MNCAD,
MNCTP

MNCAD DIAG
CVMNACO

AH-B086C-MC
FICHE 1 OF 1

OCT 1981
COPYRIGHT © 78-81
MADE IN USA

digital

IDENTIFICATION

Product Code: AC-B085C-MC

Diagnostic Code: MAINDEC-11-CVMNA-C-D

Product Name: CVMNAC0 - MNCAD Performance Test

Date: Aug. 1981

Maintainer: Diagnostic Group

1.0 ABSTRACT

This diagnostic has three starting addresses:

200 Normal
204 Restart
210 Option checkout with tester connected
     (Tighter tolerances for the MNCAD, MNCAM and MNCAG)
214 MNCTP incoming inspection test

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

When starting the diagnostic, the operator is asked about the
presence of the test modules, clock and the type of console
terminal. A list of tests available are displayed. The operator
selects the test by the ''TEST CHARACTER'' and then depresses the
''RETURN'' key on the console. The following list indicates which
''TEST CHARACTER'' corresponds to the test or function to execute:

W: Wraparound analog tests

        Analog subtests
        Noise test
        Interchannel Settling test
        Differential Linearity and Relative Accuracy test
          (only on the second pass)

n loop:fo&atherMNCAD
    P: Print converted analog values loop
    L: Logic test (MNCAD, MNCAG and MNCTP)
    A: Auto tests

        Logic subtests
        Analog subtests
        Noise test
        Interchannel Settling test
        Differential Linearity and Relative Accuracy test
          (only on the second pass)

    Q: Quiet execution of the Auto tests without Dif-Lin
    N: Noise tests on selected channels
    D: Differential Linearity and Relative Accuracy test on a
        selected channel
    S: Settling test between two selected channels
    F: Function test of the MNCAG front panel
    T: Test MNCAG channels analog input
    M: Common mode rejection test for MNCAG channels
    I: Incomming inspection test of the MNCTP
    X: GAIN or TC TYPE register loop
    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

Computer with 20K of memory
I/O Console Terminal (LA36, VT100, etc.)
MNCAD/MNCAM/MNCAG/MNCTP Module(s)
MNCAD-TA test module <optional>
MNCAM-TA test module <optional>
MNCAG-TA test module <optional>
MNCTP-TA test module <optional>
Bit map terminal for graphic output (I.E. VT105, VT55)
<optional>

2.2 Storage

This program uses 16K of memory.

3.0 LOADING PROCEDURE

Procedure for loading normal binary file should be followed.

4.0 STARTING PROCEDURE

4.1 Control Switch Settings

Standard PDP-11 Format

Switch	Value	Description
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>

4.2 Starting Addresses

Address	Description
200	Starting address for standard tolerances
204	Restart address
210	Starting address for tighter tolerances
214	Starting address for the MNCTP incomming test

5.0 OPERATING PROCEDURE

Start the diagnostic at 200, 210 or 214. The program requests an initial switch register value. The operator will normally depress the 'RETURN' key. The program now instructs the operator what position to set the front panel switches of the MNCAD AND MNCAG. 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. The request is repeated for the MNCAM-TA, MNCAG-TA AND MNCTP-TA test modules. The program will then request if a MNCKW (CLOCK) is available on the system. The final request asks if the console terminal is a 'BIT-MAP' terminal (IE VT105 or VT55). A list of tests, loops, or functions available will be printed out. The operator selects the 'TEST 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 or select the 'G' function. The program responds with the current contents of SWREG and a slash. Type the desired new contents of SWREG followed by a carriage return.

When a 'W' is selected, the program will report the number of MNCAD detected and will then give a channel table for the MNCAD (A/D) under test. If any test modules are connected, the program will then ask which channels to test. The program will run through the analog subtests, the noise test, the interchannel settling test, and on the second pass, the differential linearity and relative accuracy test.

If 'Q' is typed, the program will preform the same as 'W'.

If 'C' is typed, the program will ask for the channel to be used. It will then ask if the offset or gain adjustments are to be made. The operator should perform the operations made by the program. The program will run the calibration routine and loop on the MNCAD until it is calibrated and a 'RETURN' is typed. If an additional 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 ask for the channel to be used. It will then ask for the "GAIN" or "TC TYPE" to be used for that channel. The program will then run the print values routine and will loop on that test until the operator type "CTRL C". To change the selected channel or gain, the operator must type "CTRL G". The current switch register value will be reported. Bits 6 thru 9 select the gain and bits 0 thru 5 select the channel to be used. If an additional 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 report the number of MNCAD detected and will then give a channel table for the MNCAD (A/D) under test. If any test module is connected, the program will then ask which channels to test. The program will run through the logic test for the MNCAD, MNCAG and MNCTP, analog subtests, the noise test, the interchannel settling test, and on the second pass, the differential linearity and relative accuracy tests.

If 'L' is typed, the program will then size the number of MNCAD (A/D)'S and report the number of units found. It will then give a channel table for the current MNCAD under test. The program will then execute the logic tests for the MNCAD, MNCAG and MNCTP, 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 MNCAD-TA test module is connected, the program will ask the operator to change the "SINGLE ENDED-DIFFERENTIAL" test switch. The program will then ask the operator to depress the test module "EXTERNAL START" switch on the first pass. If a MNCAM-TA is connected, the operator should change the "SINGLE ENDED-DIFFERENTIAL" switch on the MNCAM-TA to verify that the channels do change configuration.

If 'N' is typed, the program will report the number of MNCAD detected and will then give a channel table for the MNCAD under test. The program will then ask for the "STARTING CHANNEL". The operator now inputs the desired channel and depresses the 'RETURN'. The program will now ask for the 'ENDING CHANNEL'. The operator now inputs the last channels to be tested. If only one channel is desired, depress 'RETURN' for this answer. The program will now run the noise test on the selected channels. If the channel is a MNCAG channel, the noise test is repeated at each different gain.

If 'S' is typed, the program will report the number of MNCAD detected and will then give a channel table for the MNCAD under test. The program will then ask for the two channels that are to be tested. It is important that the two channels are at opposite input values near maximum analog input (IE 0250 and 7540).

If 'D' is typed, the program will report the number of MNCAD detected and will then give a channel table for the MNCAD under test. The program will then ask for the 'STARTING ON CHANNEL'. The operator now inputs the desired channel and depresses the 'RETURN'. The program will now ask for the 'ENDING ON CHANNEL'. The operator now inputs the last channel to be tested. If only one channel is desired, depress 'RETURN' for this answer. The test requires that all channels to be run must have a 'FULL RANGE RAMP input.

If 'M' is typed, the program will request which channel will be used. The operator is now instructed to apply '+10 volts' to the channel input. The operator is then instructed to apply '-10 volts' to the channel input. The program will now report the results of the the test.

If 'F' is typed, the program will request which channel will be used. The operator is now instructed what position to set the front panel switches. No analog input values will not be checked, only the front panel switches and digital read-back logic.

If 'T' is typed, the program will request which channel will be used. The operator is now asked to verify the 'LED' state on the MNCAG-TA. The operator is now instructed what position to set the front panel and MNCAG-TA switches. The analog input values will be tested for all gains and modes.

If 'X' is typed, the program will request which channel will be used. The program will then ask for the 'GAIN OR TC TYPE' value to be loaded. Followed by the expected value ot be read back. The program will then use the values supplied to load and read the 'GAIN OR TC TYPE' registers.

If 'H' is typed, the program will tell the operator what position to set the front panel and test module switches. It will then ask about the presence of the test modules, clock and the type of console terminal. The program will then type the list of tests available.

If 'B' is typed, the program will ask for the new bus address of the MNCAD. After the new address has been selected, the new vector address is requested. Upon completion of the input, the program will reprompt the operator about the test to be run.

If 'G' is typed, the program will ask for the new switch register value. Upon completion of the new value, the program will re-prompt the operator about the tests to be run.

If 'I' is typed, the program will ask for the starting MNCTP channel. The program will ask if a gain-type is to be selected. The program will provide different voltage input to the MNCTP and monitor the output with the digital voltmeter. Upon completion, the program will repromt the operator about the starting channel.

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 (1210). The operator can 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 the end of a pass in which no errors were detected, the following typeout will occur:

'END PASS 12''

If any errors were detected, the following typeout will occur at the end of a pass:

'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.
> MNCAD 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:	Expected correct data
TOLERANCE:	The acceptable deviation from the nominal
ACTUAL:	Actual data
EXPECTED:	Expected correct data
TP TYPE:	Value of the MNCTP TYPE register

7.0 MISCELLANEOUS

7.1 Execution time

Execution time for each of the tests is:

Calibration:	5 conversions/min @110 baud
Print values:	64 conversions/8 seconds @ 110 baud
Wraparound test:	7 minutes first pass; 22 minutes for successive passes
Logic test:	30 seconds (MNCAD ONLY)
Auto test:	8 minutes first pass, 23 minutes for successive passes
Noise test:	20 seconds per selected channel
Differential Linearity	14 minutes
Settling test:	15 seconds
Front panel on MNCAG:	Operator intervention
Test MNCAG inputs:	Operator intervention
Common mode test:	Operator intervention

7.2 Status register and vector addresses

The program enables testing more than one MNCAD. The first MNCAD'S status register address must be in $BASE (1244=171000), its vector address must be in the low byte of $VECT1 (1240=400). 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 answering 'YES'' to the initial program starting question. The program will then display the results of the differential linearity and relative accuracy tests on the bit map terminal screen.

7.5 XXDP/ACT/APT

THE PROGRAM IS CHAINABLE UNDER XXDP AND ACT. The program is scriptable under APT. Only the logic tests will be run under XXDP, ACT and APT.

8.0 RESTRICTIONS

8.1 Testing

No external user connections to the MNCAD, MNCAM, MNCAG OR MNCTP
during program execution.

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 test of this program check to see if the MNCAD responds
to the expected address. If the MNCAD does not respond, a bus
error occurs and a error is reported to the operator.   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

MNCAD TESTING

These 30 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. If the MNCAD-TA (test module) is
connected, the operator is requested to change the position of
the switch on the MNCAD-TA.

MNCAG TESTING

When a MNCAG has been detected, these 5 logic subtests are run
sequentially after the MNCAD tests. Their purpose is to check
that each of the GAIN register bits can be loaded and properly
read-back. It also ensures that loading the GAIN bits of the
selected channel does not effect the condition of the GAIN bits
of another channel.

MNCTP TESTING

When a MNCTP has been detected, this logic subtest (consisting of
4 sections) is run sequentially after the MNCAD AND MNCAG tests.
The purpose is to check that each of the TC TYPE register can be
loaded and properly read-back. It also ensures that loading the
TC TYPE bits of the selected channel does not effect the
condition of another channel.

9.2 Calibration loop for MNCAD

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 converted analog value loop

The program collects 8 samples and then reports the average value
to the operator. This loop allows the operator to check the
converted values of each channel. The operator may also change
the gain of the MNCAG channels. The operator may also change the
GAIN OR TC TYPE of the MNCTP channels.

9.4 Differential linearity and relative accuracy

This test determines the width of each state to within 0.01 LSB.
The basic process consists of applyinga FULL SCALE ramp input and
creating a histogram buffer of converted values. The values in
the histogram buffer are then compared to a set of nominal limit
values.

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 2.7 standard deviation of the Gaussian curve.

9.7 Analog tests

These 8 subtests check the converted values of the selected
channels and their output.

9.8 Function test of the MNCAG front panel

This test enables the operator to verify proper operation of the
MNCAG front panel controls and digital read-back locig. The
program asks the operator to set the MNCAG front panel switches.
The program will then read the status and gain bits and compare
it to the expected value. Analog testing of the different gains
is not performed in this test.

9.9 Test MNCAG channels analog input value

This test is used to verify proper operation of the analog
control logic. The test requires that the operator set the
switches on the MNCAG-TA test module and the front panel
switches. The program will verify the converted value to an
expected value for that gain and mode settings. This test checks
all the gains and modes of the MNCAG front panel switches.

9.10 Gain or TC TYPE register loop

This loop enables the operator to read and write the GAIN or TC
TYPE register bits. The operator selects the channel, bits to be
loaded and the expected value to be read back. The program will
perform a read-write test on the channel and compare the results
to the expected. When an error is detected, the operator is
informed of the value read back.

9.11 Common mode rejection test

This test enables the operator to verify that by changing the
reference level of a channel does not change the NOISE value on
the channel. The operator is requested to supply an input
voltage near the maximum positive value. The program will then
sample the channel and then ask the operator to supply the
maximum negative value. The program will sample the channel and
compare the two values.

9.12 Incomming inspection test of the MNCTP

This test enables the CPU to verify correct operation of the
MNCTP. The tester module supplies a voltage input to the MNCTP.
The program will then use the DVM (digital voltage meter) to
verify proper output from the MNCTP module. If incorrect, the
program will report an error to the operator.

N 1

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41
CVMNAC.P11   10-JUL-81 14:30   TABLE OF CONTENTS   SEQ 0013

```
20                              .TITLE   CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC
(1)                             ;*COPYRIGHT (C) 1981
(1)                             ;*DIGITAL EQUIPMENT CORP.
(1)                             ;*MAYNARD, MASS. 01754
(1)                             ;*
(1)                             ;*
(1)                             ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
(1)                             ;*PACKAGE (MAINDEC-11-DZQAC-C4), 31 JULY 1980.
(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                             .SBTTL  TRAP CATCHER
 27
 28         000000              .=0
 29                             :*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ''.+2''
 30                             :*AND ''JSR PC,R0'' SEQUENCE TO CATCH ILLEGAL INTERRUPTS.
 31                             :*AND INTERRUPTS TO THE WRONG VECTOR.
 32                             :*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
 33                             :*VECTORS.
 41         000004              .=4
 42 000004  040354  000200      .WORD   IOTRD,200          ;HANDLE UNEXPECTED BUSS ERROR.
 43         000042              .=42
 44 000042  000000              .WORD   0                  ;XXDP CHAIN MODE FLAG/ADDRESS
 45         000100              .=100
 46 000100  000104  000340  000002   .WORD   104,340,2     ;'BEVENT'' HANDLER
 47         000174              .=174
 48 000174  000000              DISPREG: .WORD  0          ::SOFTWARE DISPLAY REGISTER.
 49 000176  000000              SWREG:   .WORD  0          ::SOFTWARE SWITCH REGISTER.
 50
 51 000200  000137  001666           JMP    BEGIN          ;START ADDRESS
 52 000204  000137  001674           JMP    BEG2           ;RESTART ADDRESS
 53 000210  000137  001702           JMP    BEGIN2         ;START ADDRESS FOR OPTION TESTER CONNECTED
 54 000214  000137  001712           JMP    BEGIN3         ;STARTING ADDRESS FOR INCOMING TESTING OF MNCTP'S
```

```
   56                                    .SBTTL  ACT11 HOOKS
  (1)
  (2)                                    ;;*****************************************************************
  (1)                                    ;HOOKS REQUIRED BY ACT11
  (1)            000220                       $SVPC=.                    ;SAVE PC
  (1)            000046                       .=46
  (1)  000046    040242                       $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)            000220                       .=$SVPC                    ;; RESTORE PC
   57            001000                    .=1000
   58                                    .SBTTL   APT PARAMETER BLOCK
  (1)
  (2)                                    ;;*****************************************************************
  (1)                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  (2)                                    ;;*****************************************************************
  (1)            001000                       .$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)
```

