828	1829	1851	1852	1853
	1942	1943	1944	1979	1980	1981	2000	2001	2003	2004	2025	2026	2027	2028	2029
	2410	2426												
BCC	2425													
BEQ	191	203	223	273	330	395	484	495	572	585	589	607	849	855	884
	914	921	931	947	949	960	988	1011	1013	1062	1064	1173	1249	1261	1289
	1332	1338	1350	1361	1368	1404	1406	1419	1427	1434	1439	1519	1621	1627	1629
	1638	1643	1649	1655	1657	1666	1671	1677	1683	1685	1694	1750	1767	1791	1846
	1900	1998	2051	2072	2130	2133	2138	2402	2404	2406	2411	2419	2420	2424	2426
	2428	2429												
BGE	820	1566	1721	1869	2406									
BGT	872	901	1492	1571	1958	2130	2402	2425	2428					
BHI	1318	2406												
BHIS	742													
BIC	237	393	453	499	526	570	584	614	628	759	1146	1159	1253	1259	1314
	1360	1378	1459	1620	1648	1576	1993	2020	2030	2130	2227	2402	2404	2428
BICB	253	1270	1402											
BIS	243	327	436	444	463	551	613	727	764	1112	1166	1265	1335	1352	1370
	1429	1557	1562	1611	1928	2021	2031	2131	2402	2407	2425	2428		
BISB	1974	2067	2068	2069	2070	2420								
BIT	299	588	1160	1172	1248	1325	1337	1354	1372	1899	1989	1997	2406	2419
BITB	191	2424	2426											
BLE	817	1728	1781	1820	1839	1872	2044	2169	2175					
BLO	712	732	751	925	976	2402								
BLOS	2402													
BLT	1714	2017	2402	2424	2425	2428								
BMI	228	298	315	317	923	1022	1039	1641	1669	1697	2082	2425		
BNE	171	191	199	201	203	226	239	255	258	261	264	267	270	275	279
	283	286	300	302	306	308	323	350	399	452	558	599	616	619	621
	645	654	678	707	720	730	823	912	916	919	986	1031	1041	1129	1133
	1148	1161	1180	1182	1231	1272	1275	1287	1315	1322	1326	1343	1355	1373	1379
	1474	1533	1569	1574	1580	1597	1608	1699	1701	1724	1731	1742	1802	1826	1858
	1875	1880	1896	1906	1908	1932	1941	1970	1978	1990	2002	2055	2059	2063	2212
	2402	2406	2419	2420	2421	2424	2425	2426	2428					
BFL	348	461	471	482	492	508	553	582	592	611	725	762	815	846	944
	970	1144	1245	1290	1309	1358	1376	1632	1660	1688	1817	1913	1955	2014	2023
	2033	2040	2088	2402	2419	2424	2425	2428						
BR	178	187	191	203	211	277	289	303	328	363	365	380	382	514	539
	617	721	728	760	765	799	800	804	874	876	903	905	954	966	968
	980	982	1034	1042	1048	1066	1077	1078	1081	1101	1102	1105	1131	1135	1175
	1184	1192	1201	1207	1218	1255	1292	1306	1324	1328	1346	1363	1381	1408	1441
	1448	1634	1636	1662	1664	1690	1692	1716	1726	1754	1771	1785	1843	1902	2066
	2084	2172	2402	2404	2406	2413	2419	2420	2421	2424	2425	2426	2428	2429
CLC	2429													
CLR	168	177	189	191	244	247	248	249	292	333	337	338	351	433	458
	468	477	478	479	489	500	510	518	521	527	536	541	545	548	576

	578	603	632	774	775	791	792	810	836	852	881	957	964	1036	1037
	1153	1155	1224	1246	1266	1283	1284	1304	1305	1344	1436	1550	1551	1595	1600
	1601	1602	1603	1604	1605	1606	1635	1663	1691	1799	1808	1809	1903	1914	1933
	1966	1994	2015	2130	2234	2402	2404	2406	2420	2421	2425	2428			