G 2

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 2-1
CVMNAC.P11   10-JUL-81 14:30   COMMON TAGS   SEQ 0019

```
   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
  (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
70                                    ;ITEM   1
71   001252  052213  053165  053614           EM1,DH1,DT1,DF1           ;MNCAD STATUS REG. ERROR
     001260  053760
72                                    ;ITEM   2
73   001262  052251  053315  053650           EM2,DH3,DT3,DF1           ;MNCAD FAILED TO INTERRUPT
     001270  053760
74                                    ;ITEM   3
75   001272  052311  053315  053650           EM3,DH3,DT3,DF1           ;MNCAD UNEXPECTED INTERRUPT
     001300  053760
76                                    ;ITEM   4
77   001302  052352  053231  053630           EM4,DH2,DT2,DF1           ;MNCAD ERROR ON A/D CHANNEL
     001310  053760
78                                    ;ITEM   5
79   001312  052413  053351  053662           EM5,DH5,DT5,DF1           ;EXISTING MNCAD NOW FAILS TO RESPOND
     001320  053760
80                                    ;ITEM   6
81   001322  052474  053375  053674           EM6,DH6,DT6,DF1           ;BUS ERROR ON SPECIFIED DEFAULT ADDRESS
     001330  053760
82                                    ;ITEM   7
83   001332  052572  053416  053704           EM7,DH7,DT7,DF1           ;INCORRECT I.D. VALUE
     001340  053760
84                                    ;ITEM   10
85   001342  052617  053165  053614           EM10,DH1,DT1,DF1          ;"MNCAG HOLD" SIGNAL IN ERROR
     001350  053760
86                                    ;ITEM   11
87   001352  052663  053454  053720           EM11,DH12,DT12,DF1        ;"INCORRECT" MNCAG (PREAMP) FRONT PANEL SWITCH POSITION
     001360  053760
88                                    ;ITEM   12
89   001362  052737  053454  053720           EM12,DH12,DT12,DF1        ;MNCAG GAIN REGISTER IN ERROR
     001370  053760
90                                    ;ITEM   13
91   001372  053005  053454  053720           EM13,DH12,DT12,DF1        ;MNCTP GAIN REGISTER IN ERROR
     001400  053760
92                                    ;ITEM   14
93   001402  053053  053454  053720           EM14,DH12,DT12,DF1        ;"GAIN OR TC TYPE" REGISTER IN ERROR
     001410  053760
94                                    ;ITEM   15
95   001412  053117  053520  053736           EM15,DH15,DT15,DF1        ;MNCTP ERROR ON THE MNCTP CHANNEL VALUE
     001420  053760
96   001422  000000                   ADTA:   0              ;MNCAD-TA INDICATOR
```

```
 97  001424  000000      AMTA:   0               ;MNCAM-TA INDICATOR
 98  001426  000000      AGTA:   0               ;MNCAG-TA INDICATOR
 99  001430  000000      TPTA:   0               ;MNCTP-TA INDICATOR
100  001432  000000      TPAB:   0               ;MNCTP AT AMBIENT TEMP.
101  001434  001000      BARFO:  BIT9    ;DELAY FACTOR FOR CPU, SO THE HELP MESSAGE WONT GET MESSED UP
102                                      ;AND OTHER TESTS
103
104  001436  171000      MNCADO: ABASE                   ;ADDRESS OF MNCAD #0
105  001440  000400              AVECT1                  ;VECTOR OF MNCAD #0
106  001442  171004              ABASE+4                 ;               #1
107  001444  000410              AVECT1+10               ;               #1
108  001446  171010              ABASE+10                ;               #2
109  001450  000460              AVECT1+60               ;               #2
110  001452  171014              ABASE+14                ;               #3
111  001454  000470              AVECT1+70               ;               #3
112
113                      .SBTTL          MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
114  001456  171000      STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
115  001460  171001      ADST1:  ABASE+1                 ;UPPER BYTE OF STATUS REG.
116  0C1462  171002      ADBUFF: ABASE+2                 ;ADDRESS OF A/D BUFFER
117  001464  000400      VECTOR: AVECT1                  ;VECTOR ADDRESS
118  001466  000402      VECTR1: AVECT1+2
119  001470  000404      VECTR2: AVECT1+4                ;ERROR VECTOR ADDRESS
120  001472  000406      VECTR3: AVECT1+6
121  001474  000000      BASECH: 0                       ;BASE CHANNEL
122  001476  000000      BASEND: 0                       ;END CHANNEL
123  001500  000060      KBVECT: 60
124  001502  171020      KWCSR:  171020                  ;NORMAL MNCKW ADDRESS
125  001504  171022      KWBPR:  171022                  ;MNCKW BUF REG.
126                      :       TESTER DEVICES
127  001506  170400      GSTREG: 170400                  ;KNOWN GOOD A/D CSR
128  001510  170402      GADBUF: 170402                  ;KNOWN GOOD A/D DBR
129  001512  000410      GVECT:  410                     ;KNOWN GOOD A/D VECTOR
130  001514  000412      GVECT1: 412                     ;GOOD A/D BR VECTOR
131  001516  170430      CLKCSR: 170430                  ;CLOCK CSR
132  001520  170432      CLKBPR: 170432                  ;CLOCK BPR
133  001522  167770      DRVCSR: 167770                  ;DRV11 CSR
134  001524  167772      DRVDOR: 167772                  ;DRV11 DOR
135  001526  167774      DRVDIR: 167774                  ;DRV11 DIR
136                      :       COMMON TAGS
137  001530  000000      WIDE:   0                       ;NO. OF WIDE STATES
138  001532  000000      NARROW: 0                       ;NO. OF NARROW STATES
139  001534  000000      FIRST:  0
140  001536  000000      SKIPST: 0                       ;NO. OF SKIPPED STATES
141  001540  000000      TEMP:   0                       ;WORK AREA
142  001542  000000      TEMP1:  0                       ;RESTART INDICATOR
143  001544  000000      CH1:    0                       ;FIRST CHANNEL
144  001546  000000      CH2:    0                       ;SECOND CHANNEL
145  001550  000000      NBEXT:  0                       ;NO. OF MNCAD'S TO BE TESTED
146  001552  000000      NMBEXT: 0                       ;NO. OF MNCAD'S TO BE TESTED
147  001554  000000      DUMMY:  0                       ;DUMMY CHANNEL
148  001556  000000      CHANL:  0                       ;CHANNEL VALUE
149  001560  000000      RMS:    0                       ;RMS NOISE VALUE
150  001562  000000      PEAK:   0                       ;PEAK NOISE VALUE
151  001564  000000      VTFLAG: 0                       ;BIT MAP TERMINAL FLAG
152  001566  000000      SPREAD: 0                       ;DEVIATION FROM THE NOMINAL
```

```
153   001570   000000                    DAC:     0                          ;SAR VALUE
154   001572   000000                    DELAY:   0                          ;TIME DELAY COUNTER
155   001574   000000                    EDGE:    0                          ;EDGE VALUE
156   001576   000000                    BITPNT:  0
157   001600   000000                    MIN:     0                          ;MIN VALUE
158   001602   000000                    WFTEST:  0                          ;TESTER AVAILABLE FLAG
159   001604   000000           .         KWAD:    0                          ;MNCKW AVAILABLE TO TEST CLOCK STARTS FLAG
160   001606   000000                    MAX:     0                          ;MAX VALUE
161   001610   000000                    PERCNT:  0                          ;PERCENT FOR SAR ROUTINE
162   001612   000000                    OUT:     0
163   001614   000000                    EVER:    0
164   001616   000000                    BADUNT:  0                          ;BAD UNIT MAP
165   001620   000001                    MASKNM:  1                          ;CURRENT UNIT MAP
166   001622   000000                    UNITBD:  0
167   001624   000000                    QUIET:   0                          ;SHORT RUN BECAUSE NO EXTRA TYPEOUT
168
169   001626                             UNEXP:
(1)   001626   012737  001642  001162              MOV     #1$,$ESCAPE       ;;ESCAPE TO 1$ ON ERROR
170   001634   005237  001103                      INC     $ERFLG
171   001640   104003                              ERROR   3
172   001642   005037  001162             1$:      CLR,    $ESCAPE           ;RETURN ESCAPE TO NORMAL
173   001646   000002                              RTI                       ;UNEXPECTED INTERRUPT
174   001650   022776  000001  000000     RETURN:  CMP     #1,a0(SP)         ;DOES IT RETURN TO A WAIT?
175   001656   001002                              BNE     RET2              ;NO
176   001660   062716  000002             RET1:    ADD     #2,(SP)           ;BUMP RETURN ADDRESS
177   001664   000002                      RET2:    RTI
178
179
180                                       .SBTTL          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
181   001666   005037  001602             BEGIN:   CLR     WFTEST
182   001672   000412                              BR      RBEG
183   001674   005237  001542             BEG2:    INC     TEMP1             ;SET RESTART FLAG
184   001700   000411                              BR      RBEG1
185   001702   012737  100000  001602     BEGIN2:  MOV     #BIT15,WFTEST     ;INDICATE TESTER IS CONNECTED
186   001710   000403                              BR      RBEG
187   001712   012737  040000  001602     BEGIN3:  MOV     #BIT14,WFTEST     ;INDICATE INCOMMING TEST FOR MNCTP
188   001720   005037  001542             RBEG:    CLR     TEMP1             ;CLEAR RESTART FLAG
189   001724   004737  037030             RBEG1:   JSR     PC,ARESET         ;GENERATE A CONTROLLED BUS RESET
190                                       .SBTTL   INITIALIZE THE COMMON TAGS
(1)                                       ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1)   001730   012706  001100                      MOV     #$CMTAG,R6        ;;FIRST LOCATION TO BE CLEARED
(1)   001734   005026                              CLR     (R6)+             ;;CLEAR MEMORY LOCATION
(1)   001736   022706  001140                      CMP     #SWR,R6 ;;DONE?
(1)   001742   001374                              BNE     .-6               ;;LOOP BACK IF NO
(1)   001744   012706  001100                      MOV     #STACK,SP         ;;SETUP THE STACK POINTER
(1)                                       ;;INITIALIZE A FEW VECTORS
(1)   001750   012737  055424  000020              MOV     #$SCOPE,a#IOTVEC  ;;IOT VECTOR FOR SCOPE ROUTINE
(1)   001756   012737  000340  000022              MOV     #340,a#IOTVEC+2 ;;LEVEL 7
(1)   001764   012737  055746  000030              MOV     #$ERROR,a#EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
(1)   001772   012737  000340  000032              MOV     #340,a#EMTVEC+2 ;;LEVEL 7
(1)   002000   012737  060060  000034              MOV     #$TRAP,a#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1)   002006   012737  000340  000036              MOV     #340,a#TRAPVEC+2;;LEVEL 7
(1)   002014   012737  056312  000024              MOV     #$PWRDN,a#PWRVEC ;;POWER FAILURE VECTOR
(1)   002022   012737  000340  000026              MOV     #340,a#PWRVEC+2 ;;LEVEL 7
(1)   002030   013737  040210  040202              MOV     $ENDCT,$EOPCT     ;;SETUP END-OF-PROGRAM COUNTER
(1)   002036   005037  001160                      CLR     $TIMES            ;;INITIALIZE NUMBER OF ITERATIONS
```

```
 (1)    002042  005037  001162                      CLR     $ESCAPE            ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 (1)    002046  112737  000001  001115              MOVB    #1,$ERMAX          ;;ALLOW ONE ERROR PER TEST
 (1)    002054  012737  002054  001106              MOV     #.,$LPADR          ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 (1)    002062  012737  002062  001110              MOV     #.,$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)    002070  013746  000004              MOV     @#ERRVEC,-(SP)     ;;SAVE ERROR VECTOR
 (2)    002074  012737  002130  000004      MOV     #64$,@#ERRVEC      ;;SET UP ERROR VECTOR
 (2)    002102  012737  177570  001140      MOV     #DSWR,SWR          ;;SETUP FOR A HARDWARE SWICH REGISTER
 (2)    002110  012737  177570  001142      MOV     #DDISP,DISPLAY     ;;AND A HARDWARE DISPLAY REGISTER
 (2)    002116  022777  177777  177014      CMP     #-1,@SWR           ;;TRY TO REFERENCE HARDWARE SwR
 (2)    002124  001012                      BNE     66$                ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 (2)                                                                   ;;AND  THE HARDWARE SWR IS NOT = -1
 (2)    002126  000403                      BR      65$                ;;BRANCH IF NO TIMEOUT
 (2)    002130  012716  002136      64$:    MOV     #65$,(SP)          ;;SET UP FOR TRAP RETURN
 (2)    002134  000002                      RTI
 (2)    002136  012737  000176  001140  65$: MOV    #SWREG,SWR         ;;POINT TO SOFTWARE SWR
 (2)    002144  012737  000174  001142      MOV     #DISPREG,DISPLAY
 (2)    002152  012637  000004      66$:    MOV     (SP)+,@#ERRVEC     ;;RESTORE ERROR VECTOR
 (1)
 (2)    002156  005037  001176              CLR     $PASS              ;;CLEAR PASS COUNT
 (2)    002162  132737  000200  001211      BITB    #APTSIZE,$ENVM     ;;TEST USER SIZE UNDER APT
 (2)    002170  001403                      BEQ     67$                ;;YES,USE NON-APT SWITCH
 (2)    002172  012737  001212  001140      MOV     #$SWREG,SWR        ;;NO,USE APT SWITCH REGISTER
 (2)    002200                      67$:
 191                                ;ROUTINE TO OVERLAY THE ''$TYPE'' ROUTINE
 192    002200  012737  005046  056530      MOV     #5046,$TYPE        ;CLR -(SP)
 193    002206  012737  012746  056532      MOV     #12746,$TYPE+2     ;MOV #$TYPE+12,-(SP)
 194    002214  012737  056542  056534      MOV     #$TYPE+12,$TYPE+4
 195    002222  012737  000002  056536      MOV     #RTI,$TYPE+6       ;RTI
 196    002230  004737  054040              JSR     PC,$TKINT          ;ENABLE TKB INTR.
 197    002234  005737  001542              TST     TEMP1              ;TEST IF RESTART
 198    002240  001005                      BNE     20$                ;BR IF YES
 199    002242  005737  000042              TST     @#42               ;TEST IF CHAIN MODE
 200    002246  001002                      BNE     20$                ;BR IF CHAIN MODE
 201    002250  104401  050731              TYPE    ,INITVT            ;INITILIZE THE TERMINAL
 202    002254                      20$:
 (1)                                .SBTTL  TYPE PROGRAM NAME
 (1)                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
 (1)    002254  005227  177777              INC     #-1                ;;FIRST TIME?
 (1)    002260  001051                      BNE     68$                ;;BRANCH IF NO
 (1)    002262  022737  040242  000042      CMP     #$ENDAD,@#42       ;;ACT-11?
 (1)    002270  001445                      BEQ     68$                ;;BRANCH IF YES
 (1)    002272  104401  002340              TYPE    ,69$               ;;TYPE ASCIZ STRING
 (2)                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 (2)    002276  005737  000042              TST     @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
 (2)    002302  001012                      BNE     70$                ;;BRANCH IF YES
 (2)    002304  123727  001210  000001      CMPB    $ENV,#1            ;;ARE WE RUNNING UNDER APT?
 (2)    002312  001406                      BEQ     70$                ;;BRANCH IF YES
 (2)    002314  023727  001140  000176      CMP     SWR,#SWREG         ;;SOFTWARE SWITCH REG SELECTED?
 (2)    002322  001005                      BNE     71$                ;;BRANCH IF NO
 (2)    002324  104407                      GTSWR                      ;;GET SOFT-SWR SETTINGS
 (2)    002326  000403                      BR      71$                ;;
 (2)    002330  112737  000C01  001134  70$: MOVB   #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
 (2)    002336                      71$:
 (1)    002336  000422                      BR      68$                ;;GET OVER THE ASCIZ
```

```
 (1)                                  ;;69$:  .ASCIZ   <CRLF>#CVMNA-C     MNCAD (A/D) DIAGNOSTIC#<CRLF>
 (1)  002404                          68$:
 203  002404  013746  000010                  MOV      @#RESVEC,-(SP)    ;SAVE RESERVED VECTOR
 204  002410  012737  002450  000010          MOV      #1$,RESVEC        ;SET UP ILLEGAL INST. TRAP
 205  002416  012700  000001                  MOV      #1,R0             ;SET R0 TO ONE
 206  002422  077001                          SOB      R0,.              ;TRY SOB INSTRUCTION
 207  002424  012737  077001  034602          MOV      #77001,DELAY1     ;SET UP FOR SOB
 208  002432  012737  077001  034716          MOV      #77001,DELAY2     ;
 209  002440  012737  077001  035032          MOV      #77001,DELAY3     ;
 210  002446  000412                          BR       2$
 211  002450  022626                  1$:     CMP      (SP)+,(SP)+       ;POP TWO WORDS OFF STACK
 212  002452  012737  104420  034602          MOV      #DELY,DELAY1      ;INSTRUCTION FAILED
 213  002460  012737  104420  034716          MOV      #DELY,DELAY2      ;
 214  002466  012737  104420  035032          MOV      #DELY,DELAY3      ;
 215  002474  012637  000010          2$:     MOV      (SP)+,@#RESVEC    ;RESTORE ERROR VECTOR
 216  002500  004737  033454          3$:     JSR      PC,FIXONE         ;INITIALIZE ADDRESSES
 217  002504  004737  037260                  JSR      PC,WFADJ          ;SET UP TOLLERANCES
 218  002510  105737  001134                  TSTB     $AUTOB            ;TEST IF CHAIN/APT
 219  002514  001402                          BEQ      4$
 220  002516  000137  020524                  JMP      BEGL              ;GO TO LOGIC TESTS
 221  002522  005737  001542          4$:     TST      TEMP1             ;TEST IF RESTART
 222  002526  001151                          BNE      MTEST1
 223  002530  005737  001602                  TST      WFTEST            ;CHECK IF TESTER CONNECTED ?
 224  002534  100014                          BPL      MTEST             ;BR IF NO TESTER
 225  002536  104401  044235                  TYPE     ,SDDIF            ;SET MNCAD-TA TO DIFF
 226  002542  104401  044353                  TYPE     ,SDMDIF           ;SET MNCAM-TA TO DIFF
 227  002546  005237  001422                  INC      ADTA              ;SET AD-TA AVAIL FLAG
 228  002552  005237  001424                  INC      AMTA              ;SET AM-TA AVAIL FLAG
 229  002556  005237  001426                  INC      AGTA              ;SET AG-TA AVAIL FLAG
 230  002562  000137  003046                  JMP      MTEST0            ;BYPASS NORMAL START-UP Q + A
 231
 232                                          .SBTTL   OPERATOR INPUT ABOUT TEST MODULES CONNECTED
 233  002566  104401  001165          MTEST:  TYPE     ,$CRLF
 234  002572  104401  044011                  TYPE     ,SADTST           ;TELL OPER. ABOUT MNCAD FRONT PANEL SW.
 235  002576  104401  044072                  TYPE     ,SAGTST           ;TELL OPER. ABOUT MNCAG FRONT PANEL SW.
 236  002602  104401  043672                  TYPE     ,YESNO            ;ASK FOR INPUT
 237  002606  032737  040000  001602          BIT      #BIT14,WFTEST     ;TEST IF 214 START
 238  002614  001407                          BEQ      MTESTA            ;BR IF NOT
 239  002616  004537  003002                  JSR      R5,ASKTA          ;ASK OPERATOR IF MNCTP TESTER MODULE IS CONNECTED ?
 240  002622  030225                          DWTSTP
 241  002624  001540                          TEMP
 242  002626  000402                          BR       MTESTA            ;BR IF NOT CONNECTED
 243  002630  000137  026466                  JMP      AMNCTP            ;JUMP AND RUN MNCTP TESTER
 244  002634  004537  003002          MTESTA: JSR      R5,ASKTA          ;ASK ABOUT MNCAD-TA
 245  002640  041457                          DWRFAD
 246  002642  001422                          ADTA
 247  002644  000402                          BR       1$                ;BR IF NONE
 248  002646  104401  044165                  TYPE     ,SDSE             ;TELL OPER. TO SET MNCAD-TA SWITCH TO SINGLE END
 249  002652  004537  003002          1$:     JSR      R5,ASKTA          ;ASK ABOUT MNCAM-TA
 250  002656  041533                          DWRFAM
 251  002660  001424                          AMTA
 252  002662  000402                          BR       2$                ;BR IF NONE
 253  002664  104401  044305                  TYPE     ,SDMSE            ;TELL OPER. TO SET MNCAM-TA SWITCH TO SINGLE END
 254  002670  004537  003002          2$:     JSR      R5,ASKTA          ;ASK ABOUT MNCAG-TA
 255  002674  041605                          DWRFAG
 256  002676  001426                          AGTA
```

```
257   002700   000406                              BR       3$             ;BR IF NONE
258   002702   104401   045036                     TYPE     ,TXTP2         ;TELL OPER. TO SET MNCAG-TA SWITCHES
259   002706   104401   044652                     TYPE     ,SVM           ;AND MODE SWITCHES TO VOLTAGE
260   002712   104401   001165                     TYPE     ,$CRLF
261   002716   005037   001432            3$:       CLR      TPAB
262   002722   004537   003002                      JSR      R5,ASKTA       ;ASK ABOUT MNCTP-TA
263   002726   041662                              DWRFTP
264   002730   001430                              TPTA
265   002732   000410                              BR       4$             ;BR IF NONE
266   002734   004537   003002                     JSR      R5,ASKTA       ;ASK IF MNCTP IS AT AMBIENT TEMP.
267   002740   041737                              ASKAMB
268   002742   001432                              TPAB                    ;INDICATOR
269   002744   000401                              BR       6$             ;BR IF NOT AT AMBIENT TEMP
270   002746   000402                              BR       4$             ;BR IF AT AMBIENT TEMP
271   002750   104401   042025            6$:       TYPE     ,NOTPOF        ;TELL OPER. THAT THE MNCTP OFFSET TEST
272                                                                        ;  WILL NOT BE RUN
273   002754   004537   003002            4$:       JSR      R5,ASKTA       ;ASK IF MNCKW IS IN SYSTEM
274   002760   042774                              SCLOCK
275   002762   001604                              KWAD
276   002764   000240                              NOP                     ;MUST LEAVE NOP HERE
277   002766   004537   003002                     JSR      R5,ASKTA       ;ASK IF VT55/VT105 TERMINAL IS CONNECTED
278   002772   042101                              DWRMAP
279   002774   001564                              VTFLAG
280   002776   000240                              NOP                     ;MUST LEAVE NOP HERE
281   003000   000422                              BR       MTEST0
282   003002   012537   003014            ASKTA:    MOV      (R5)+,10$      ;GET MESSAGE POINTER
283   003006   104401   001165                     TYPE     ,$CRLF         ;FRESH LINE
284   003012   104401                              TYPE                    ;ABOUT DWARF MODULE
285   003014   041457            10$:      DWRFAD
286   003016   104412                              RDLIN
287   003020   012600                              MOV      (SP)+,R0       ;GET INPUT
288   003022   005075   000000                     CLR      @(R5)          ;SET NO MNCXX-TA FLAG
289   003026   042710   000040                     BIC      #40,(R0)       ;ENSURE UPPER CASE
290   003032   122710   000131                     CMPB     #'Y,(R0)       ;TEST IF 1ST CHAR IS Y
291   003036   001001                              BNE      1$             ;BR IF NOT 'Y'
292   003040   005235                              INC      @(R5)+         ;SET MNCXX-TA CONNECTED FLAG
293   003042   005725            1$:       TST      (R5)+          ;BUMP EXIT
294   003044   000205                              RTS      R5             ;EXIT
295
296                                        .SBTTL   OPERATOR INPUT DECODER TO TEST CHARACTER
297
298   003046   104401   050763            MTEST0:   TYPE     ,PRIME1        ;TELL THE OPERATOR THE STORY
299   003052   004737   037030            MTEST1:   JSR      PC,ARESET      ;ISSUE A BUS RESET
300   003056   052777   000100   176060             BIS      #BIT6,@$TKS    ;ENABLE TKB INTR.
301   003064   005046                              CLR      -(SP)
302   003066   012746   003074                     MOV      #1$,-(SP)
303   003072   000002                              RTI                     ;LOWER PS
304   003074   005037   001176            1$:       CLR      $PASS          ;INIT
305   003100   005037   001112                     CLR      $ERTTL         ;         THINGS
306   003104   005037   001614                     CLR      EVER           ;
307   003110   005037   001624                     CLR      QUIET
308   003114   104401   052123                     TYPE     ,DOT           ;TYPE THE 'DOT'
309   003120   104412                              RDLIN
310   003122   012600                              MOV      (SP)+,R0       ;READ ANSWER
311   003124   142710   000040                     BICB     #40,(R0)       ;FORCE UPPER CASE
312   003130   121027   000101                     CMPB     (R0),#'A       ;IS IT A?
```

```
313  003134  001002                          BNE    2$           ;;NO, TRY C
314  003136  000137  020572                  JMP    BEGINA       ;GO TO AUTO TEST
315  003142  121027  000103          2$:     CMPB   (R0),#'C     ;IS IT C?
316  003146  001002                          BNE    3$           ;;NO, TRY P
317  003150  000137  015632                  JMP    BEGINC       ;GO TO CALIBRATION LOOP
318  003154  121027  000120          3$:     CMPB   (R0),#'P     ;IS IT P?
319  003160  001002                          BNE    4$           ;;NO, TRY L
320  003162  000137  020212                  JMP    BEGINP       ;GO TO TYPE/DISPLAY CONVERSIONS TEST
321  003166  121027  000114          4$:     CMPB   (R0),#'L     ;IS IT L?
322  003172  001002                          BNE    5$           ;;NO, TRY M
323  003174  000137  020524                  JMP    BEGL         ;GO TO LOGIC TESTS
324  003200  121027  000127          5$:     CMPB   (R0),#'W     ;IS IT W?
325  003204  001002                          BNE    6$           ;;NO,TRY AGAIN
326  003206  000137  020636                  JMP    BEGINW       ;GO TO WRAPAROUND TEST
327  003212  121027  000102          6$:     CMPB   (R0),#'B     ;IS IT B?
328  003216  001002                          BNE    7$           ;;NO TRY AGAIN
329  003220  000137  033252                  JMP    BASEXC       ;GO CHANGE BASE AND VECTOR ADDRESS
330  003224  121027  000110          7$:     CMPB   (R0),#'H     ;IF IT H?
331  003230  001002                          BNE    10$          ;;NO, TRY AGAIN
332  003232  000137  002566                  JMP    MTEST        ;YES, HELP THE OPERATOR
333  003236  121027  000107          10$:    CMPB   (R0),#'G     ;IS IT G
334  003242  001002                          BNE    11$          ;;NO, TRY AGAIN
335  003244  104407                           GTSWR
336  003246  000701                           BR     MTEST1
337  003250  121027  000126          11$:    CMPB   (R0),#'V     ;IS IT V?
338  003254  001004                          BNE    12$          ;;NO, TRY AGAIN
339  003256  005237  001564                  INC    VTFLAG       ;SET BIT MAP AVAILABLE FLAG + RUN WRAPAROUND
340  003262  000137  020636                  JMP    BEGINW       ;AND RUN WRAP TEST'S
341  003266  121027  000116          12$:    CMPB   (R0),#'N     ;IS IT N?
342  003272  001002                          BNE    13$          ;;NO, TRY AGAIN
343  003274  000137  020676                  JMP    BEGINN       ;RUN NOISE TESTS
344  003300  121027  000106          13$:    CMPB   (R0),#'F     ;IS IT F
345  003304  001002                          BNE    14$          ;;NO, TRY AGAIN
346  003306  000137  016106                  JMP    BEGINF       ;RUN SWITCH GAIN/PREAMP FRONT PANEL TEST
347  003312  121027  000124          14$:    CMPB   (R0),#'T     ;IT IT T?
348  003316  001002                          BNE    15$          ;;NO, TRY AGAIN
349  003320  000137  016400                  JMP    BEGINT       ;RUN TEST MODULE VERIFY TESTS
350  003324  121027  000104          15$:    CMPB   (R0),#'D     ;IS IT D?
351  003330  001002                          BNE    16$          ;;NO,TRY AGAIN
352  003332  000137  021362                  JMP    BEGIND       ;RUN DIFFERENTIAL AND RELAC. TEST ONLY
353  003336  121027  000115          16$:    CMPB   (R0),#'M     ;IS IT M?
354  003342  001002                          BNE    17$          ;;NO, TRY AGAIN
355  003344  000137  021022                  JMP    BEGINM       ;RUN COMMON MODE TESTS
356  003350  121027  000123          17$:    CMPB   (R0),#'S     ;IS IT S?
357  003354  001002                          BNE    20$          ;;NO, TRY AGAIN
358  003356  000137  021522                  JMP    BEGINS       ;RUN SETTLING TEST ONLY
359  003362  121027  000130          20$:    CMPB   (R0),#'X     ;IS IT X?
360  003366  001002                          BNE    21$          ;;NO, TRY AGAIN
361  003370  000137  020006                  JMP    BEGINX       ;RUN READ-WRITE GAIN BITS LOOP
362  003374  121027  000121          21$:    CMPB   (R0),#'Q     ;IS IT Q?
363  003400  001004                          BNE    22$          ;;NO, TRY AGAIN
364  003402  005237  001624                  INC    QUIET        ;SET QUIET MODE
365  003406  000137  020572                  JMP    BEGINA       ;RUN AUTO TESTS (LOGIC + WRAPAROUND)
366  003412  121027  000111          22$:    CMPB   (R0),#'I     ;IS IT I?
367  003416  001012                          BNE    77$          ;;NO, TRY AGAIN
368  003420  052737  040000  001602          BIS    #BIT14,WFTEST ;SET INCOMMING INSPECTION TEST FLAG
```

```
369  003426  004537  003002               JSR     R5,ASKTA        ;ASK ABOUT MNCTP TESTER MODULE
370  003432  030225                        DWTSTP                  ;TEXT POINTER FOR QUESTION
371  003434  001540                        TEMP
372  003436  000402                        BR      77$             ;BR IF ANSWER WAS NO
373  003440  000137  026466                JMP     AMNCTP          ;RUN THE TESTER MODULE
374  003444  104401  042156      77$:      TYPE    .QUEST
375  003450  000600                        BR      MTEST1          ;WAIT FOR CHARACTER
```

```
377                                          .SBTTL  DETERMINE THE NUMBER OF MNCAD'S ON THE SYSTEM
378   003452  013737  001244  001126  TESTAD: MOV    $BASE,$BDDAT     ;GET BASE ADDRESS
379   003460  005037  001202            CLR    $UNIT            ;CLR UNIT NUMBER
380   003464  012737  003540  000004     MOV    #2$,ERRVEC        ;LOAD RETURN ADDRESS
381   003472  005777  175430       1$:   TST    @$BDDAT          ;TEST IF ADDRESS EXISTS
382   003476  062737  000004  001126     ADD    #4,$BDDAT        ;UPDATE BUS ADDRESS
383   003504  005237  001202            INC    $UNIT            ;UPDATE UNIT COUNT
384   003510  005737  001210            TST    $ENV             ;TEST IF 'DO NOT SIZE''
385   003514  100424                     BMI    3$               ;BR IF NO SIZEING
386   003516  032777  010000  175414     BIT    #SW12,@SWR       ;TEST IF INHIBIT SIZING IS SET
387   003524  001020                     BNE    3$               ;BR IF SET
388   003526  022737  000004  001202     CMP    #4.,$UNIT        ;TEST IF MAX NUMBER
389   003534  001356                     BNE    1$               ;BR IF NOT
390   003536  000413                     BR     3$               ;BR IF MAX
391   003540  022626              2$:    CMP    (SP)+,(SP)+      ;RESTORE STACK
392   003542  005737  001202            TST    $UNIT            ;TEST IF ANY EXIST
393   003546  001007                     BNE    3$               ;BR IF ANY ARE THERE
394   003550  005737  000042            TST    @#42             ;TEST IF XXDP CHAIN MODE
395   003554  001004                     BNE    3$               ;BR IF YES
396   003556  104006                     ERROR  6                ;BASE ADDRESS CAUSED A BUS TRAP
397   003560  005726                     TST    (SP)+            ;POP 1 ARG.
398   003562  000137  040154            JMP    $EOP
399   003566  012737  040354  000004  3$: MOV   #IOTRD,ERRVEC
400   003574  012737  000200  000006     MOV    #200,ERRVEC+2
401   003602  005737  001614            TST    EVER             ;TEST IF # HAS BEEN REPORTED
402   003606  100427                     BMI    4$               ;IF YES BRANCH
403   003610  005737  001602            TST    WFTEST           ;TEST IF IN TESTER MODE
404   003614  100415                     BMI    7$               ;BR IF TESTER
405   003616  104401  050001            TYPE   ,FOUND1          ;TELL OPERATOR # OF MNCAD'S FOUND
406   003622  013746  001202            MOV    $UNIT,-(SP)      ;PUT # TO BE TYPED ON STACK
407   003626  104405                     TYPDS
408   003630  104401  050024            TYPE   ,FOUND2          ;FINISH MESSAGE
409   003634  005737  001202            TST    $UNIT            ;TEST IF ANY UNITS
410   003640  001003                     BNE    7$               ;ANY UNIT
411   003642  005726                     TST    (SP)+            ;POP 1 ARG. OFF STACK
412   003644  000137  040154            JMP    $EOP             ;REPORT EOP
413   003650  013737  001202  001614  7$: MOV   $UNIT,EVER       ;SAVE THE # OF MNCAD'S FOR LATER
414   003656  052737  100000  001614     BIS    #BIT15,EVER      ;SET 'REPORTED # FLAG'
415   003664  000410                     BR     5$
416   003666  123737  001614  001202  4$: CMPB  EVER,$UNIT       ;TEST IF ANY HAVE GONE AWAY
417   003674  001404                     BEQ    5$               ;BR IF ALL ARE STILL THERE
418   003676  113737  001614  001540     MOVB   EVER,TEMP        ;SAVE FOR ERROR REPORT
419   003704  104005                     ERROR  5                ;EXISTING DEVICE FAILED TO RESPOND
420   003706  005037  001202  5$:        CLR    $UNIT            ;RESET UNIT POINTER
421   003712  113737  001614  001552     MOVB   EVER,NMBEXT      ;GET # OF UNITS
422   003720  005337  001552            DEC    NMBEXT           ;ADJUST IT
423   003724  004737  033454            JSR    PC,FIXONE        ;FIX BUS AND VECTOR ADDRESSES
424   003730  005037  001616            CLR    BADUNT           ;RESET BAD UNIT INDICATOR
425   003734  005046                     CLR    -(SP)            ;LOWER PRIORITY LEVEL 0
426   003736  012746  003744            MOV    #6$,-(SP)
427   003742  000002                     RTI
428   003744  000207              6$:    RTS    PC               ;EXIT
```

```
430  003746                            BEGINL:
431                                     ;;*************************************************************
(3)                                     ;*TEST 1         +15 VOLT TEST    (TESTER ONLY)
(3)                                     ;;*************************************************************
(2)  003746  012737  003746  001106     TST1:   MOV    #TST1,$LPADR
(1)  003754  012737  000001  001160             MOV    #1,$TIMES         ;;DO 1 ITERATION
432  003762  012737  000001  001102             MOV    #$TN-1,$TSTNM     ;SET UP TEST NUMBER
433  003770  012737  003746  001110             MOV    #TST1,$LPERR
434  003776  005737  001602                      TST    WFTEST           ;IS PROGRAM RUNNING IN TESTER MODE?
435  004002  100100                              BPL    TST3             ;;NO, SKIP FIRST 2 TESTS
436  004004  005737  001176                      TST    $PASS            ;DO FIRST 2 TESTS ON 1ST PASS ONLY
437  004010  001075                              BNE    TST3
438  004012  005737  022134                      TST    WFAG             ;TEST IF RUNNING MNCAG ON TESTER
439  004016  001072                              BNE    TST3             ;;BR IF TESTING MNCAG
440  004020  005046                              CLR    -(SP)            ;RESET PRIORITY
441  004022  012746  004030                      MOV    #1$,-(SP)
442  004026  000002                              RTI
443  004030  104401  044504             1$:     TYPE   ,TP15             ;TYPE ''+15 = ''
444  004034  004537  036526                      JSR    R5,GCONVT        ;CONVERT CHANNEL 12
445  004040  000012                              12
446  004042  013703  001540                      MOV    TEMP,R3          ;GET TEMP
447  004046  004737  036642                      JSR    PC,CONV15        ;TYPE VOLTAGE
448  004052  104401  046502                      TYPE   ,SPACE           ;TYPE 4 SPACES
449  004056  004537  036462                      JSR    R5,COMPAR        ;TEST RESULTS
450  004062  006020                              6020
451  004064  037376                              V100D
452  004066  000403                              BR     2$               ;;ERROR
453  004070  104401  046620                      TYPE   ,OKMSG           ;TYPE 'OK'
454  004074  000406                              BR     TST2             ;;GOTO NEXT TEST
455  004076  104401  047505             2$:     TYPE   ,ERMSG           ;TYPE ''**ERROR**''
456  004102  004737  055704                      JSR    PC,WHICHV        ;INDICATE ERROR UNIT
457  004106  005237  001112                      INC    $ERTTL           ;UPDATE ERROR COUNT
458
459                                     ;;*************************************************************
(3)                                     ;*TEST 2         -15 VOLT TEST    (TESTER ONLY)
(3)                                     ;;*************************************************************
(2)  004112  000004                     TST2:   SCOPE
(1)  004114  012737  000001  001160             MOV    #1,$TIMES         ;;DO 1 ITERATION
460  004122  104401  044513                      TYPE   ,TM15             ;TYPE ''-15 = ''
461  004126  004537  036526                      JSR    R5,GCONVT        ;CONVERT CHANNEL 11
462  004132  000011                              11
463  004134  013703  001540                      MOV    TEMP,R3          ;GET TEMP
464  004140  004737  036642                      JSR    PC,CONV15        ;TYPE VOLTAGE
465  004144  104401  046502                      TYPE   ,SPACE           ;TYPE 4 SPACES
466  004150  004537  036462                      JSR    R5,COMPAR        ;TEST RESULTS
467  004154  001760                              1760
468  004156  037376                              V100D
469  004160  000403                              BR     1$               ;;ERROR
470  004162  104401  046620                      TYPE   ,OKMSG           ;TYPE 'OK'
471  004166  000406                              BR     TST3             ;;GOTO NEXT TEST
472  004170  104401  047505             1$:     TYPE   ,ERMSG           ;TYPE ''**ERROR**''
473  004174  004737  055704                      JSR    PC,WHICHV        ;INDICATE BAD UNIT
474  004200  005237  001112                      INC    $ERTTL           ;UPDATE ERROR COUNT
475
```

```
 477                                          ;;************************************************************
 (3)                                          ;*TEST 3         FLOAT A ONE THRU MULTIPLEXER BITS
 (3)                                          ;;************************************************************
 (2)   004204  000004              TST3:   SCOPE
 478   004206  012737  000003  001102        MOV     #$TN-1,$TSTNM             ;ENSURE PROPER TEST NUMBER
 479   004214  012737  000400  001124        MOV     #BIT8,$GDDAT             ;LOAD FIRST BIT
 480   004222  013777  001124  175226  2$:   MOV     $GDDAT,@STREG            ;LOAD EXPECTED VALUE
 481   004230  017737  175222  001126        MOV     @STREG,$BDDAT            ;READ STATUS REGISTER
 482   004236  042737  000002  001126        BIC     #BIT1,$BDDAT            ;CLEAR NXC BIT
 483   004244  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
 484   004252  001401                        BEQ     1$
 485   004254  104001                        ERROR   1                       ;FAILED TO LOAD + READ BIT
 486   004256  006337  001124        1$:   ASL     $GDDAT                  ;GET NEXT BIT
 487   004262  023727  001124  040000        CMP     $GDDAT,#BIT14           ;FINISHED?
 488   004270  001354                        BNE     2$              ;;NO,GO TO NEXT TEST
 489                                          ;;************************************************************
 (3)                                          ;*TEST 4         LOAD AND READ BACK ERROR I.E. BIT14
 (3)                                          ;;************************************************************
 (2)   004272  000004              TST4:   SCOPE
 490   004274  012737  040000  001124        MOV     #BIT14,$GDDAT
 491   004302  104415                        CHKIT
 492   004304  104001                        ERROR   1                       ;FAILED TO LOAD + READ ERROR I.E.
 493                                          ;;************************************************************
 (3)                                          ;*TEST 5         LOAD AND READ BACK INTERRUPT ENABLE BIT6
 (3)                                          ;;************************************************************
 (2)   004306  000004              TST5:   SCOPE
 494   004310  012777  001626  175146        MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
 495   004316  012777  000200  175142        MOV     #200,@VECTR1            ;LOAD BR LEVEL
 496   004324  012737  000100  001124        MOV     #BIT6,$GDDAT            ;LOAD EXPECTED DATA
 497   004332  104415                        CHKIT
 498   004334  104001                        ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE
 499                                          ;;************************************************************
 (3)                                          ;*TEST 6         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
 (3)                                          ;;************************************************************
 (2)   004336  000004              TST6:   SCOPE
 500   004340  012737  000040  001124        MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
 501   004346  104415                        CHKIT
 502   004350  104001                        ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB
 503                                          ;;************************************************************
 (3)                                          ;*TEST 7         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
 (3)                                          ;;************************************************************
 (2)   004352  000004              TST7:   SCOPE
 504   004354  012737  000020  001124        MOV     #BIT4,$GDDAT            ;LOAD EXPECTED DATA
 505   004362  104415                        CHKIT
 506   004364  104001                        ERROR   1                       ;FAILED TO LOAD + READ EXT. START ENABLE
 507                                          ;;************************************************************
 (3)                                          ;*TEST 10        LOAD AND READ BACK MAINT. TST BIT2
 (3)                                          ;;************************************************************
 (2)   004366  000004              TST10:  SCOPE
 508   004370  012737  000004  001124        MOV     #BIT2,$GDDAT            ;LOAD EXPECTED DATA
 509   004376  104415                        CHKIT
 510   004400  104001                        ERROR   1                       ;FAILED TO LOAD + READ BACK MAINT. TST
```

```
 512       ;;***********************************************************************
 (3)       ;*TEST 11         LOAD AND READ BACK ENABLE I.D. BIT3
 (3)       ;;***********************************************************************
 (2)  004402  000004        TST11:  SCOPE
 513  004404  012737  000010  001124          MOV     #BIT3,$GDDAT     ;LOAD EXPECTED DATA
 514  004412  104415                          CHKIT
 515  004414  104001                          ERROR   1               ;FAILED TO LOAD + READ ENABLE I.D. BIT
 516
 517       ;;***********************************************************************
 (3)       ;*TEST 12         LOAD AND READ BACK ERROR FLAG BIT15
 (3)       ;;***********************************************************************
 (2)  004416  000004        TST12:  SCOPE
 518  004420  012737  100000  001124          MOV     #BIT15,$GDDAT    ;LOAD EXPECTED DATA
 519  004426  104415                          CHKIT
 520  004430  104001                          ERROR   1               ;FAILED TO LOAD + READ ERROR FLAG
 521       ;;***********************************************************************
 (3)       ;*TEST 13         TEST INIT CLEARS BITS 2-6,8-14
 (3)       ;;***********************************************************************
 (2)  004432  000004        TST13:  SCOPE
 (1)  004434  012737  000300  001160          MOV     #300,$TIMES      ;;DO 300 ITERATIONS
 522  004442  005037  001124                  CLR     $GDDAT           ;LOAD EXPECTED DATA
 523  004446  012777  077574  175002  2$:     MOV     #77574,@STREG    ;SET STATUS REGISTER
 524  004454  000005                          RESET                    ;INITIALIZE
 525  004456  052777  000100  174460          BIS     #100,@$TKS       ;SET INTRPT. ENABLE
 526  004464  104414                          CHECK                    ;GO CHECK RESULTS
 527  004466  104001                          ERROR   1               ;RESET FAILED TO CLEAR AD ST. REG. BITS
 528       ;;***********************************************************************
 (3)       ;*TEST 14         BYTE TEST FOR THE LOW BYTE OF AD STATUS REG
 (3)       ;;***********************************************************************
 (2)  004470  000004        TST14:  SCOPE
 529  004472  012777  012440  174756          MOV     #12440,@STREG    ;LOAD A PATTERN INTO A/D STATUS REG.
 530  004500  112777  025010  174750          MOVB    #25010,@STREG    ;LOAD ONLY THE LOW BYTE WITH NEW VALUE
 531  004506  017737  174744  001126          MOV     @STREG,$BDDAT    ;READ NEW A/D STATUS VALUE
 532  004514  042737  000002  001126          BIC     #BIT1,$BDDAT     ;REMOVE NON-EXISTANT CHANNEL BIT
 533  004522  012737  012410  001124          MOV     #12410,$GDDAT    ;LOAD EXPECTED VALUE
 534  004530  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;COMPARE VALUES
 535  004536  001401                          BEQ     TST15            ;;BR IF SAME
 536  004540  104001                          ERROR   1               ;HIGH BYTE CHANGED IN ERROR
 537       ;;***********************************************************************
 (3)       ;*TEST 15         BYTE TEST FOR THE HIGH BYTE OF AD STATUS REG
 (3)       ;;***********************************************************************
 (2)  004542  000004        TST15:  SCOPE
 538  004544  012777  012440  174704          MOV     #12440,@STREG    ;LOAD A PATTERN INTO A/D STATUS REG.
 539  004552  112777  025052  174700          MOVB    #25052,@ADST1    ;LOAD ONLY THE HIGH BYTE WITH NEW VALUE
 540  004560  017737  174672  001126          MOV     @STREG,$BDDAT    ;READ NEW A/D STATUS VALUE
 541  004566  042737  000002  001126          BIC     #BIT1,$BDDAT     ;REMOVE NON-EXISTANT CHANNEL BIT
 542  004574  012737  025040  001124          MOV     #25040,$GDDAT    ;LOAD EXPECTED VALUE
 543  004602  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;COMPARE VALUES
 544  004610  001401                          BEQ     TST16            ;;BR IF SAME
 545  004612  104001                          ERROR   1               ;LOW BYTE CHANGED IN ERROR
```

```
  547                              ;:*******************************************************************
  (3)                             ;*TEST 16         TEST INIT CLEARS ERROR FLAG
  (3)                             ;:*******************************************************************
  (2)   004614  000004            TST16:  SCOPE
  (1)   004616  012737  000300  001160      MOV    #300,$TIMES     ;;DO 300 ITERATIONS
  548   004624  012777  100000  174624      MOV    #BIT15,@STREG   ;SET BIT 15
  549   004632  005037  001124            CLR    $GDDAT          ;CLEAR EXPECTED
  550   004636  000005                    RESET                  ;ISSUE INIT
  551   004640  052777  000100  174276      BIS    #100,@$TKS      ;SET INTRPT. EN. FOR KEYBOARD
  552   004646  104414                    CHECK
  553   004650  104001                    ERROR  1
  554                              ;:*******************************************************************
  (3)                             ;*TEST 17         TEST DONE FLAG SETS AND BIT0 CLEARS ON END OF CONV.
  (3)                             ;:*******************************************************************
  (2)   004652  000004            TST17:  SCOPE
  (1)   004654  012737  000100  001160      MOV    #100,$TIMES     ;;DO 100 ITERATIONS
  555   004662  005277  174570            INC    @STREG          ;START CONVERSION
  556   004666  012737  000200  001124      MOV    #BIT7,$GDDAT    ;LOAD EXPECTED
  557   004674  004737  022144            JSR    PC,STALL        ;DELAY
  558   004700  042777  100000  174550      BIC    #BIT15,@STREG   ;MASK OUT ERROR BIT
  559   004706  104414                    CHECK
  560   004710  104001                    ERROR  1               ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
  561   004712  017700  174544            MOV    @ADBUFF,R0      ;CLEAR DONE FLAG FOR ITERATIONS
  562                              ;:*******************************************************************
  (3)                             ;*TEST 20         TEST INIT CLEARS DONE FLAG
  (3)                             ;:*******************************************************************
  (2)   004716  000004            TST20:  SCOPE
  (1)   004720  012737  000300  001160      MOV    #300,$TIMES     ;;DO 300 ITERATIONS
  563   004726  005037  001124            CLR    $GDDAT          ;CLEAR EXPECTED
  564   004732  005277  174520            INC    @STREG          ;START CONVERSION
  565   004736  105777  174514    2$:      TSTB   @STREG
  566   004742  100375                    BPL    2$
  567   004744  000005                    RESET
  568   004746  052777  000100  174170      BIS    #BIT6,@$TKS     ;ENABLE INTR.
  569   004754  104414                    CHECK
  570   004756  104001                    ERROR  1               ;DONE FLAG FAILED TO CLEAR
  571
  572                              ;:*******************************************************************
  (3)                             ;*TEST 21         TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
  (3)                             ;:*******************************************************************
  (2)   004760  000004            TST21:  SCOPE
  573   004762  005037  001124            CLR    $GDDAT          ;CLEAR EXPECTED
  574   004766  005277  174464            INC    @STREG          ;SET A/D START CONVERSION BIT
  575   004772  105777  174460    1$:      TSTB   @STREG          ;WAIT FOR FLAG
  576   004776  100375                    BPL    1$
  577   005000  017700  174456            MOV    @ADBUFF,R0      ;READ CONVERTED VALUE
  578   005004  104414                    CHECK
  579   005006  104001                    ERROR  1               ;DONE FLAG FAILED TO CLEAR
```