CLRB	236	2046	2047	2048	2049	2056	2060	2064	2402	2406	2424	2425	2426		
CMP	170	191	203	212	301	304	394	398	494	515	540	571	644	711	719
	731	741	750	816	819	848	854	871	883	900	913	915	920	924	930
	946	959	975	1012	1128	1132	1147	1254	1260	1317	1331	1342	1349	1367	1473
	1491	1518	1532	1565	1570	1628	1637	1656	1665	1684	1693	1713	1720	1727	1780
	1819	1825	1838	1868	1871	1878	1957	2016	2043	2168	2209	2210	2402	2406	2419
	2425														
CMPB	203	238	254	257	260	263	266	269	272	274	278	282	285	329	1061
	1063	1271	1274	1403	1405	2054	2058	2062	2071	2174	2402	2406	2419	2424	2426
DEC	335	451	557	598	615	822	1030	1179	1181	1382	1447	1568	1579	1596	1607
	1642	1670	1698	1700	1741	1801	1857	1874	1879	1895	1905	1907	1931	1940	1969
	1977	2052	2130	2402	2420										
DECB	2424	2428													
EMT	21														
HALT	1276	1901	2170	2419	2421	2424									
INC	166	186	203	280	296	368	385	449	459	469	580	718	757	803	853
	879	882	908	910	917	952	958	1040	1142	1169	1307	1330	1356	1374	1457
	1497	1567	1623	1631	1639	1640	1651	1659	1667	1668	1679	1687	1695	1696	1715
	1718	1722	1725	1729	1732	1753	1770	1784	1836	1842	1937	2130	2402	2406	2412
	2419	2421	2425	2426	2428										
INCB	240	813	2053	2057	2061	2065	2406	2419	2424						
IOT	21														
JMP	52	53	54	224	256	259	262	265	268	271	281	284	287	311	325
	679	838	922	926	932	1016	1079	1103	1136	1194	1209	1220	1239	1277	1430
	2130	2143	2236	2402											
JSR	197	217	221	336	355	358	360	367	372	375	377	384	503	530	623
	655	657	662	664	670	672	680	682	688	690	696	698	713	714	752
	753	776	779	781	787	793	795	796	802	825	830	837	840	857	859
	861	863	865	870	878	886	888	890	892	894	899	907	938	962	965
	967	974	989	995	999	1045	1074	1075	1076	1091	1095	1099	1100	1187	1189
	1190	1191	1197	1198	1199	1200	1203	1205	1206	1212	1213	1214	1216	1217	1223
	1237	1299	1345	1380	1475	1482	1488	1496	1503	1505	1508	1510	1585	1752	1769
	1783	1792	1798	1800	1804	1841	1884	1887	1889	1894	1917	2130	2214	2402	2419
	2424	2426													
MOV	165	179	180	181	182	183	184	185	188	191	193	194	195	196	204
	205	206	208	209	210	213	214	215	216	218	219	220	235	245	252
	291	293	312	313	319	326	339	344	345	346	352	357	370	374	389
	390	391	392	401	405	406	407	411	415	419	424	429	432	434	441
	442	447	448	450	456	457	472	480	483	487	488	490	493	497	498
	501	504	505	506	509	511	512	516	517	519	524	525	528	531	532
	533	534	535	543	544	546	550	554	555	556	562	565	566	569	575
	579	583	593	594	595	596	597	605	608	609	612	624	642	643	651
	652	661	669	676	687	695	704	705	706	708	709	710	723	726	737
	738	739	740	743	744	749	772	773	777	778	785	788	808	809	811
	812	818	821	824	829	835	841	847	850	851	856	858	862	867	869
	875	880	885	887	891	896	898	904	928	936	939	945	948	951	953
	955	956	961	963	972	973	977	978	979	981	984	994	997	998	1021