I 3

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 8
CVMNAC.P11   10-JUL-81 14:30   T22   TEST ALL '0'S RESULTS USING MAINT. ADTST. BIT   SEQ 0034

```
 581                                    ;;********************************************************
 (3)                                    ;*TEST 22          TEST ALL '0'S RESULTS USING MAINT. ADTST. BIT
 (3)                                    ;;********************************************************
 (2)   005010  000004             TST22:  SCOPE
 582   005012  005037  001124             CLR     $GDDAT               ;CLEAR EXPECTED VALUE
 583   005016  005037  001556             CLR     CHANL                ;SET CHANL = 0
 584   005022  005037  001566             CLR     SPREAD               ;SET SPREAD = 0
 585   005026  012777  000005   174422     MOV     #5,@STREG            ;CONVERT EVEN CHANNEL WITH MAINT. BIT SET
 586   005034  105777  174416     1$:      TSTB    @STREG               ;WAIT FOR DONE
 587   005040  100375                      BPL     1$
 588   005042  017737  174414   001126     MOV     @ADBUFF,$BDDAT       ;RESULTS TO BDDAT FOR CHECKING
 589   005050  001401                      BEQ     TST23                ;;GOTO NEXT TEST
 590   005052  104004                      ERROR   4                    ;DID NOT GET ALL '0'S RSULT WITH MAINT. ADTST
 591                                    ;;********************************************************
 (3)                                    ;*TEST 23          TEST ALL '1'S RESULT USING MAINT. ADTST. BIT
 (3)                                    ;;********************************************************
 (2)   005054  000004             TST23:  SCOPE
 592   005056  012737  007777   001124     MOV     #7777,$GDDAT         ;EXPECT ALL '1'S RESULT
 593   005064  012737  000001   001556     MOV     #1,CHANL             ;SET CHANL = 1
 594   0C5072  005037  001566             CLR     SPREAD               ;SET SPREAD = 0
 595   005076  012777  000405   174352     MOV     #405,@STREG          ;CONVERT ODD CHANNEL WITH MAINT. BIT SET
 596   005104  105777  174346     1$:      TSTB    @STREG               ;WAIT FOR DONE
 597   005110  100375                      BPL     1$
 598   005112  017737  174344   001126     MOV     @ADBUFF,$BDDAT       ;RESULTS TO BDDAT FOR CHECKING
 599   005120  023737  001124   001126     CMP     $GDDAT,$BDDAT        ;EQUAL?
 600   005126  001401                      BEQ     TST24                ;;GOTO NEXT TEST
 601   005130  104004                      ERROR   4                    ;DID NOT GET ALL '1'S RESULT WITH MAINT. ADTST
 602                                    ;;********************************************************
 (3)                                    ;*TEST 24          GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
 (3)                                    ;;********************************************************
 (2)   005132  000004             TST24:  SCOPE
 (1)   005134  012737  000100   001160     MOV     #100,$TIMES          ;;DO 100 ITERATIONS
 603   005142  012737  005150   001106     MOV     #10$,$LPADR          ;LOAD RETURN ADDRESS
 604   005150  042777  000100   173766 10$: BIC     #BIT6,@$TKS          ;REMOVE TKB INTERRUPT
 605   005156  005046                      CLR     -(SP)                ;RESET PRIORITY
 606   005160  012746  005166             MOV     #1$,-(SP)
 607   005164  000002                      RTI
 608   005166  004737  034132     1$:      JSR     PC,SETINT            ;LOAD VECTOR AREA WITH TRAP CATCHER
 609   005172  012777  005254   174264     MOV     #3$,@VECTOR          ;INTERRUPT VECTOR ADDRESS
 610   005200  012777  000200   174260     MOV     #200,@VECTR1         ;SET UP NEW PSW
 611   005206  012777  000101   174242     MOV     #BIT6!BIT0,@STREG    ;SET INTERRUPT ENABLE BIT + START CONVERSION
 612   005214  105777  174236     2$:      TSTB    @STREG               ;WAIT FOR DONE
 613   005220  100375                      BPL     2$                   ;FLAG TO SET
 614   005222  017737  174230   001126     MOV     @STREG,$BDDAT        ;READ STATUS REGISTER
 615   005230  005077  174222             CLR     @STREG               ;ENSURE INTR. ENABLE IS CLEARED
 616   005234  017737  174222   001124     MOV     @ADBUFF,$GDDAT       ;READ TO CLEAR DONE FLAG
 617   005242  012737  000300   001124     MOV     #BIT7!BIT6,$GDDAT    ;LOAD EXPECTED GOOD DATA
 618   005250  104002                      ERROR   2                    ;FAILED TO INTERRUPT ON DONE
 619   005252  000401                      BR      4$                   ;;BRANCH TO NEXT TEST
 620   005254  022626             3$:      CMP     (SP)+,(SP)+          ;RESET STACK POINTER
 621   005256  013777  001466   174200 4$: MOV     VECTR1,@VECTOR       ;SET UP FOR UNEXPECTED INTERRUPT
 622   005264  012777  004700   174174     MOV     #4700,@VECTR1
 623   005272  005046                      CLR     -(SP)                ;CLEAR PSW
 624   005274  012746  005302             MOV     #5$,-(SP)
 625   005300  000002                      RTI
 626   005302  005077  174150     5$:      CLR     @STREG
```

J 3

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 8-1
CVMNAC.P11   10-JUL-81 14:30   T24   GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION   SEQ 0035

```
  627  005306  005777  174150                    TST     @ADBUFF         ;CLEAR DONE BIT
  628
  629                                     ;;************************************************************
  (3)                                     ;*TEST 25        TEST INTERRUPT OCCURS WHEN ERROR AND I.E.E. IS SET
  (3)                                     ;;************************************************************
  (2)  005312  000004             TST25:   SCOPE
  (1)  005314  012737  000100  001160     MOV     #100.$TIMES     ;;DO 100 ITERATIONS
  630  005322  012737  005330  001106     MOV     #10$,$LPADR     ;LOAD RETURN ADDRESS
  631  005330  042777  000100  173606  10$: BIC     #BIT6,@$TKS     ;REMOVE TKB INTERRUPT
  632  005336  005046                      CLR     -(SP)           ;LOWER PRIORITY
  633  005340  012746  005346              MOV     #1$,-(SP)
  634  005344  000002                      RTI
  635  005346  004737  034132      1$:     JSR     PC,SETINT       ;LOAD VECTOR AREA WITH TRAP CATCHER
  636  005352  012777  005424  174110      MOV     #2$,@VECTR2     ;SETUP VECTOR ADDRESS
  637  005360  012777  000200  174104      MOV     #200,@VECTR3    ;SET UP NEW PSW
  638  005366  012777  140000  174062      MOV     #BIT15!BIT14,@STREG     ;CAUSE AN INTERRUPT
  639  005374  017737  174056  001126      MOV     @STREG,$BDDAT   ;BAD DATA
  640  005402  012737  140000  001124      MOV     #BIT15!BIT14,$GDDAT     ;GOOD DATA
  641  005410  005077  174042              CLR     @STREG          ;CLEAR STATUS
  642  005414  005777  174042              TST     @ADBUFF         ;AND CLEAR DONE
  643  005420  104002                      ERROR   2               ;'ERROR' BIT FAILED TO GENERATE AN INTERRUPT
  644  005422  000401                      BR      3$
  645  005424  022626              2$:     CMP     (SP)+,(SP)+     ;POP STACK
  646  005426  005077  174024      3$:     CLR     @STREG          ;CLEAR STATUS REG.
  647  005432  005777  174024              TST     @ADBUFF         ;FALSE READ TO CLEAR DONE
  648  005436  013777  001472  174024      MOV     VECTR3,@VECTR2  ;RESET VECTOR
  649  005444  012777  004700  174020      MOV     #4700,@VECTR3   ;
  650  005452  005046                      CLR     -(SP)           ;RESET PRIORITY
  651  005454  012746  005462              MOV     #4$,-(SP)
  652  005460  000002                      RTI
  653  005462  005077  173770      4$:     CLR     @STREG
  654                                     ;;************************************************************
  (3)                                     ;*TEST 26        TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
  (3)                                     ;;************************************************************
  (2)  005466  000004             TST26:   SCOPE
  655  005470  012777  000001  173760      MOV     #BIT0,@STREG    ;START CONVERSION
  656  005476  052777  000100  173440      BIS     #BIT6,@$TKS     ;ENABLE TKB INTERRUPT
  657  005504  105777  173746      1$:     TSTB    @STREG          ;WAIT FOR
  658  005510  100375                      BPL     1$
  659  005512  012737  100200  001124  2$: MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
  660  005520  012777  000001  173730      MOV     #BIT0,@STREG    ;START 2ND CONVERSION
  661  005526  004737  022144              JSR     PC,STALL        ;DELAY
  662  005532  104414              4$:     CHECK
  663  005534  104001                      ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
  664                                                              ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
  665  005536  017700  173720              MOV     @ADBUFF,R0      ;CLEAR DONE FLAG
```

```
667                                     ;;*****************************************************************
(3)                                     ;*TEST 27          TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
(3)                                     ;;*****************************************************************
(2)   005542   000004                   TST27:  SCOPE
668   005544   012737  100000  001124           MOV     #BIT15,$GDDAT    ;LOAD EXPECTED DATA
669   005552   012777  000001  173676           MOV     #BIT0,@STREG     ;START CONVERSION
670   005560   112777  000001  173670           MOVB    #BIT0,@STREG     ;START NEXT CONVERSION
671   005566   112777  000001  173662           MOVB    #BIT0,@STREG     ;ONCE AGAIN IN CASE REFRESH INTERVENED
672   005574   017737  173656  001126           MOV     @STREG,$BDDAT    ;READ STATUS REGISTER
673   005602   042737  077777  001126           BIC     #77777,$BDDAT    ;MASK OUT BIT 15
674   005610   023737  001124  001126           CMP     $GDDAT,$BDDAT    ;COMPARE RESULTS
675   005616   001401                           BEQ     1$               ;;BRANCH OVER ERROR
676   005620   104001                           ERROR   1                ;ERROR FLAG NOT SET WHEN 2ND
677                                                                       ;CONVERT BEGINS BEFORE FIRST DONE
678   005622   105777  173630           1$:     TSTB    @STREG           ;WAIT FOR DONE
679   005626   100375                           BPL     1$               ;WAIT
680   005630   017700  173626                   MOV     @ADBUFF,R0
681   005634   005077  173616                   CLR     @STREG           ;CLEAR STATUS REGISTER
682                                     ;;*****************************************************************
(3)                                     ;*TEST 30          TEST CHANNELS 0-7 FOR SINGLE ENDED
(3)                                     ;;*****************************************************************
(2)   005640   000004                   TST30:  SCOPE
683   005642   005037  001124                   CLR     $GDDAT
684   005646   012777  000010  173602           MOV     #BIT3,@STREG     ;ENABLE PREAMP STATUS
685   005654   005277  173576           1$:     INC     @STREG           ;START A CONVERSION
686   005660   105777  173572           2$:     TSTB    @STREG           ;IS CONVERSION DONE?
687   005664   100375                           BPL     2$               ;NO, WAIT TILL IT IS DONE
688   005666   017737  173570  001126           MOV     @ADBUFF,$BDDAT   ;GET PREAMP STATUS
689   005674   042737  007777  001126           BIC     #7777,$BDDAT     ;MASK OUT CONVERTED VALUE
690   005702   001401                           BEQ     3$               ;;SKIP OVER ERROR IF ZERO
691   005704   104001                           ERROR   1                ;CHANNEL 0-7 CANNOT EVER BE DIFFERENTIAL
692   005706   062777  000400  173542   3$:     ADD     #BIT8,@STREG     ;INCREMENT CHANNEL TO BE TESTED
693   005714   032777  004000  173534           BIT     #BIT11,@STREG    ;IS IT DONE?
694   005722   001754                           BEQ     1$               ;;NO
695                                     ;;*****************************************************************
(3)                                     ;*TEST 31          TEST CLOCK OVERFLOW STARTS A/D  (TESTER ONLY)
(3)                                     ;;*****************************************************************
(2)   005724   000004                   TST31:  SCOPE
696   005726   005737  001602                   TST     WFTEST           ;RUNNING ON TESTER ?
697   005732   100020                           BPL     2$               ;;NO, GO TO NEXT TEST
698   005734   012737  000240  001124           MOV     #BIT7!BIT5,$GDDAT     ;SET UP EXPECTED RESULT
699   005742   013777  001124  173506           MOV     $GDDAT,@STREG    ;ENABLE CLOCK OVERFLOW START
700   005750   012777  177776  173542           MOV     #177776,@CLKBPR  ;SET CLOCK NEAR OVERFLOW
701   005756   012777  000011  173532           MOV     #11,@CLKCSR      ;START CLOCK AT LINE RATE
702   005764   004737  022144                   JSR     PC,STALL         ;DELAY
703   005770   104414                           CHECK                    ;CHECK RESULT
704   005772   104001                           ERROR   1                ;DONE FLAG FAILED TO SET
705   005774   005777  173462           2$:     TST     @ADBUFF          ;CLEAR DONE FLAG
706   006000   005077  173452                   CLR     @STREG           ;INHIBIT CLOCK OVERFLOW START
```

L 3

```
708                                      ;;**********************************************************
(3)                                      ;*TEST 32          TEST CLOCK OVERFLOW STARTS A/D  (IF MNCKW IS AVAILABLE)
(3)                                      ;;**********************************************************
(2)   006004  000004                     TST32:  SCOPE
709   006006  005737  001604                     TST     KWAD                  ;TEST IF OPERATOR SAID MNCKW WAS AVAILABLE
710   006012  001424                             BEQ     TST33                 ;;BR IF ANSWER WAS NO
711   006014  012737  000240  001124             MOV     #BIT7!BIT5,$GDDAT         ;LOAD EXPECTED
712   006022  013777  001124  173426             MOV     $GDDAT,@STREG  ;LOAD STATUS REG.
713   006030  012777  177777  173446             MOV     #177777,@KWBPR  ;LOAD PRESET REGISTER
714   006036  012777  000011  173436             MOV     #11,@KWCSR    ;ENABLE CLOCK
715   006044  004737  022144                      JSR     PC,STALL      ;DELAY
716   006050  104414                             CHECK                  ;CHECK RESULTS
717   006052  104001                             ERROR   1              ;DONE FLAG FAILED TO SET WITH CLOCK STARTS
718   006054  005777  173402                      TST     @ADBUFF       ;CLEAR DONE FLAG
719   006060  005077  173372                      CLR     @STREG        ;INHIBIT CLOCK START
720
721                                      ;;**********************************************************
(3)                                      ;*TEST 33          TEST MNCAD S.E.- DIFF MODE STATUS BIT    (TESTER ONLY)
(3)                                      ;;**********************************************************
(2)   006064  000004                     TST33:  SCOPE
(1)   006066  012737  000100  001160             MOV     #100,$TIMES    ;;DO 100 ITERATIONS
722   006074  005737  022130                      TST     WFAD           ;TEST IF TESTING MNCAD
723   006100  001412                             BEQ     TST34          ;;BR IF NOT
724   006102  004537  015364                      JSR     R5,TSTSDF      ;GO TO SUBROUTINE AND DO THE TESTING
725   006106  000200                             BIT7                   ;1ST IN DIFFERENTIAL MODE
726   006110  020000                             20000                  ;EXPECTED DATA
727   006112  004010                             4010                   ;ON CHANNEL 10
728   006114  004537  015364                      JSR     R5,TSTSDF      ;REPEAT
729   006120  000000                             0                      ;THEN IN SINGLE ENDED MODE
730   006122  000000                             0                      ;EXPECTED DATA
731   006124  004010                             4010                   ;ON CHANNEL 10
732
733                                      ;;**********************************************************
(3)                                      ;*TEST 34          TEST MNCAM S.E.- DIFF MODE STATUS BIT    (TESTER ONLY)
(3)                                      ;;**********************************************************
(2)   006126  000004                     TST34:  SCOPE
(1)   006130  012737  000100  001160             MOV     #100,$TIMES    ;;DO 100 ITERATIONS
734   006136  005737  022132                      TST     WFAM           ;TEST IF TESTING MNCAM
735   006142  001412                             BEQ     TST35          ;;BR IF NOT
736   006144  004537  015364                      JSR     R5,TSTSDF      ;GO TO SUBROUTINE AND DO THE TESTING
737   006150  000200                             BIT7                   ;1ST IN DIFFERENTIAL MODE
738   006152  020000                             20000                  ;EXPECTED DATA
739   006154  006010                             6010                   ;ON CHANNEL 14 <1ST MNCAM ON TESTER IF DIFF.>
740   006156  004537  015364                      JSR     R5,TSTSDF      ;REPEAT
741   006162  000000                             0                      ;THEN IN SINGLE ENDED MODE
742   006164  000000                             0                      ;EXPECTED DATA
743   006166  010010                             10010                  ;ON CHANNEL 20 <1ST MNCAM ON TESTER IF S.E.>
```

```
 745                                      ;:*********************************************************
 (3)                                      ;*TEST 35        TEST MNCAD S.E.- DIFF MODE STATUS BIT   (MNCAD-TA ONLY)
 (3)                                      ;:*********************************************************
 (2)   006170  000004           TST35:    SCOPE
 (1)   006172  012737  000001  001160     MOV    #1,$TIMES         ;;DO 1 ITERATION
 746   006200  005737  001602             TST    WFTEST            ;RUNNING ON TESTER ?
 747   006204  100443                      BMI    TST36             ;;BR IF YES
 748   006206  005737  001422             TST    ADTA              ;IS MNCAD-TA AVAILABLE ?
 749   006212  001440                      BEQ    TST36             ;;BR IF NO
 750   006214  013700  060166             MOV    CHTABL+10,R0      ;GET CHANNEL #10 TYPE
 751   006220  042700  177700             BIC    #177700,R0        ;MASK OFF OTHER BITS
 752   006224  022700  000003             CMP    #3,R0             ;TEST IF MNCAG
 753   006230  001431                      BEQ    TST36             ;;BR IF AG CHANNEL-CANT CHANGE SE/DIF IF MNCAG IS CH10
 754   006232  005737  001176             TST    $PASS             ;TEST IF FIRST PASS
 755   006236  001026                      BNE    TST36             ;;BR IF NOT
 756   006240  104401  044235             TYPE   .SDDIF            ;TELL OPERATOR TO SET MNCAD-TA TO DIFFERENTIAL
 757   006244  104401  047151             TYPE   .CRWR             ;TELL OPERATOR TO DEPRESS ''RETURN''
 758   006250  104412                      RDLIN                    ;WAIT FOR ''CR''
 759   006252  005726                      TST    (SP)+             ;CLEAN STACK
 760   006254  004537  015364             JSR    R5,TSTSDF         ;GO TO SUBROUTINE TO DO THE TESTING
 761   006260  000000                      0                        ;NA
 762   006262  020000                      20000                    ;EXPECTED DATA
 763   006264  004010                      4010                     ;ON CHANNEL 10
 764   006266  104401  044165             TYPE   .SDSE             ;TELL OPERATOR TO SET MNCAD-TA TO S.E.
 765   006272  104401  047151             TYPE   .CRWR             ;TELL OPERATOR TO DEPRESS ''RETURN''
 766   006276  104412                      RDLIN
 767   006300  005726                      TST    (SP)+             ;CLEAN STACK
 768   006302  004537  015364             JSR    R5,TSTSDF         ;TEST THE MODE BIT
 769   006306  000000                      0                        ;NA
 770   006310  000000                      0                        ;EXPECTED DATA
 771   006312  004010                      4010                     ;ON CHANNEL 10
```

N 3

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063)   10-JUL-81 14:41   PAGE 12
CVMNAC.P11   10-JUL-81 14:30   T35   TEST MNCAD S.E.- DIFF MODE STATUS BIT   (MNCAD-TA ONLY)   SEQ 0039

```
773
774                          ;:************************************************************************
(3)                          ;*TEST 36          TEST EXTERNAL START STARTS A/D   (MNCAD-TA OR TESTER)
(3)                          ;:************************************************************************
(2)  006314  000004          TST36:  SCOPE
(1)  006316  012737  000001  001160   MOV    #1,$TIMES              ;;DO 1 ITERATION
775  006324  005737  022134          TST    WFAG                   ;TEST IF TESTING MNCAG ON TESTER
776  006330  001067                  BNE    TST37                  ;;BR IF YES
777  006332  013700  060166          MOV    CHTABL+10,R0           ;GET CHANNEL 10 TYPE
778  006336  042700  177700          BIC    #177700,R0             ;MASK OFF OTHER BITS
779  006342  022700  000003          CMP    #3,R0                  ;TEST IF CH10 IS A MNCAG CHANNEL
780  006346  001460                  BEQ    TST37                  ;;BR IF IT IS A MNCAG
781  006350  005737  001176          TST    $PASS                  ;TEST IF FIRST PASS
782  006354  001055                  BNE    TST37                  ;;BR IF NOT FIRST PASS
783  006356  012737  000220  001124   MOV   #BIT7!BIT4,$GDDAT       ;SET UP EXPECTED RESULT
784  006364  013777  001124  173064   MOV   $GDDAT,@STREG          ;ENABLE EXTERNAL START
785  006372  005737  001602          TST    WFTEST                 ;RUNNING IN TESTER MODE?
786  006376  100011                  BPL    2$                     ;;NO
787  006400  052777  000400  173116   BIS   #BIT8,@DRVDOR          ;GENERATE EXTERNAL START
788  006406  042777  000400  173110   BIC   #BIT8,@DRVDOR          ;RESET BIT
789  006414  004737  022144          JSR    PC,STALL               ;DELAY
790  006420  000425                  BR     3$                     ;;TEST RESULTS
791  006422  004737  022156   2$:    JSR    PC,AFIRST              ;TEST IF FIRST PASS
792  006426  000424                  BR     4$                     ;;BR IF NOT FIRST PASS
793  006430  005737  001422          TST    ADTA                   ;IF MNCAD-TA AVAILABLE ?
794  006434  001421                  BEQ    4$                     ;;BR IF NO
795  006436  104401  044421          TYPE   .EXTST                 ;TYPE MESSAGE ABOUT EXT. START
796  006442  004737  055712          JSR    PC,WHICHU              ;DETERMINE UNIT #
797  006446  013746  001622          MOV    UNITBD,-(SP)           ;;SAVE UNITBD FOR TYPEOUT
(1)  006452  104403                  TYPOS                         ;;GO TYPE--OCTAL ASCII
(1)  006454  001                     .BYTE  1                      ;;TYPE 1 DIGIT(S)
(1)  006455  000                     .BYTE  0                      ;;SUPPRESS LEADING ZEROS
798  006456  104401  047151          TYPE   .CRWR                  ;TYPE 'TYPE CR WHEN READY'
799  006462  104412                  RDLIN                         ;WAIT FOR CR
800  006464  005726                  TST    (SP)+                  ;POP WORD OFF STACK
801  006466  042777  100000  172762   BIC   #BIT15,@STREG          ;CLEAR A/D ERROR
802  006474  104414   3$:    CHECK                                 ;CHECK RESULT
803  006476  104001                  ERROR  1                      ;DONE FLAG FAILED TO SET
804  006500  005777  172756   4$:    TST    @ADBUFF                ;CLEAR DONE FLAG
805  006504  005077  172746          CLR    @STREG                 ;INHIBIT EXTERNAL START
806
844
```

```
  846                              ;;****************************************************************
  (3)                             ;*TEST 37        VERIFY 'HOLD' FROM MNCAG CHANNEL 10 (TESTER ONLY)
  (3)                             ;;****************************************************************
  (2)   006510  000004           TST37:  SCOPE
  (1)   006512  012737  000100  001160    MOV     #100,$TIMES      ;;DO 100 ITERATIONS
  847   006520  005737  022134            TST     WFAG             ;CHECK IF 'WFCHK' FOUND AN MNCAG
  (3)   006524  001470                    BEQ     TST40            ;;BR IF NO MNCAG FOUND
  (1)   006526  012737  006540  001110    MOV     #1$,$LPERR       ;LOAD ERROR RETURN
  (1)   006534  004737  017040            JSR     PC,CLRCHT        ;DO CONVERSION ON AG CHANNELS TO INIT. THE LOGIC
  (1)                             ;NOW SELECT CHANNEL 10 BUT DONT TELL THE TESTER TO 'HOLD'
  (1)                             ;        CHECK FOR FALSE 'MNCAG HOLD'
  (1)   006540  112777  000010  172712  1$:     MOVB    #10,@ADST1       ;LOAD MUX WITH MNCAG CHANNEL
  (1)   006546  012737  000001  001124    MOV     #1,$GDDAT        ;LOAD EXPECTED DATA
  (1)   006554  017737  172746  001126    MOV     @DRVDIR,$BDDAT   ;READ TESTER INPUT REGISTER
  (1)   006562  042737  177776  001126    BIC     #177776,$BDDAT   ;MASK OFF OTHER BITS
  (1)   006570  001001                    BNE     2$               ;BR IF BIT IS ON
  (1)   006572  104010                    ERROR   10               ;UNEXPECTED 'HOLD' SENSED FROM M.U.T. CHANNEL 10
  (1)                             ;NOW TELL THE TESTER TO 'HOLD' THE CHANNEL
  (1)                             ;        AND VERIFY THAT MNCAG CHANNEL DOES HOLD
  (1)   006574  012777  000170  172722  2$:     MOV     #170,@DRVDOR     ;TELL TESTER TO HOLD
  (1)   006602  042777  000010  172714    BIC     #10,@DRVDOR      ;BY SETTING ALL THESE BITS AND CLEARING
  (1)                                                              ;THE BIT FOR THE CHANNEL
  (1)   006610  112777  000010  172642    MOVB    #10,@ADST1       ;RE-CLOCK 'QUAD HOLD BUFFER LATCH'
  (1)                                                              ;IN THE MNCAG 'HOLD' LOGIC
  (1)   006616  005037  001124            CLR     $GDDAT           ;CLEAR EXPECTED VALUE
  (1)   006622  017737  172700  001126    MOV     @DRVDIR,$BDDAT   ;READ TESTER
  (1)   006630  042737  177776  001126    BIC     #177776,$BDDAT   ;CLEAR OFF BITS
  (1)   006636  001401                    BEQ     3$               ;BR IF BIT IS OFF
  (1)   006640  104010                    ERROR   10               ;'HOLD' FROM MNCAG FAILED TO SET CHANNEL 10
  (1)                             ;NOW CONVERT ON THE SELECTED CHANNEL AND CHECK 'HOLD' CLEARS
  (1)   006642  105277  172610  3$:     INCB    @STREG           ;CONVERT
  (1)   006646  105777  172604  4$:     TSTB    @STREG           ;WAIT FOR READY
  (1)   006652  100375                    BPL     4$
  (1)   006654  017737  172646  001126    MOV     @DRVDIR,$BDDAT   ;READ TESTER
  (1)   006662  017700  172574            MOV     @ADBUFF,R0       ;READ 10/D BUFFER
  (1)   006666  012737  000007  001124    MOV     #1,$GDDAT        ;LOAD EXPECTED
  (1)   006674  042737  177776  001124    BIC     #177776,$GDDAT   ;CLEAR OTHER BITS
  (2)   006702  001001                    BNE     5$               ;;BR IF BIT IS OFF
  (1)   006704  104010                    ERROR   10               ;'MNCAG HOLD' FAILED TO CLEAR FOR CHANNEL 10
  (1)   006706                    5$:
```

```
849                          ;;***********************************************************************
(3)                          ;*TEST 40          VERIFY 'HOLD' FROM MNCAG CHANNEL 11 (TESTER ONLY)
(3)                          ;;***********************************************************************
(2)  006706  000004         TST40:  SCOPE
(1)  006710  012737  000100  001160      MOV    #100,$TIMES        ;;DO 100 ITERATIONS
850  006716  005737  022134              TST    WFAG               ;CHECK IF 'WFCHK' FOUND AN MNCAG
(3)  006722  001470                      BEQ    TST41              ;;BR IF NO MNCAG FOUND
(1)  006724  012737  006736  001110      MOV    #1$,$LPERR         ;LOAD ERROR RETURN
(1)  006732  004737  017040              JSR    PC,CLRCHT          ;DO CONVERSION ON AG CHANNELS TO INIT. THE LOGIC
(1)                          ;NOW SELECT CHANNEL 11 BUT DONT TELL THE TESTER TO 'HOLD'
(1)                          ;       CHECK FOR FALSE 'MNCAG HOLD'
(1)  006736  112777  000011  172514  1$:  MOVB   #11,@ADST1         ;LOAD MUX WITH MNCAG CHANNEL
(1)  006744  012737  000001  001124      MOV    #1,$GDDAT          ;LOAD EXPECTED DATA
(1)  006752  017737  172550  001126      MOV    @DRVDIR,$BDDAT     ;READ TESTER INPUT REGISTER
(1)  006760  042737  177776  001126      BIC    #177776,$BDDAT     ;MASK OFF OTHER BITS
(1)  006766  001001                      BNE    2$                 ;BR IF BIT IS ON
(1)  006770  104010                      ERROR  10                 ;UNEXPECTED 'HOLD' SENSED FROM M.U.T. CHANNEL 11
(1)                          ;NOW TELL THE TESTER TO 'HOLD' THE CHANNEL
(1)                          ;       AND VERIFY THAT MNCAG CHANNEL DOES HOLD
(1)  006772  012777  000170  172524  2$:  MOV    #170,@DRVDOR       ;TELL TESTER TO HOLD
(1)  007000  042777  000020  172516      BIC    #20,@DRVDOR        ;BY SETTING ALL THESE BITS AND CLEARING
(1)                                                                 ;THE BIT FOR THE CHANNEL
(1)  007006  112777  000011  172444      MOVB   #11,@ADST1         ;RE-CLOCK 'QUAD HOLD BUFFER LATCH'
(1)                                                                 ;IN THE MNCAG 'HOLD' LOGIC
(1)  007014  005037  001124              CLR    $GDDAT             ;CLEAR EXPECTED VALUE
(1)  007020  017737  172502  001126      MOV    @DRVDIR,$BDDAT     ;READ TESTER
(1)  007026  042737  177776  001126      BIC    #177776,$BDDAT     ;CLEAR OFF BITS
(1)  007034  001401                      BEQ    3$                 ;BR IF BIT IS OFF
(1)  007036  104010                      ERROR  10                 ;'HOLD' FROM MNCAG FAILED TO SET CHANNEL 11
(1)                          ;NOW CONVERT ON THE SELECTED CHANNEL AND CHECK 'HOLD' CLEARS
(1)  007040  105277  172412  3$:  INCB   @STREG             ;CONVERT
(1)  007044  105777  172406  4$:  TSTB   @STREG             ;WAIT FOR READY
(1)  007050  100375                      BPL    4$
(1)  007052  017737  172450  001126      MOV    @DRVDIR,$BDDAT     ;READ TESTER
(1)  007060  017700  172376              MOV    @ADBUFF,R0         ;READ 11/D BUFFER
(1)  007064  012737  000001  001124      MOV    #1,$GDDAT          ;LOAD EXPECTED
(1)  007072  042737  177776  001124      BIC    #177776,$GDDAT     ;CLEAR OTHER BITS
(2)  007100  001001                      BNE    5$                 ;;BR IF BIT IS OFF
(1)  007102  104010                      ERROR  10                 ;'MNCAG HOLD' FAILED TO CLEAR FOR CHANNEL 11
(1)  007104                       5$:
```

```
  852                                    ;;************************************************************
  (3)                                    ;*TEST 41        VERIFY 'HOLD' FROM MNCAG CHANNEL 12 (TESTER ONLY)
  (3)                                    ;;************************************************************
  (2)  007104  000004             TST41:  SCOPE
  (1)  007106  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
  853  007114  005737  022134                    TST     WFAG            ;CHECK IF 'WFCHK' FOUND AN MNCAG
  (3)  007120  001470                            BEQ     TST42           ;;BR IF NO MNCAG FOUND
  (1)  007122  012737  007134  001110            MOV     #1$,$LPERR      ;LOAD ERROR RETURN
  (1)  007130  004737  017040                    JSR     PC,CLRCHT       ;DO CONVERSION ON AG CHANNELS TO INIT. THE LOGIC
  (1)                                    ;NOW SELECT CHANNEL 12 BUT DONT TELL THE TESTER TO 'HOLD'
  (1)                                    ;       CHECK FOR FALSE 'MNCAG HOLD'
  (1)  007134  112777  000012  172316    1$:     MOVB    #12,@ADST1      ;LOAD MUX WITH MNCAG CHANNEL
  (1)  007142  012737  000001  001124            MOV     #1,$GDDAT       ;LOAD EXPECTED DATA
  (1)  007150  017737  172352  001126            MOV     @DRVDIR,$BDDAT  ;READ TESTER INPUT REGISTER
  (1)  007156  042737  177776  001126            BIC     #177776,$BDDAT  ;MASK OFF OTHER BITS
  (1)  007164  001001                            BNE     2$              ;BR IF BIT IS ON
  (1)  007166  104010                            ERROR   10              ;UNEXPECTED 'HOLD' SENSED FROM M.U.T. CHANNEL 12
  (1)                                    ;NOW TELL THE TESTER TO 'HOLD' THE CHANNEL
  (1)                                    ;       AND VERIFY THAT MNCAG CHANNEL DOES HOLD
  (1)  007170  012777  000170  172326    2$:     MOV     #170,@DRVDOR    ;TELL TESTER TO HOLD
  (1)  007176  042777  000040  172320            BIC     #40,@DRVDOR     ;BY SETTING ALL THESE BITS AND CLEARING
  (1)                                    ;                                ;THE BIT FOR THE CHANNEL
  (1)  007204  112777  000012  172246            MOVB    #12,@ADST1      ;RE-CLOCK 'QUAD HOLD BUFFER LATCH'
  (1)                                    ;                                ;IN THE MNCAG 'HOLD' LOGIC
  (1)  007212  005037  001124                    CLR     $GDDAT          ;CLEAR EXPECTED VALUE
  (1)  007216  017737  172304  001126            MOV     @DRVDIR,$BDDAT  ;READ TESTER
  (1)  007224  042737  177776  001126            BIC     #177776,$BDDAT  ;CLEAR OFF BITS
  (1)  007232  001401                            BEQ     3$              ;BR IF BIT IS OFF
  (1)  007234  104010                            ERROR   10              ;'HOLD' FROM MNCAG FAILED TO SET CHANNEL 12
  (1)                                    ;NOW CONVERT ON THE SELECTED CHANNEL AND CHECK 'HOLD' CLEARS
  (1)  007236  105277  172214    3$:     INCB    @STREG          ;CONVERT
  (1)  007242  105777  172210    4$:     TSTB    @STREG          ;WAIT FOR READY
  (1)  007246  100375                            BPL     4$
  (1)  007250  017737  172252  001126            MOV     @DRVDIR,$BDDAT  ;READ TESTER
  (1)  007256  017700  172200                    MOV     @ADBUFF,R0      ;READ 12/D BUFFER
  (1)  007262  012737  000001  001124            MOV     #1,$GDDAT       ;LOAD EXPECTED
  (1)  007270  042737  177776  001124            BIC     #177776,$GDDAT  ;CLEAR OTHER BITS
  (2)  007276  001001                            BNE     5$              ;;BR IF BIT IS OFF
  (1)  007300  104010                            ERROR   10              ;'MNCAG HOLD' FAILED TO CLEAR FOR CHANNEL 12
  (1)  007302                            5$:
```

```
 855                              ;;************************************************************
 (3)                              ;*TEST 42           VERIFY 'HOLD' FROM MNCAG CHANNEL 13 (TESTER ONLY)
 (3)                              ;;************************************************************
 (2)  007302  000004         TST42:  SCOPE
 (1)  007304  012737  000100  001160     MOV    #100,$TIMES        ;;DO 100 ITERATIONS
 856  007312  005737  022134               TST    WFAG             ;CHECK IF 'WFCHK' FOUND AN MNCAG
 (3)  007316  001470                       BEQ    TST43            ;;BR IF NO MNCAG FOUND
 (1)  007320  012737  007332  001110     MOV    #1$,$LPERR         ;LOAD ERROR RETURN
 (1)  007326  004737  017040               JSR    PC,CLRCHT        ;DO CONVERSION ON AG CHANNELS TO INIT. THE LOGIC
 (1)                              ;NOW SELECT CHANNEL 13 BUT DONT TELL THE TESTER TO 'HOLD'
 (1)                              ;         CHECK FOR FALSE 'MNCAG HOLD'
 (1)  007332  112777  000013  172120  1$:  MOVB   #13,@ADST1       ;LOAD MUX WITH MNCAG CHANNEL
 (1)  007340  012737  000001  001124       MOV    #1,$GDDAT        ;LOAD EXPECTED DATA
 (1)  007346  017737  172154  001126       MOV    @DRVDIR,$BDDAT   ;READ TESTER INPUT REGISTER
 (1)  007354  042737  177776  001126       BIC    #177776,$BDDAT   ;MASK OFF OTHER BITS
 (1)  007362  001001                       BNE    2$               ;BR IF BIT IS ON
 (1)  007364  104010                       ERROR  10               ;UNEXPECTED 'HOLD' SENSED FROM M.U.T. CHANNEL 13
 (1)                              ;NOW TELL THE TESTER TO 'HOLD' THE CHANNEL
 (1)                              ;         AND VERIFY THAT MNCAG CHANNEL DOES HOLD
 (1)  007366  012777  000170  172130  2$:  MOV    #170,@DRVDOR     ;TELL TESTER TO HOLD
 (1)  007374  042777  000100  172122       BIC    #100,@DRVDOR     ;BY SETTING ALL THESE BITS AND CLEARING
 (1)                                                                ;THE BIT FOR THE CHANNEL
 (1)  007402  112777  000013  172050       MOVB   #13,@ADST1       ;RE-CLOCK 'QUAD HOLD BUFFER LATCH'
 (1)                                                                ;IN THE MNCAG 'HOLD' LOGIC
 (1)  007410  005037  001124               CLR    $GDDAT           ;CLEAR EXPECTED VALUE
 (1)  007414  017737  172106  001126       MOV    @DRVDIR,$BDDAT   ;READ TESTER
 (1)  007422  042737  177776  001126       BIC    #177776,$BDDAT   ;CLEAR OFF BITS
 (1)  007430  001401                       BEQ    3$               ;BR IF BIT IS OFF
 (1)  007432  104010                       ERROR  10               ;'HOLD' FROM MNCAG FAILED TO SET CHANNEL 13
 (1)                              ;NOW CONVERT ON THE SELECTED CHANNEL AND CHECK 'HOLD' CLEARS
 (1)  007434  105277  172016  3$:  INCB   @STREG           ;CONVERT
 (1)  007440  105777  172012  4$:  TSTB   @STREG           ;WAIT FOR READY
 (1)  007444  100375                       BPL    4$
 (1)  007446  017737  172054  001126       MOV    @DRVDIR,$BDDAT   ;READ TESTER
 (1)  007454  017700  172002               MOV    @ADBUFF,R0       ;READ 13/D BUFFER
 (1)  007460  012737  000001  001124       MOV    #1,$GDDAT        ;LOAD EXPECTED
 (1)  007466  042737  177776  001124       BIC    #177776,$GDDAT   ;CLEAR OTHER BITS
 (2)  007474  001001                       BNE    5$               ;;BR IF BIT IS OFF
 (1)  007476  104010                       ERROR  10               ;'MNCAG HOLD' FAILED TO CLEAR FOR CHANNEL 13
 (1)  007500                       5$:
 857
```

```
859                                      ;;******************************************************************
(3)                                      ;*TEST 43           MNCAG GAIN BITS LOGIC TESTS.
(3)                                      ;;******************************************************************
(2)   007500  000004                     TST43:  SCOPE
(1)   007502  012737  000200  001160             MOV     #200,$TIMES         ;;DO 200 ITERATIONS
860                                      ;NOW TO PROVE THAT THE MNCAG LOGIC IS WORKING CORRECTLY
861                                      ;       1ST.    WRITE CH00-77 WITH GAIN BITS = 01
862                                      ;       2ND.    WRITE CHXX WITH GAIN BITS = 10
863                                      ;       3RD.    READ CHXX AND CHECK GAIN BITS = 10
864                                      ;       4TH.    READ CH00-77 EXCEPT CHXX AND CHECK GAIN STILL = 01
865   007510  012737  000010  017764             MOV     #10,CHXX            ;PRIME THE CHANNEL UNDER TEST TO 10
866
867   007516  013700  017764            1$:      MOV     CHXX,R0             ;GET CHANNEL VALUE