	1032	1044	1056	1057	1060	1073	1089	1090	1093	1094	1110	1113	1115	1145	1156
	1158	1163	1164	1167	1168	1171	1174	1177	1178	1188	1193	1208	1219	1227	1230
	1232	1236	1238	1247	1250	1252	1256	1258	1267	1269	1300	1310	1311	1319	1320

	1333	1336	1348	1353	1359	1364	1366	1371	1377	1383	1385	1387	1394	1396	1398
	1401	1415	1420	1422	1423	1428	1437	1442	1444	1445	1446	1450	1451	1452	1453
	1454	1455	1456	1460	1462	1464	1467	1468	1469	1470	1471	1480	1484	1486	1487
	1490	1501	1502	1507	1516	1517	1522	1526	1527	1528	1529	1530	1544	1549	1555
	1554	1556	1559	1560	1563	1586	1590	1591	1592	1593	1594	1598	1599	1609	1612
	1613	1614	1615	1619	1626	1633	1647	1654	1661	1675	1682	1689	1706	1707	1708
	1734	1737	1746	1756	1759	1763	1773	1776	1795	1796	1797	1811	1812	1815	1816
	1821	1822	1831	1834	1837	1847	1848	1850	1856	1859	1860	1861	1862	1863	1864
	1865	1866	1867	1870	1873	1876	1877	1883	1886	1891	1892	1904	1919	1923	1924
	1925	1929	1930	1934	1935	1936	1949	1950	1951	1952	1953	1963	1964	1967	1968
	1971	1972	1973	1995	1996	2018	2019	2045	2080	2083	2085	2086	2130	2135	2140
	2166	2173	2215	2218	2220	2222	2223	2224	2225	2226	2228	2230	2232	2402	2404
	2406	2408	2409	2419	2420	2421	2424	2425	2426	2428	2429	2431			
MOVB	191	203	331	334	567	568	1555	1745	1778	1926	2024	2034	2402	2404	2406
	2419	2424	2425	2426	2428	2429	2431								
NEG	758	971	1818	1956	1991	2042	2425	2428							
NOP	1233	1234	1235	1534	1535	1625	1653	1681	2130						
RESET	190	242	435	443	462	763	2130								
ROL	2404	2428	2429												
RTI	169	173	191	196	246	340	353	502	520	529	547	647	1150	1157	1268
	1581	2074	2076	2208	2402	2404	2406	2419	2421	2424	2425	2428	2429	2431	
RTS	341	637	933	990	1003	1014	1026	1051	1273	1389	1410	1476	1494	1498	1513
	1523	1536	1575	1890	1897	1910	1920	1946	1960	1983	2008	2035	2089	2402	2415
	2420	2424	2426	2431											
SEC	2429														
SOB	207	1622	1650	1678											
SUB	745	780	866	895	929	1002	1098	1511	1572	1735	1813	1823	1954	2167	2419
	2425	2426													
SWAB	1111	1165	1334	1351	1369	1561	1610	1927	1965						
TRAP	2431	2432	2433	2434	2435	2436									
TST	198	200	203	225	227	294	297	305	307	310	314	316	322	324	347
	349	522	537	542	591	602	606	610	620	627	631	653	677	724	729
	736	761	769	845	911	918	943	969	985	987	1010	1015	1043	1071	1086
	1141	1244	1286	1288	1297	1321	1418	1426	1433	1472	1531	1723	1730	1749	1766
	1790	1845	1893	1912	1959	2013	2039	2050	2081	2087	2132	2137	2211	2235	2402
	2404	2406	2419	2420	2424	2425	2426	2428	2431						
TSTB	222	460	470	481	491	507	552	581	618	814	1038	1143	1308	1357	1375
	2022	2032	2402	2406	2420	2424	2425	2426							
WAIT	1170	1558	1564	1624	1652	1680	1938	1975							
.ASCII	59	2259	2297	2298	2300	2314	2317	2318	2319	2323	2365	2366	2367	2368	2369
	2370	2371	2372	2373											
.ASCIZ	59	203	2130	2241	2242	2243	2244	2245	2246	2249	2250	2251	2252	2253	2254