868   007522  004737  014146                     JSR     PC,CHKAGC           ;CHECK IF THIS IS AN MNCAG CHANNEL
869   007526  001002                             BNE     2$                  ;BR IF NOT
870   007530  004737  014320                     JSR     PC,CHKGAN           ;READ-WRITE TEST OF GAIN BITS
871   007534  005237  017764            2$:      INC     CHXX                ;UPDATE TESTED CHANNEL
872   007540  022737  000100  017764             CMP     #100,CHXX           ;TEST IF ALL CHANNELS HAVE BEEN RUN
873   007546  001363                             BNE     1$                  ;BR IF NOT
874
875                                      ;;******************************************************************
(3)                                      ;*TEST 44           MNCTP GAIN AND TC TYPE BITS LOGIC TESTS
(3)                                      ;;******************************************************************
(2)   007550  000004                     TST44:  SCOPE
(1)   007552  012737  000200  001160             MOV     #200,$TIMES         ;;DO 200 ITERATIONS
876                                      ;NOW TO PROVE THAT THE MNCTP GAIN LOGIC IS WORKING CORRECTLY
877                                      ;       1ST.    WRITE CH00-77 WITH GAIN BITS = 0001
878                                      ;       2ND.    WRITE CHXX WITH GAIN BITS = 1000,0100, OR 0010
879                                      ;       3RD.    READ CHXX AND CHECK GAIN = 1000,0100, OR 0010
880                                      ;       4TH.    READ CH00-77 EXCEPT CHXX AND CHECK GAIN STILL = 0001
881                                      ;
882   007560  012737  000010  017764             MOV     #10,CHXX            ;PRIME TO CHANNEL 10 (8)
883
884   007566  013700  017764            1$:      MOV     CHXX,R0             ;GET CHANNEL XX
885   007572  004737  014174                     JSR     PC,CHKTCC           ;CHECK IF THIS IS AN ''TP'' CHANNEL
886   007576  001002                             BNE     2$                  ;BR IF NOT AN ''TP'' CHANNEL
887   007600  004737  014534                     JSR     PC,CHKTCN           ;READ-WRITE TEST OF GAIN BITS
888   007604  005237  017764            2$:      INC     CHXX                ;UPDATE TO NEXT CHANNEL
889   007610  022737  000100  017764             CMP     #100,CHXX           ;TEST IF ALL CHANNELS DONE
890   007616  001363                             BNE     1$                  ;BR IF MORE CHANNELS
891
```

```
 893                                       ;;************************************************************
 (3)                                       ;*TEST 45          CHECK THAT "CLOCK STARTS" ON MNCAG CHANNELS DO NOT SET ERROR FLAG
 (3)                                       ;;************************************************************
 (2)   007620  000004                      TST45:   SCOPE
 (1)   007622  012737  000010  001160               MOV     #10,$TIMES          ;;DO 10 ITERATIONS
 894   007630  005737  001604                       TST     KWAD                ;TEST IF MNCKW IS AVAILABLE
 895   007634  001464                               BEQ     TST46               ;;BR IF NO MNCKW
 896   007636  012737  000010  017764               MOV     #10,CHXX            ;LOAD STARTING CHANNEL
 897
 898   007644  005077  171606              1$:      CLR     @STREG              ;CLEAR A/D STATUS
 899   007650  017700  171606                       MOV     @ADBUFF,R0          ;READ BUFFER
 900   007654  013700  017764                       MOV     CHXX,R0             ;LOAD CHANNEL NUMBER
 901   007660  004737  014146                       JSR     PC,CHKAGC           ;TEST IF MNCAG CHANNEL
 902   007664  001040                               BNE     3$                  ;BR IF NOT MNCAG CHANNEL
 903   007666  013701  001434                       MOV     BARF0,R1                    ;LOAD DELAY COUNTER
 904   007672  012777  177760  171604               MOV     #-20,@KWBPR         ;LOAD CLOCK PRESET
 905   007700  012777  000040  171550               MOV     #BIT5,@STREG        ;ENABLE "CLOCK STARTS"
 906   007706  110077  171546                       MOVB    R0,@ADST1           ;SELECT MNCAG CHANNEL
 907   007712  012777  000011  171562               MOV     #11,@KWCSR          ;START CLOCK
 908   007720  005777  171532              2$:      TST     @STREG              ;TEST ERROR FLAG
 909   007724  100405                               BMI     4$                  ;BR IF SET
 910   007726  105777  171524                       TSTB    @STREG              ;TEST DONE FLAG
 911   007732  100415                               BMI     3$                  ;BR IF SET
 912   007734  005301                               DEC     R1                  ;DELAY
 913   007736  001370                               BNE     2$                  ;BR IF NOT DONE DELAY
 914   007740  017737  171512  001126      4$:      MOV     @STREG,$BDDAT       ;READ A/D STATUS
 915   007746  005077  171530                       CLR     @KWCSR              ;ENSURE STOP CLOCK
 916   007752  012737  000240  001124               MOV     #240,$GDDAT         ;LOAD EXPECTED
 917   007760  110037  001125                       MOVB    R0,$GDDAT+1         ;LOAD CHANNEL NUMBER
 918   007764  104001                               ERROR   1                   ;CLOCK START ON MNCAG CHANNEL SET MNCAD ERROR FLAG
 919   007766  017700  171470              3$:      MOV     @ADBUFF,R0          ;READ A/D BUFF TO CLEAR A/D READY
 920   007772  005237  017764                       INC     CHXX                ;UPDATE CHANNEL
 921   007776  022737  000100  017764               CMP     #100,CHXX           ;TEST IF NOMORE CHANNELS
 922   010004  001317                               BNE     1$                  ;BR IF MORE TO TEST
```

```
924                              ;;*********************************************************************
(3)                              ;*TEST 46        CHECK THAT "CLOCK STARTS" ON MNCTP CHANNELS DO NOT SET ERROR FLAG
(3)                              ;;*********************************************************************
(2)  010006  000004             TST46:  SCOPE
(1)  010010  012737  000010  001160      MOV     #10,$TIMES       ;;DO 10 ITERATIONS
925  010016  005737  001604              TST     KWAD             ;TEST IF MNCKW IS AVAILABLE
926  010022  001464                      BEQ     TST47            ;;BR IF NO MNCKW
927  010024  012737  000010  017764      MOV     #10,CHXX         ;LOAD STARTING CHANNEL
928
929  010032  005077  171420     1$:      CLR     @STREG           ;CLEAR A/D STATUS
930  010036  017700  171420              MOV     @ADBUFF,R0       ;READ BUFFER
931  010042  013700  017764              MOV     CHXX,R0          ;LOAD CHANNEL NUMBER
932  010046  004737  014174              JSR     PC,CHKTCC        ;TEST IF MNCTP CHANNEL
933  010052  001040                      BNE     3$               ;BR IF NOT MNCTP CHANNEL
934  010054  013701  001434              MOV     BARF0,R1              ;LOAD DELAY COUNTER
935  010060  012777  177760  171416      MOV     #-20,@KWBPR      ;LOAD CLOCK PRESET
936  010066  012777  000040  171362      MOV     #BIT5,@STREG     ;ENABLE "CLOCK STARTS"
937  010074  110077  171360              MOVB    R0,@ADST1        ;SELECT MNCTP CHANNEL
938  010100  012777  000011  171374      MOV     #11,@KWCSR       ;START CLOCK
939  010106  005777  171344     2$:      TST     @STREG           ;TEST ERROR FLAG
940  010112  100405                      BMI     4$               ;BR IF SET
941  010114  105777  171336              TSTB    @STREG           ;TEST DONE FLAG
942  010120  100415                      BMI     3$               ;BR IF SET
943  010122  005301                      DEC     R1               ;DELAY
944  010124  001370                      BNE     2$               ;BR IF NOT DONE DELAY
945  010126  017737  171324  001126 4$:  MOV     @STREG,$BDDAT    ;READ A/D STATUS
946  010134  005077  171342              CLR     @KWCSR           ;ENSURE STOP CLOCK
947  010140  012737  000240  001124      MOV     #240,$GDDAT      ;LOAD EXPECTED
948  010146  110037  001125              MOVB    R0,$GDDAT+1      ;LOAD CHANNEL NUMBER
949  010152  104001                      ERROR   1                ;CLOCK START ON MNCTP CHANNEL SET MNCAD ERROR FLAG
950  010154  017700  171302     3$:      MOV     @ADBUFF,R0       ;READ A/D BUFF TO CLEAR A/D READY
951  010160  005237  017764              INC     CHXX             ;UPDATE CHANNEL
952  010164  022737  000100  017764      CMP     #100,CHXX        ;TEST IF NOMORE CHANNELS
953  010172  001317                      BNE     1$               ;BR IF MORE TO TEST
954
955                              ;;*********************************************************************
(3)                              ;*TEST 47        END OF MNCAD, MNCAG AND MNCTP LOGIC TESTS
(3)                              ;;*********************************************************************
(2)  010174  000004             TST47:  SCOPE
(1)  010176  012737  000001  001160      MOV     #1,$TIMES        ;;DO 1 ITERATION
956  010204  000207                      RTS     PC
```

```
 958                                           .SBTTL   WRAPAROUND ANALOG TEST SECTION
 959   010206                         WRAP:
 960                                  ;;****************************************************************
 (3)                                  ;*TEST 50          TEST CH0 GROUND
 (3)                                  ;;****************************************************************
 (2)   010206  012737  000050  001102 TST50:  MOV      #$TN,$TSTNM
 (1)   010214  012737  000010  001160         MOV      #10,$TIMES       ;;DO 10 ITERATIONS
 961   010222  012737  010206  001106         MOV      #TST50,$LPADR    ;*SET UP LOOP ADDRESS
 (2)   010230  012737  010206  001110         MOV      #TST50,$LPERR    ;*SET UP ERROR LOOP ADDRESS
 962   010236  004537  036332                 JSR      R5,CONVRT        ;CONVERT 8 TIMES
 963   010242  000000                         0
 964   010244  004537  036462                 JSR      R5,COMPAR        ;COMPARE RESULTS
 965   010250  004000                         4000                     ;NOMINAL
 966   010252  037372                         V12                      ;TOLERANCE
 967   010254  104004                         ERROR    4                ;ERROR ON A/D CHANNEL
 968                                  ;;****************************************************************
 (3)                                  ;*TEST 51          TEST CH1 +4.5 VOLT
 (3)                                  ;;****************************************************************
 (2)   010256  000004                 TST51:  SCOPE
 (1)   010260  012737  000010  001160         MOV      #10,$TIMES       ;;DO 10 ITERATIONS
 969   010266  004537  036332                 JSR      R5,CONVRT        ;CONVERT 8 TIMES
 970   010272  000001                         1                        ;CHANNEL 1
 971   010274  004537  036462                 JSR      R5,COMPAR        ;COMPARE RESULTS
 972   010300  007344                         7344                     ;NOMINAL
 973   010302  037400                         V326                     ;TOLERANCE
 974   010304  104004                         ERROR    4                ;ERROR ON A/D CHANNEL
 975                                  ;;****************************************************************
 (3)                                  ;*TEST 52          TEST CH2 -4.5 VOLT
 (3)                                  ;;****************************************************************
 (2)   010306  000004                 TST52:  SCOPE
 (1)   010310  012737  000010  001160         MOV      #10,$TIMES       ;;DO 10 ITERATIONS
 976   010316  004537  036332                 JSR      R5,CONVRT        ;CONVERT 8 TIMES
 977   010322  000002                         2                        ;CHANNEL 2
 978   010324  004537  036462                 JSR      R5,COMPAR        ;COMPARE RESULTS
 979   010330  000434                         434                      ;NOMINAL
 980   010332  037400                         V326                     ;TOLERANCE
 981   010334  104004                         ERROR    4                ;ERROR ON A/D CHANNEL
 982                                  ;;****************************************************************
 (3)                                  ;*TEST 53          TEST CH5 GROUND (MNCAD-TA OR TESTER EXCEPT IF MNCAG)
 (3)                                  ;;****************************************************************
 (2)   010336  000004                 TST53:  SCOPE
 (1)   010340  012737  000010  001160         MOV      #10,$TIMES       ;;DO 10 ITERATIONS
 983   010346  005737  022134                 TST      WFAG             ;TEST IF TESTING MNCAG'S
 984   010352  001402                         BEQ      1$               ;BR IF NOT
 985   010354  000137  011530                 JMP      WRAPY            ;BYPASS MANY TESTS
 986   010360  005737  001602         1$:     TST      WFTEST           ;RUNNING ON THE TESTER ?
 987   010364  100403                         BMI      2$               ;BR IF YES
 988   010366  105737  060163                 TSTB     CHTABL+5         ;TEST IF TESTING CH4-7 ?
 989   010372  100040                         BPL      WRAPX            ;BYPASS SOME TESTS
 990   010374  004537  036332         2$:     JSR      R5,CONVRT        ;CONVERT 8 TIMES
 991   010400  000005                         5                        ;CHANNEL 5
 992   010402  004537  036462                 JSR      R5,COMPAR        ;COMPARE RESULTS
 993   010406  004000                         4000                     ;NOMINAL
 994   010410  037372                         V12                      ;TOLLERANCE
 995   010412  104004                         ERROR    4                ;ERROR ON A/D CHANNEL
 996
```

```
 998                                     ;;*****************************************************************
 (3)                                     ;*TEST 54        TEST CH4 +2.6 VOLTS       (MNCAD-TA OR TESTER)
 (3)                                     ;;*****************************************************************
 (2)   010414  000004                    TST54:  SCOPE
 (1)   010416  012737  000010  001160            MOV     #10,$TIMES       ;;DO 10 ITERATIONS
 999   010424  004537  036332                    JSR     R5,CONVRT        ;CONVERT 8 TIMES
1000   010430  000004                            4                        ;CHANNEL 4
1001   010432  004537  036462                    JSR     R5,COMPAR        ;COMPARE RESULTS
1002   010436  006020                            6020                     ;NOMINAL
1003   010440  037400                            V326                     ;TOLLERANCE
1004   010442  104004                            ERROR   4                ;ERROR ON A/D CHANNEL
1005                                     ;;*****************************************************************
 (3)                                     ;*TEST 55        TEST CH6 -2.2 VOLTS       (MNCAD-TA OR TESTER)
 (3)                                     ;;*****************************************************************
 (2)   010444  000004                    TST55:  SCOPE
 (1)   010446  012737  000010  001160            MOV     #10,$TIMES       ;;DO 10 ITERATIONS
1006   010454  004537  036332                    JSR     R5,CONVRT        ;CONVERT 8 TIMES
1007   010460  000006                            6                        ;CHANNEL 6
1008   010462  004537  036462                    JSR     R5,COMPAR        ;COMPARE RESULTS
1009   010466  001760                            1760                     ;NOMINAL
1010   010470  037400                            V326                     ;TOLLERANCE
1011   010472  104004                            ERROR   4                ;ERROR ON A/D CHANNEL
1012   010474                            WRAPX:
1013                                     ;;*****************************************************************
 (3)                                     ;*TEST 56        TEST VOLTAGE ON SINGLE-ENDED CHANNELS (MNCAD-TA OR MNCAM-TA OR TESTER)
 (3)                                     ;;*****************************************************************
 (2)   010474  000004                    TST56:  SCOPE
 (1)   010476  012737  000010  001160            MOV     #10,$TIMES       ;;DO 10 ITERATIONS
1014   010504  012737  000056  001102            MOV     #$TN-1,$TSTNM    ;SET UP TEST NUMBER
1015   010512  012702  060166                    MOV     #CHTABL+10,R2    ;LOAD POINTER TO CHANNEL LIST
1016   010516  105712                    1$:     TSTB    (R2)             ;TEST IF EXISTANT CHANNEL
1017   010520  001446                            BEQ     TST57            ;;BR IF NO MORE CHANNLES
1018   010522  100043                            BPL     4$               ;BR IF NOT TO TEST THIS CHANNEL
1019   010524  111237  023176                    MOVB    (R2),CHA         ;GET TYPE OF CHANNEL
1020   010530  042737  177700  023176            BIC     #177700,CHA      ;MASK OFF OTHER BITS
1021   010536  022737  000001  023176            CMP     #1,CHA           ;TEST IF A SINGLE ENDED CHANNEL
1022   010544  001032                            BNE     4$               ;BR IF NOT S.E. CHANNEL
1023   010546  010203                            MOV     R2,R3            ;COPY R2
1024   010550  162703  060156                    SUB     #CHTABL,R3       ;CONVERT INDEX INTO CHANNEL NUMBER
1025   010554  010337  001556                    MOV     R3,CHANL         ;SAVE CHANNEL NUMBER
1026   010560  012703  037622                    MOV     #VTABLE,R3       ;MAKE INDEX INTO EXPECTED VALUE TABLE
1027   010564  012337  010600            2$:     MOV     (R3)+,3$         ;GET EXPECTED VALUE
1028   010570  004537  036340                    JSR     R5,CONVTC        ;CONVERT 8 TIMES
1029   010574  004537  036462                    JSR     R5,COMPAR        ;COMPARE RESULTS
1030   010600  005560                    3$:     5560                     ;VOLTAGE
1031   010602  037400                            V326                     ;TOLLERANCE
1032   010604  104004                            ERROR   4                ;ERROR ON SINGLE ENDED A/D CHANNEL
1033   010606  022737  000077  001556            CMP     #77,CHANL        ;TEST IF LAST CHANNEL IN SYSTEM
1034   010614  001410                            BEQ     TST57            ;;BR IF LAST
1035   010616  005237  001556                    INC     CHANL            ;UPDATE CHANNEL NUMBER
1036   010622  005713                            TST     (R3)             ;TEST IF END OF LIST
1037   010624  100357                            BPL     2$               ;BR IF NOT
1038   010626  062702  000007                    ADD     #7,R2            ;UPDATE CHANNEL LOOKUP VALUE
1039   010632  105722                    4$:     TSTB    (R2)+            ;BUMP CHANNEL POINTER
1040   010634  000730                            BR      1$               ;TEST NEXT CHANNEL
```

```
 1042                                        ;;********************************************************************
  (3)                                        ;*TEST 57         TEST VOLTAGE ON DIFFERENTIAL CHANNELS (MNCAD-TA OR MNCAM-TA OR TESTER)
  (3)                                        ;;********************************************************************
  (2)   010636  000004                TST57:  SCOPE
  (1)   010640  012737  000001  001160         MOV     #1,$TIMES           ;;DO 1 ITERATION
 1043
 1044   010646  012702  060166                 MOV     #CHTABL+10,R2       ;LOAD POINTER TO CHANNEL LIST
 1045   010652  012737  010666  001106         MOV     #1$,$LPADR          ;SET UP LOOP ADDRESS
 1046   010660  012737  010666  001110         MOV     #1$,$LPERR          ;SET UP ERROR LOOP ADDRESS
 1047   010666  105712                1$:      TSTB    (R2)                ;TEST IF EXISTANT CHANNEL
 1048   010670  001443                         BEQ     TST60               ;;BR IF NOT
 1049   010672  100040                         BPL     3$                  ;BR IF NOT TO TEST THE CHANNEL
 1050   010674  111237  023176                 MOVB    (R2),CHA            ;GET CHANNEL TYPE
 1051   010700  042737  177700  023176         BIC     #177700,CHA         ;MASK OFF OTHER BITS
 1052   010706  022737  000002  023176         CMP     #2,CHA              ;TEST IF DIFFERENTIAL CHANNEL
 1053   010714  001027                         BNE     3$                  ;BR IF NOT A DIFF. CHANNEL
 1054   010716  010203                         MOV     R2,R3               ;COPY R2
 1055   010720  162703  060156                 SUB     #CHTABL,R3          ;CREATE CHANNEL NUMBER FROM OFFSET
 1056   010724  010337  001556                 MOV     R3,CHANL            ;SAVE CHANNEL NUMBER
 1057   010730  012737  002220  010766         MOV     #2220,2$            ;SET UP INITIAL EXPECTED VALUE -2.2 V
 1058   010736  032703  000001                 BIT     #BIT0,R3            ;TEST IF ODD OR EVEN CHANNEL
 1059   010742  001405                         BEQ     5$                  ;BR IF EVEN CHANNEL
 1060   010744  005437  010766                 NEG     2$                  ;CONVERT EXPECTED VALUE
 1061   010750  042737  170000  010766         BIC     #170000,2$          ;MASK OFF OTHER BITS
 1062   010756  004537  036340         5$:     JSR     R5,CONVTC           ;CONVERT 8 TIMES
 1063   010762  004537  036462                 JSR     R5,COMPAR           ;TEST RESULTS
 1064   010766  002220                2$:      2220                        ;NOMINAL
 1065   010770  037400                         V326                        ;TOLLERANCE
 1066   010772  104004                         ERROR   4                   ;ERROR ON A/D CHANNEL
 1067   010774  105722                3$:      TSTB    (R2)+               ;BUMP THE CHANNEL POINTER
 1068   010776  000733                         BR      1$                  ;RETEST
```

```
 1070                                    ;;**********************************************************
  (3)                                    ;*TEST 60        TEST VOLTAGES ON MNCTP CHANNELS (MNCTP-TA SWITCHES 'OFF')
  (3)                                    ;;**********************************************************
  (2)  011000  000004          TST60:  SCOPE
  (1)  011002  012737  000001  001160           MOV    #1,$TIMES           ;:DO 1 ITERATION
 1071  011010  005737  001176                   TST    $PASS               ;TEST IF FIRST PASS EXECUTION
 1072  011014  001121                           BNE    TST61               ;:BR IF NOT FIRST PASS
 1073  011016  005737  001430                   TST    TPTA                ;TEST IF MNCTP-TA IS CONNECTED?
 1074  011022  001516                           BEQ    TST61               ;:BR IF NO MNCTP-TA
 1075  011024  012737  011036  001110           MOV    #1$,$LPERR          ;LOAD ERROR RETURN ADDRESS
 1076  011032  012702  060166                   MOV    #CHTABL+10,R2       ;LOAD POINTER TO CHANNEL TYPE BUFFER
 1077  011036  105712                   1$:     TSTB   (R2)                ;TEST IF EXISTANT CHANNEL
 1078  011040  001507                           BEQ    TST61               ;:BR IF NO MORE
 1079  011042  100104                           BPL    4$                  ;BR IF DONT TEST THIS CHANNEL
 1080  011044  111237  023176                   MOVB   (R2),CHA            ;GET CHANNEL TYPE
 1081  011050  042737  177700  023176           BIC    #177700,CHA         ;MASK OFF OTHER BITS
 1082  011056  022737  000004  023176           CMP    #4,CHA              ;TEST IF MNCTP CHANNEL
 1083  011064  001073                           BNE    4$                  ;BR IF NOT MNCTP CHANNEL
 1084  011066  004737  017566                   JSR    PC,SRLOOP           ;TEST IF LOOPING ON THIS TEST FOR ANY REASON
 1085  011072  000406                           BR     10$                 ;BR IF YES, SO WE DONT KEEP TYPING THE TEXT STUFF
 1086  011074  104401  045721                   TYPE   .TPSWOF             ;TELL OPERATOR TO SET THE MNCTP-TA SWITCHES
 1087                                                                      ;TO THE OFF (OPEN OR NOT SHORTED) POSITION
 1088  011100  104401  047151                   TYPE   .CRWR               ;TELL OPERATOR TO DEPRESS ''CR'' WHEN READY
 1089  011104  104412                           RDLIN
 1090  011106  005726                           TST    (SP)+
 1091  011110  010237  001556          10$:     MOV    R2,CHANL            ;COPY CHANNEL INDEX
 1092  011114  162737  060156  001556           SUB    #CHTABL,CHANL       ;REMOVE STARTING VALUE
 1093  011122  012703  037456                   MOV    #TPVALS,R3          ;LOAD POINTER TO CONVERTED VALUES
 1094  011126  012337  011222          5$:      MOV    (R3)+,2$            ;GET EXPECTED VALUE
 1095  011132  012337  011224                   MOV    (R3)+,6$            ;GET ALLOWABLE SPREAD
 1096  011136  013704  011222                   MOV    2$,R4               ;GET GAIN BITS
 1099  011142  006104                           ROL    R4                  ;MOVE LEFT
  (1)  011144  006104                           ROL    R4                  ;MOVE LEFT
  (1)  011146  006104                           ROL    R4                  ;MOVE LEFT
  (1)  011150  006104                           ROL    R4                  ;MOVE LEFT
  (1)  011152  006104                           ROL    R4                  ;MOVE LEFT
 1100  011154  042704  177760                   BIC    #177760,R4          ;MASK OFF ALL BUT GAIN/TC TYPE
 1101  011160  010437  013546                   MOV    R4,TPVAL            ;SAVE FOR ERROR REPORT IF NEEDED
 1102  011164  112777  000077  170266           MOVB   #77,@ADST1          ;LOAD
 1103  011172  110477  170262                   MOVB   R4,@ADST1           ;        GAIN AND TC
 1104  011176  113777  001556  170254           MOVB   CHANL,@ADST1        ;               TYPE
 1105  011204  042737  170000  011222           BIC    #170000,2$          ;MASK OFF GAIN INFORMATION
 1106  011212  004537  036340                   JSR    R5,CONVTC           ;CONVERT 8 TIMES
 1107  011216  004537  036462                   JSR    R5,COMPAR           ;COMPARE RESULTS
 1108  011222  000000          2$:      0                                  ;EXPECTED VALUE
 1109  011224  000000          6$:      0                                  ;TOLERANCE
 1110  011226  104015                           ERROR  15                  ;INCORRECT VALUE ON MNCTP CHANNEL
 1111  011230  022737  000077  001556  3$:      CMP    #77,CHANL           ;LAST CHANNEL
 1112  011236  001410                           BEQ    TST61               ;:BR IF END CHANNEL
 1113  011240  005237  001556                   INC    CHANL               ;UPDATE CHANNEL
 1114  011244  005713                           TST    (R3)                ;TEST IS END OF TABLE
 1115  011246  001327                           BNE    5$                  ;BR IF YES
 1116  011250  062702  000007                   ADD    #7,R2               ;UPDATE POINTER
 1117  011254  105722                   4$:      TSTB   (R2)+               ;BUMP POINTER
 1118  011256  000667                           BR     1$
```

```
1120                                           ;;**********************************************************
 (3)                                           ;*TEST 61          MNCTP INTERCHANNEL SETTLING TEST (MNCTP-TA SWITCHES 'OFF')
 (3)                                           ;;**********************************************************
 (2)   011260  000004                          TST61:   SCOPE
 (1)   011262  012737  000001  001160                   MOV      #1,$TIMES          ;;DO 1 ITERATION
1121   011270  005737  001176                           TST      $PASS              ;TEST IF FIRST PASS
1122   011274  001115                                   BNE      TST62              ;;BR IF NOT
1123   011276  005737  001430                           TST      TPTA               ;TEST IF MNCTP-TA AVAILABLE
1124   011302  001512                                   BEQ      TST62              ;;BR IF NOT
1125   011304  000240                                   NOP
1126   011306  000240                                   NOP
1127   011310  000240                                   NOP
1128   011312  005037  011476                           CLR      12$                ;CLEAR A FLAG
1129                                           ;DETERMINE IF THE MNCTP CHANNELS ARE TO RUN THE SETTLING TEST
1130   011316  012737  060166  011474  1$:     MOV      #CHTABL+10,10$     ;LOAD CHANNEL TYPE POINTER
1131   011324  013700  011474          2$:     MOV      10$,R0             ;GET CHANNEL POINTER
1132   011330  105710                                   TSTB     (R0)               ;TEST IF CHANNEL IS TO BE TESTED
1133   011332  001462                                   BEQ      4$                 ;BR IF NONE EXISTS
1134   011334  100054                                   BPL      3$                 ;BR IF DO NOT TEST IT
1135   011336  111037  001544                           MOVB     (R0),CH1           ;GET CHANNEL TYPE
1136   011342  042737  177700  001544                   BIC      #177700,CH1        ;MASK OFF OTHER BITS
1137   011350  022737  000004  001544                   CMP      #4,CH1             ;TEST IF MNCTP TYPE
1138   011356  001043                                   BNE      3$                 ;BR IF NOT
1139   011360  005237  011476                           INC      12$                ;SET THE FLAG
1140   011364  004737  017566                           JSR      PC,SRLOOP          ;TEST IF LOOPING ON THE TEST
1141   011370  000405                                   BR       11$
1142   011372  005737  001624                           TST      QUIET              ;TEST QUIET FLAG
1143   011376  001002                                   BNE      11$                ;BR IF QUIET MODE
1144   011400  104401  041343                           TYPE     ,SETTPM            ;TELL OPERATOR WHAT'S RUNNING
1145   011404  010037  001544          11$:    MOV      R0,CH1             ;LOAD CHANNEL #
1146   011410  162737  060156  001544           SUB      #CHTABL,CH1        ;REMOVE INDEX VALUE
1147   011416  022737  000074  001544           CMP      #74,CH1            ;TEST IF THIS MNCTP STARTS ON CHANNEL 74
1148   011424  001425                                   BEQ      4$                 ;BYPASS SETTLING ON LAST MNCTP CHANNEL
1149   011426  062737  000003  001544           ADD      #3,CH1             ;AND ADJUST TO SETTLING TEST CHANNELS
1150   011434  013737  001544  001546           MOV      CH1,CH2            ;        ..
1151   011442  005237  001546                           INC      CH2                ;        ..
1152   011446  004537  014104                           JSR      R5,LDSETG          ;LOAD THE TP TYPE AND GAIN VALUE
1153   011452     011·      011                         .BYTE    11,11              ;TYPE 11 - 11
1154   011454  004737  013662                           JSR      PC,SETTLE          ;RUN THE TEST
1155   011460  062737  000007  011474           ADD      #7.,10$            ;UPDATE CHANNEL
1156   011466  005237  011474          3$:     INC      10$                ;   ..    ..
1157   011472  000714                                   BR       2$                 ;TRY NEXT CHANNEL
1158   011474  000000                  10$:    0
1159   011476  000000                  12$:    0
1160   011500  004737  017566          4$:     JSR      PC,SRLOOP          ;TEST IF WE ARE LOOPING ON THIS TEST FOR ANY REASON
1161   011504  000411                                   BR       TST62              ;;BR IF YES, SO WE DONT KEEP TYPING THE TEXT STUFF
1162   011506  005737  011476                           TST      12$                ;TEST IF ANY MNCTP'S WERE TESTED
1163   011512  001406                                   BEQ      TST62              ;;BR IF NO MNCTP'S WERE SELECTED
1164   011514  104401  046011                           TYPE     ,TPSWON            ;TELL OPERATOR TO SET THE MNCTP-TA SWITCHES
1165                                                                                ;TO THE ON (CLOSED OR SHORTED) POSITION
1166   011520  104401  047151                           TYPE     ,CRWR              ;TELL OPERATOR TO DEPRESS ''CR'' WHEN READY
1167   011524  104412                                   RDLIN
1168   011526  005726                                   TST      (SP)+              ;THROW IT AWAY!
```

```
 1170   011530                            WRAPY:
 1171                                      ;;************************************************************
 (3)                                       ;*TEST 62       TEST VERNIER OFFSET DAC ON MNCAD CH0
 (3)                                       ;;************************************************************
 (2)   011530  000004                      TST62:  SCOPE
 (1)   011532  012737  000001  001160              MOV     #1,$TIMES       ;;DO 1 ITERATION
 1172  011540  012737  000062  001102              MOV     #$TN-1,$TSTNM   ;SET UP TEST NUMBER
 1173  011546  005077  167710              CLR     @ADBUFF         ;SET VERNIER DAC = 0
 1174  011552  005037  001556              CLR     CHANL           ;SET UP TO CONVERT ON CHANNEL 0
 1175  011556  004537  036346              JSR     R5,CONVCD       ;CONV. CH0, DIRECT VERNIER DAC
 1176  011562  013704  001540              MOV     TEMP,R4         ;SAVE VALUE IN R4
 1177  011566  012777  000377  167666  1$: MOV     #377,@ADBUFF    ;SET VERNIER DAC = 377
 1178  011574  004537  036346              JSR     R5,CONVCD       ;CONVERT IT
 1179  011600  160437  001540              SUB     R4,TEMP         ;TEMP=DIFF. BETWEEN VALUE & PREVIOUS
 1180  011604  004537  036462              JSR     R5,COMPAR       ;COMPARE RESULTS
 1181  011610  000005                      5
 1182  011612  037366                      V2
 1183  011614  104004                      ERROR   4
 1184                                      ;;************************************************************
 (3)                                       ;*TEST 63       OFFSET ON MNCAD CH0
 (3)                                       ;;************************************************************
 (2)   011616  000004                      TST63:  SCOPE
 (1)   011620  012737  000001  001160              MOV     #1,$TIMES       ;;DO 1 ITERATION
 1185  011626  005037  001556              CLR     CHANL           ;LOAD CHANNEL
 1186  011632  005037  001554              CLR     DUMMY           ;LOAD DUMMY
 1187  011636  004737  015252              JSR     PC,OFFSET       ;FIND OFFSET
 1188  011642  005037  011756              CLR     77$             ;INIT THE PASSING FLAG
 1189  011646  004537  036462              JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
 1190  011652  000000                      0
 1191  011654  037374                      V50D
 1192  011656  000401                      BR      10$             ;ERROR RETURN
 1193  011660  000402                      BR      11$             ;NOT AN ERROR
 1194  011662  005137  011756      10$:    COM     77$             ;ERROR RETURN - SET FAIL FLAG
 1195  011666  005737  001624      11$:    TST     QUIET           ;TEST IF QUIET MODE
 1196  011672  001403                      BEQ     1$              ;BR IF NOT QUIET MODE
 1197  011674  005737  011756              TST     77$             ;TEST IF ERROR
 1198  011700  001431                      BEQ     TST64           ;;BR IF NO FAILURE
 1199  011702  104401  041374      1$:     TYPE    ,OFSET          ;TELL OPERATOR THE TEST
 1200  011706  004737  055712              JSR     PC,WHICHU       ;GET UNIT #
 1201  011712  013746  001622              MOV     UNITBD,-(SP)    ;PUSH IT FOR TYPE OUT
 1202  011716  104403                      TYPOS
 1203  011720     001     000              .BYTE   1,0
 1204  011722  104401  050631              TYPE    ,MOFSET         ;TYPE OFFSET=''
 1205  011726  004737  015502              JSR     PC,TOFF         ;TYPE OFFSET VALUE
 1206  011732  005737  011756              TST     77$             ;TEST IF FAILURE
 1207  011736  001410                      BEQ     2$              ;;BR IF NO ERROR
 1208  011740  104401  047505              TYPE    ,ERMSG
 1209  011744  004737  055704              JSR     PC,WHICHV       ;INDICATE BAD UNIT
 1210  011750  005237  001112              INC     $ERTTL          ;UPDATE ERROR COUNT
 1211  011754  000403                      BR      TST64           ;;GO TO NEXT TEST
 1212  011756  000000              77$:    0                       ;NON-ZERO = FAILURE
 1213  011760  104401  046620      2$:     TYPE    ,OKMSG
```

```
 1215                                       ;:******************************************************************
  (3)                                       ;*TEST 64        OFFSET ON THE MNCTP CHANNELS (MNCTP-TA SWITCHES 'ON')
  (3)                                       ;:******************************************************************
  (2)  011764  000004                TST64:  SCOPE
  (1)  011766  012737  000001  001160        MOV    #1,$TIMES           ;;DO 1 ITERATION
 1216  011774  012737  012022  001110        MOV    #1$,$LPERR          ;LOAD ERROR RETURN ADDRESS
 1217  012002  005737  001432               TST    TPAB                ;TEST IF MNCTP IS AT AMBIENT TEMP
 1218  012006  001513                        BEQ    TST65               ;;BR IF NOT AT AMBIENT
 1219  012010  005737  001176               TST    $PASS               ;TEST IF FIRST PASS
 1220  012014  001510                        BEQ    TST65               ;;DONT RUN OFFSET TEST ON FIRST PASS-TIN CANS NEED TIME
 1221  012016  012702  060166               MOV    #CHTABL+10,R2       ;LOAD POINTER TO CHANNEL TYPE BUFFER
 1222  012022  105712               1$:      TSTB   (R2)                ;TEST IF EXISTANT CHANNEL
 1223  012024  001504                        BEQ    TST65               ;;BR IF NO MORE
 1224  012026  100101                        BPL    6$                  ;BR IF DONT TEST THIS CHANNEL