	2255	2256	2257	2258	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270
	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285
	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296	2299	2301	2302	2303
	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2315	2316	2320	2321	2322
	2324	2325	2326	2327	2328	2329	2330	2331	2374	2376	2377	2378	2379	2380	2381
	2382	2383	2384	2385	2386	2387	2402	2420	2422						
.BLKB	2402														
.BLKW	2425	2437	2438												
.BYTE	59	624	790	843	941	1163	1177	1302	1311	1320	1385	1387	1394	1398	1423
	1480	1484	1490	1517	1522	1588	1737	1834	1837	1919	2130	2218	2220	2247	2248
	2333	2334	2335	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347
	2348	2349	2350	2351	2352	2354	2355	2356	2357	2358	2359	2360	2375	2388	2389

	2390	2391	2392	2400	2402	2419	2426	2428	2429						
.DSABL	2402														
.ENABL	4	2402													
.FND	2440														
.ENDC	20	21	22	56	58	59	60	165	191	203	255	258	261	264	267
	270	273	275	279	283	286	344	348	363	365	370	380	382	388	399
	400	404	410	414	418	423	428	432	440	447	457	467	476	484	486
	495	497	514	524	549	564	572	577	585	589	590	592	605	607	611
	617	619	621	624	645	651	661	669	676	678	687	695	704	707	712
	720	721	723	725	728	730	732	742	751	762	765	772	785	804	807
	835	846	849	855	874	876	884	903	905	914	916	919	921	923	925
	931	936	944	947	949	954	960	976	980	982	984	986	1011	1013	1022
	1031	1034	1039	1041	1042	1048	1062	1064	1066	1077	1078	1101	1102	1129	1133
	1148	1161	1163	1177	1245	1249	1255	1261	1272	1275	1289	1290	1292	1306	1311
	1315	1318	1320	1322	1324	1326	1328	1332	1338	1343	1346	1350	1355	1358	1361
	1363	1368	1373	1376	1379	1381	1385	1387	1394	1398	1404	1406	1408	1423	1480
	1484	1490	1517	1522	1737	1834	1837	1900	1902	1919	1990	1998	2002	2063	2072
	2082	2084	2088	2130	2176	2207	2212	2218	2220	2402	2404	2406	2419	2420	2421
	2424	2425	2426	2428	2429	2431	2432	2433	2434	2435	2436				
.EQUIV	21														
.EVEN	59	203	2393	2402	2420	2426									
.IF	20	21	22	56	58	59	60	165	191	203	255	258	261	264	267
	270	273	275	279	283	286	344	348	363	365	370	380	382	388	399
	400	404	410	414	418	423	428	432	440	447	457	467	476	484	486
	495	497	514	524	549	564	572	577	585	589	590	592	605	607	611
	617	619	621	624	645	651	661	669	676	678	687	695	704	707	712
	720	721	723	725	728	730	732	742	751	762	765	772	785	804	807
	835	846	849	855	874	876	884	903	905	914	916	919	921	923	925
	931	936	944	947	949	954	960	976	980	982	984	986	1011	1013	1022
	1031	1034	1039	1041	1042	1048	1062	1064	1066	1077	1078	1101	1102	1129	1133
	1148	1161	1163	1177	1245	1249	1255	1261	1272	1275	1289	1290	1292	1306	1311
	1315	1318	1320	1322	1324	1326	1328	1332	1338	1343	1346	1350	1355	1358	1361
	1363	1368	1373	1376	1379	1381	1385	1387	1394	1398	1404	1406	1408	1423	1480
	1484	1490	1517	1522	1737	1834	1837	1900	1902	1919	1990	1998	2002	2063	2072
	2082	2084	2088	2130	2176	2207	2212	2218	2220	2402	2404	2406	2419	2420	2421
	2424	2425	2426	2428	2429	2431	2432	2433	2434	2435	2436				
.IFF	21	22	56	58	59	165	191	203	255	258	261	264	267	270	273
	275	279	283	286	344	348	363	365	370	380	382	388	399	400	404
	410	414	418	423	428	432	440	447	457	467	476	484	486	495	497
	514	524	549	564	572	577	585	589	590	592	605	607	611	617	619
	621	624	645	651	661	669	676	678	687	695	704	707	712	720	721
	723	725	728	730	732	742	751	762	765	772	785	804	807	835	846
	849	855	874	876	884	903	905	914	916	919	921	923	925	931	936
	944	947	949	954	960	976	980	982	984	986	1011	1013	1022	1031	1034
	1039	1041	1042	1048	1062	1064	1066	1077	1078	1101	1102	1129	1133	1148	1161
	1163	1177	1245	1249	1255	1261	1272	1275	1289	1290	1292	1306	1311	1315	1318
	1320	1322	1324	1326	1328	1332	1338	1343	1346	1350	1355	1358	1361	1363	1368
	1373	1376	1379	1381	1385	1387	1394	1398	1404	1406	1408	1423	1480	1484	1490
	1517	1522	1737	1834	1837	1900	1902	1919	1990	1998	2002	2063	2072	2082	2084
	2088	2130	2176	2207	2212	2218	2220	2402	2404	2406	2419	2420	2421	2424	2425
	2426	2428	2429	2431											
.IFT	203	2402	2404	2406	2419										
.IFTF	203	2402	2404	2406	2419										
.IIF	20	22	59	191	203	624	1163	1177	1311	1320	1385	1387	1394	1398	1415

	1423	1480	1484	1490	1517	1522	1737	1834	1837	1919	2130	2218	2220	2402	2406
	2419	2420	2424	2431	2432	2433	2434	2435	2436						
.IRP	60	344	370	388	400	404	410	414	418	423	428	432	440	447	457
	467	476	486	497	524	549	564	577	590	605	651	652	661	669	676
	687	695	704	723	772	785	807	835	936	984	2130	2404	2419	2421	2425
	2426														
.LIST	3	19	21	22	45	59	60	191	203	344	370	388	400	404	410
	414	418	423	428	432	440	447	457	467	476	486	497	524	549	564
	577	590	605	651	661	669	676	687	695	704	723	772	785	807	835
	936	984	2130	2402	2406	2419	2431	2432	2433	2434	2435	2436			
.MACRO	12	22	59	64	191	2416	2431								
.MCALL	7	8	9	10	21	59	191	203							
.MEXIT	59														
.NLIST	1	2	21	22	37	59	60	191	203	344	370	388	400	404	410
	414	418	423	428	432	440	447	457	467	476	486	497	524	549	564
	577	590	605	651	661	669	676	687	695	704	723	772	785	807	835
	936	984	2130	2402	2406	2419	2431	2432	2433	2434	2435	2436			
.PAGE	59														
.REPT	39														
.SBTTL	21	22	29	56	58	59	113	176	191	203	230	290	344	370	388
	400	404	410	414	418	423	428	432	440	447	457	467	476	486	497
	524	549	564	577	590	605	649	651	661	669	676	687	695	704	723
	772	785	807	835	936	984	1053	1107	1152	1186	1196	1211	1222	1432	2130
	2240	2363	2402	2404	2406	2419	2420	2421	2424	2425	2426	2428	2429	2431	
.TITLE	20														
.WORD	27	38	44	47	49	50	56	58	59	2130	2237	2238	2402	2404	2420
	2421	2424	2426	2428	2431										

ERRORS DETECTED: 0

```
*CVMNAA,CVMNAA/CRF=CVMNAA
RUN-TIME: 30 17 3 SECONDS
CORE USED:  27K
```