 1225  012030  111237  023176               MOVB   (R2),CHA            ;GET CHANNEL TYPE
 1226  012034  042737  177700  023176        BIC    #177700,CHA         ;MASK OFF OTHER BITS
 1227  012042  022737  000004  023176        CMP    #4,CHA              ;TEST IF MNCTP CHANNEL
 1228  012050  001070                        BNE    6$                  ;BR IF NOT MNCTP CHANNEL
 1229  012052  010237  001556               MOV    R2,CHANL            ;COPY CHANNEL INDEX
 1230  012056  162737  060156  001556        SUB    #CHTABL,CHANL       ;REMOVE STARTING VALUE
 1231  012064  005003                        CLR    R3                  ;PRIME TO GAIN/TP TYPE = 0
 1232  012066  005004                        CLR    R4                  ;INITILIZE INDEX VALUE
 1233  012070  016437  037522  012206 2$:    MOV    TPOFFK(R4),3$       ;LOAD EXPECTED CONVERTED VALUE FROM TABLE
 1234  012076  016437  037562  012210        MOV    TPOFFS(R4),4$       ;LOAD EXPECTED CONVERTED SPREAD FROM TABLE
 1235  012104  112777  000077  167346        MOVB   #77,@ADST1          ;LOAD
 1236  012112  110377  167342               MOVB   R3,@ADST1           :        GAIN AND TC
 1237  012116  113777  001556  167334        MOVB   CHANL,@ADST1        :               TYPE
 1238  012124  010337  013546               MOV    R3,TPVAL            ;SAVE GAIN/TC TYPE IF AN ERROR IS DETECTED
 1239  012130  005000                        CLR    R0                  ;PRIME THE AVERAGE VALUE
 1240  012132  012701  000020               MOV    #16.,R1             ;LOAD COUNTER
 1241  012136  105277  167314      10$:      INCB   @STREG              ;START CONVERSION
 1242  012142  105777  167310      11$:      TSTB   @STREG              ;WAIT FOR READY
 1243  012146  100375                        BPL    11$
 1244  012150  067700  167306               ADD    @ADBUFF,R0          ;UPDATE AVERAGE
 1245  012154  005301                        DEC    R1                  ;FINISHED?
 1246  012156  001367                        BNE    10$                 ;BR IF NOT
 1247  012160  000241                        CLC                        ;ENSURE CLEAR "C" BIT
 1248  012162  006000                        ROR    R0                  ;MOVE RIGHT
 1249  012164  010037  001540               MOV    R0,TEMP             ;SAVE FOR LATER
 1250  012170  013700  015334               MOV    OFFSAV,R0           ;GET MNCAC OFFSET VALUE
 1253  012174  006300                        ASL    R0                  ;MOVE LEFT
  (1)  012176  006300                        ASL    R0                  ;MOVE LEFT
  (1)  012200  006300                        ASL    R0                  ;MOVE LEFT
 1254  012202  004537  036462      77$:      JSR    R5,COMPAR           ;COMPARE RESULTS
 1255  012206  000000            3$:          0                          ;EXPECTED VALUE
 1256  012210  000000            4$:          0                          ;TOLERANCE
 1257  012212  104015                        ERROR  15                  ;MNCTP OFFSET VALUE OUT OF RANGE
 1258  012214  022703  000017      5$:       CMP    #17,R3              ;LAST GAIN/TYPE
 1259  012220  001404                        BEQ    6$                  ;;BR IF END GAIN-TYPE
 1260  012222  005203                        INC    R3                  ;UPDATE GAIN-TYPE
 1261  012224  062704  000002               ADD    #2,R4               ;UPDATE INDEX VALUE
 1262  012230  000717                        BR     2$
 1263  012232  105722               6$:      TSTB   (R2)+               ;BUMP THE CHANNEL POINTER
 1264  012234  000672                        BR     1$                  ;TRY NEXT CHANNEL
```

```
 1266                                      ;;***********************************************************************
  (3)                                      ;*TEST 65          TEST RAMP RANGE ON MNCAD CH3
  (3)                                      ;;***********************************************************************
  (2)    012236  000004             TST65:  SCOPE
  (1)    012240  012737  000001  001160     MOV     #1,$TIMES           ;;DO 1 ITERATION
 1267    012246  012703  007777             MOV     #7777,R3            ;INIT R3 VALUE
 1268    012252  005004                      CLR     R4                  ;AND R4
 1269    012254  012777  001400  167174      MOV     #1400,@STREG        ;SETUP FOR CH3
 1270    012262  012702  047040             MOV     #20000.,R2          ;SETUP FOR 20,000 CONVERSIONS
 1271    012266  105277  167164      1$:     INCB    @STREG
 1272    012272  105777  167160      2$:     TSTB    @STREG
 1273    012276  100375                      BPL     2$
 1274    012300  027704  167156             CMP     @ADBUFF,R4
 1275    012304  003402                      BLE     3$
 1276    012306  017704  167150             MOV     @ADBUFF,R4          ;HIT A NEW HIGH
 1277    012312  027703  167144      3$:     CMP     @ADBUFF,R3
 1278    012316  002002                      BGE     4$
 1279    012320  017703  167136             MOV     @ADBUFF,R3          ;HIT A NEW LOW
 1280    012324  005302             4$:     DEC     R2
 1281    012326  001357                      BNE     1$
 1282    012330  010337  001540             MOV     R3,TEMP
 1283    012334  004537  036462             JSR     R5,COMPAR
 1284    012340  000000                      0
 1285    012342  037364                      V0
 1286    012344  104004                      ERROR   4                   ;RAMP DIDN'T REACH LOW END OF RANGE
 1287    012346  010437  001540             MOV     R4,TEMP
 1288    012352  004537  036462             JSR     R5,COMPAR
 1289    012356  007777                      7777
 1290    012360  037364                      V0
 1291    012362  104004                      ERROR   4                   ;RAMP DIDN'T REACH HIGH END OF RANGE
 1292
 1293                                      ;;***********************************************************************
  (3)                                      ;*TEST 66          NOISE TEST, 1 EDGE      (SINGLE ENDED, MNCTP AND MNCAG CHANNELS ONLY)
  (3)                                      ;;***********************************************************************
  (2)    012364  000004             TST66:  SCOPE
  (1)    012366  012737  000001  001160     MOV     #1,$TIMES           ;;DO 1 ITERATION
 1294    012374  005037  001530             CLR     WIDE                ;CLEAR ENTRY FLAG
 1295    012400  004737  012410             JSR     PC,NOITST           ;RUN NOISE TEST
 1296    012404  000137  013550             JMP     NOIJMP              ;NEXT TEST
```

```
1298                                      ;MAJOR SUBROUTINE THAT DOES THE NOISE TESTING
1299  012410  005737  001624     NOITST:  TST    QUIET           ;TEST IF SHORT TYPEOUT MODE
1300  012414  001012                      BNE    1$              ;BR IF YES AND BYPASS SOME TYPEOUT
1301  012416  104401  041252              TYPE   .NOIMSG
1302  012422  004737  055712              JSR    PC,WHICHU       ;DETERMINE UNIT #
1303  012426  013746  001622              MOV    UNITBD,-(SP)
1304  012432  104403                       TYPOS                 ;TELL OPER.
1305  012434     001     000              .BYTE  1,0
1306  012436  104401  001165              TYPE   .SCRLF
1307  012442  005737  001530     1$:      TST    WIDE            ;TEST IF MANUAL ENTRY
1308  012446  001010                      BNE    NOITS1          ;BR IF MANUAL
1309  012450  005037  001556              CLR    CHANL           ;INITLIZE TO CHAN 0
1310  012454  005737  022132              TST    WFAM            ;RUNNING MNCAM'S ON THE TESTER
1311  012460  001403                      BEQ    NOITS1          ;;BR IF NOT
1312  012462  012737  000020  001556      MOV    #20,CHANL       ; TESTING AM
1313                                      ;DETERMINE IF CHANNEL IS TO BE TESTED
1314  012470  013700  001556     NOITS1:  MOV    CHANL,R0        ;LOAD R0
1315  012474  005737  001530              TST    WIDE            ;TEST ENTRY FLAG
1316  012500  001007                      BNE    2$              ;BR IF MANUAL ENTRY
1317  012502  105760  060156              TSTB   CHTABL(R0)      ;TEST IF EXISTANT CHANNEL
1318  012506  001001                      BNE    1$              ;BR IF DONE
1319  012510  000207                      RTS    PC              ;EXIT
1320  012512  100402             1$:      BMI    2$              ;BR IF OPER SAID TO TEST THIS CHANNEL
1321  012514  000137  013500              JMP    UPCHAN
1322  012520  016037  060156  013544 2$:  MOV    CHTABL(R0),CHANIS        ;GET CHANNEL TYPE
1323  012526  042737  177700  013544      BIC    #177700,CHANIS  ;MASK OFF BITS
1324  012534  022737  000003  013544      CMP    #3,CHANIS       ;TEST IF MNCAG CHANNEL
1325  012542  001152                      BNE    4$              ;BR IF NOT
1326
1327                                      ;CHANNEL IS A MNCAG
1328  012544  005737  001624              TST    QUIET           ;TEST IF QUIET TYPEOUT MODE
1329  012550  001002                      BNE    11$             ;BR IF YES
1330  012552  104401  045346              TYPE   .GANP5          ;TELL OPER. THAT GAIN OF .5
1331  012556  112777  000077  166674 11$: MOVB   #77,@ADST1      ;ESC.
1332  012564  112777  000000  166666      MOVB   #0,@ADST1       ;LOAD GAIN BITS TO 0
1333  012572  113777  001556  166660      MOVB   CHANL,@ADST1    ;SELECT CHANNEL
1334
1335  012600  004537  015010              JSR    R5,RMSPEK       ;DO RMS NOISE TESTING
1336  012604     020     124              .BYTE  16.,84.         ;RMS VALUES
1337  012606  046551                      .WORD  RMSNOI          ;RMS MESSAGE TEXT POINTER
1338  012610  037650                       VNRAG0                ;POINTER TO TOLERANCE
1339
1340  012612  004537  015010              JSR    R5,RMSPEK       ;DO PEAK NOISE TESTING
1341  012616     001     143              .BYTE  1.,99.          ;PEAK VALUES
1342  012620  046565                      .WORD  PKNOI           ;PEAK MESSAGE TEXT POINTER
1343  012622  037652                       VNPAG0                ;POINTER TO TOLERANCE
1344
1345  012624  005737  001624              TST    QUIET           ;TEST IF QUIET MODE
1346  012630  001002                      BNE    12$             ;BR IF YES
1347  012632  104401  045401              TYPE   .GAN5P          ;TELL OPERATOR GAIN IS NOW 5.0
1348  012636  112777  000077  166614 12$: MOVB   #77,@ADST1      ;SELECT
1349  012644  112777  000001  166606      MOVB   #01,@ADST1      ;      GAIN
1350  012652  113777  001556  166600      MOVB   CHANL,@ADST1    ;          OF 5.
1351  012660  004537  015010              JSR    R5,RMSPEK       ;DO RMS TESTING
1352  012664     020     124              .BYTE  16.,84.         ;RMS VALUES
1353  012666  046551                      .WORD  RMSNOI          ;RMS MESSAGE TEXT POINTER
```

```
 1354   012670  037654                        VNRAG1                    ;POINTER TO TOLERANCE
 1355
 1356   012672  004537  015010                JSR       R5,RMSPEK       ;DO PEAK NOISE TESTING
 1357   012676     001     143                .BYTE     1.,99.          ;PEAK VALUES
 1358   012700  046565                        .WORD     PKNOI           ;PEAK MESSAGE TEXT POINTER
 1359   012702  037656                        VNPAG1                    ;POINTER TO TOLERANCE
 1360
 1361   012704  005737  001624                TST       QUIET           ;TEST IF QUIET MODE
 1362   012710  001002                        BNE       13$             ;BR IF YES
 1363   012712  104401  045434                TYPE      ,GAN5D          ;TELL OPERATOR GAIN IS NOW 50.
 1364   012716  112777  000077  166534  13$:  MOVB      #77,@ADST1      ;SELECT
 1365   012724  112777  000002  166526        MOVB      #2,@ADST1       ;       GAIN
 1366   012732  113777  001556  166520        MOVB      CHANL,@ADST1    ;            OF 50.
 1367   012740  013737  037660  024170        MOV       VRAG2A,AGCHRA   ;LOAD MSW OF RMS LIMIT
 1368   012746  013737  037662  024172        MOV       VRAG2B,AGCHRB   ;LOAD LSW OF RMS LIMIT
 1369   012754  013737  037664  024322        MOV       VPAG2A,AGCHPA   ;LOAD MSW OF PEAK LIMIT
 1370   012762  013737  037666  024324        MOV       VPAG2B,AGCHPB   ;LOAD LSW OR PEAK LIMIT
 1371   012770  004737  023202                JSR       PC,PRI4A        ;DO NOISE TESTING USING DIFFERENT METHOD
 1372
 1373   012774  005737  001624                TST       QUIET           ;TEST IF QUIET MODE
 1374   013000  001002                        BNE       14$             ;BR IF YES
 1375   013002  104401  045470                TYPE      ,GAN5T          ;TELL OPERATOR GAIN IS NOW 500
 1376   013006  112777  000077  166444  14$:  MOVB      #77,@ADST1      ;SELECT
 1377   013014  112777  000003  166436        MOVB      #3,@ADST1       ;       GAIN
 1378   013022  113777  001556  166430        MOVB      CHANL,@ADST1    ;            OF 500
 1379   013030  013737  037670  024170        MOV       VRAG3A,AGCHRA   ;LOAD MSW OF RMS LIMIT
 1380   013036  013737  037672  024172        MOV       VRAG3B,AGCHRB   ;LOAD LSW OF RMS LIMIT
 1381   013044  013737  037674  024322        MOV       VPAG3A,AGCHPA   ;LOAD MSW OF PEAK LIMIT
 1382   013052  013737  037676  024324        MOV       VPAG3B,AGCHPB   ;LOAD LSW OF PEAK LIMIT
 1383   013060  004737  023202                JSR       PC,PRI4A        ;DO NOISE TESTING USING DIFFERENT METHOD
 1384   013064  000137  013500                JMP       UPCHAN          ;CHECK NEXT CHANNEL
 1385
 1386   013070  022737  000004  013544   4$:  CMP       #4,CHANIS       ;IS THE CHANNEL A MNCTP?
 1387   013076  001161                        BNE       5$              ;BR IF NOT
 1388                                          ;CHANNEL IS A MNCTP
 1389   013100  005737  001624                TST       QUIET           ;TEST IF QUIET MODE
 1390   013104  001002                        BNE       15$             ;BR IF YES
 1391   013106  104401  045525                TYPE      ,TPGN00         ;TELL OPERATOR ''GAIN-TYPE'' = 0
 1392   013112  112777  000077  166340  15$:  MOVB      #77,@ADST1      ;ESC.
 1393   013120  112777  000000  166332        MOVB      #0,@ADST1       ;LOAD GAIN AND TYPE TO 0
 1394   013126  113777  001556  166324        MOVB      CHANL,@ADST1    ;SELECT CHANNEL
 1395   013134  013737  037706  024170        MOV       VRTP0A,AGCHRA   ;LOAD MSW OF RMS LIMIT
 1396   013142  013737  037710  024172        MOV       VRTP0B,AGCHRB   ;LOAD LSW OF RMS LIMIT
 1397   013150  013737  037712  024322        MOV       VPTP0A,AGCHPA   ;LOAD MSW OF PEAK LIMIT
 1398   013156  013737  037714  024324        MOV       VPTP0B,AGCHPB   ;LOAD LSW OF PEAK LIMIT
 1399   013164  004737  023202                JSR       PC,PRI4A        ;DO NOISE TEST NOW
 1400
 1401   013170  005737  001624                TST       QUIET           ;TEST IF QUIET MODE
 1402   013174  001002                        BNE       16$             ;BR IF YES
 1403   013176  104401  045564                TYPE      ,TPGN01         ;TELL OPERATOR ''GAIN-TYPE'' = 1
 1404   013202  112777  000077  166250  16$:  MOVB      #77,@ADST1      ;SELECT
 1405   013210  112777  000001  166242        MOVB      #01,@ADST1      ;       GAIN AND TYPE
 1406   013216  113777  001556  166234        MOVB      CHANL,@ADST1    ;
 1407   013224  013737  037716  024170        MOV       VRTP1A,AGCHRA   ;LOAD MSW OF RMS LIMIT
 1408   013232  013737  037720  024172        MOV       VRTP1B,AGCHRB   ;LOAD LSW OF RMS LIMIT
 1409   013240  013737  037722  024322        MOV       VPTP1A,AGCHPA   ;LOAD MSW OF PEAK LIMIT
```

```
1410  013246  013737  037724  024324         MOV    VPTP1B,AGCHPB     ;LOAD LSW OF PEAK LIMIT
1411  013254  004737  023202                 JSR    PC,PRI4A          ;DO NOISE TEST NOW
1412
1413  013260  005737  001624                 TST    QUIET             ;TEST IF QUIET MODE
1414  013264  001002                          BNE    17$               ;BR IF YES
1415  013266  104401  045623                 TYPE   ,TPGN10           ;TELL OPERATOR ''GAIN-TYPE'' = 10
1416  013272  112777  000077  166160  17$:   MOVB   #77,@ADST1        ;SELECT
1417  013300  112777  000010  166152         MOVB   #10,@ADST1        ;          GAIN AND TYPE
1418  013306  113777  001556  166144         MOVB   CHANL,@ADST1      ;
1419  013314  013737  037726  024170         MOV    VRTP2A,AGCHRA     ;LOAD MSW OF RMS LIMIT
1420  013322  013737  037730  024172         MOV    VRTP2B,AGCHRB     ;LOAD LSW OF RMS LIMIT
1421  013330  013737  037732  024322         MOV    VPTP2A,AGCHPA     ;LOAD MSW OF PEAK LIMIT
1422  013336  013737  037734  024324         MOV    VPTP2B,AGCHPB     ;LOAD LSW OR PEAK LIMIT
1423  013344  004737  023202                 JSR    PC,PRI4A          ;DO NOISE TESTING USING DIFFERENT METHOD
1424
1425  013350  005737  001624                 TST    QUIET             ;TEST IF QUIET MODE
1426  013354  001002                          BNE    18$               ;BR IF YES
1427  013356  104401  045662                 TYPE   ,TPGN11           ;TELL OPERATOR ''GAIN-TYPE'' = 11
1428  013362  112777  000077  166070  18$:   MOVB   #77,@ADST1        ;SELECT
1429  013370  112777  000011  166062         MOVB   #11,@ADST1        ;          GAIN AND TYPE
1430  013376  113777  001556  166054         MOVB   CHANL,@ADST1      ;
1431  013404  013737  037736  024170         MOV    VRTP3A,AGCHRA     ;LOAD MSW OF RMS LIMIT
1432  013412  013737  037740  024172         MOV    VRTP3B,AGCHRB     ;LOAD LSW OF RMS LIMIT
1433  013420  013737  037742  024322         MOV    VPTP3A,AGCHPA     ;LOAD MSW OF PEAK LIMIT
1434  013426  013737  037744  024324         MOV    VPTP3B,AGCHPB     ;LOAD LSW OF PEAK LIMIT
1435  013434  004737  023202                 JSR    PC,PRI4A          ;DO NOISE TESTING USING DIFFERENT METHOD
1436  013440  000417                          BR     UPCHAN            ;CHECK NEXT CHANNEL
1437
1438                                          ;CHANNEL IS A MNCAD/MNCAM
1439  013442  004537  015010         5$:     JSR    R5,RMSPEK         ;DO RMS NOISE TESTING
1440  013446    020     124                  .BYTE  16.,84.           ;RMS VALUES
1441  013450  046551                         RMSNOI                   ;RMS MESSAGE TEXT POINTER
1442  013452  037644                         VNR                      ;POINTER TO TOLERANCE
1443
1444  013454  004537  015010                 JSR    R5,RMSPEK         ;DO PEAK NOISE TESTING
1445  013460    001     143                  .BYTE  1.,99.            ;PEAK VALUES
1446  013462  046565                         PKNOI                    ;PEAK MESSAGE TEXT POINTER
1447  013464  037646                         VNP                      ;POINTER TO TOLERANCE
1448  013466  005737  001624                 TST    QUIET             ;TEST IF QUIET MODE
1449  013472  001002                          BNE    UPCHAN            ;BR IF YES
1450  013474  104401  001165                 TYPE   ,$CRLF
1451
1452                                          ;NOW UPDATE CHANNEL NUMBER AND DETERMINE IF MORE CHANNELS ARE TO BE TESTED
1453  013500  005737  001530         UPCHAN: TST    WIDE              ;CHECK ENTRY FLAG
1454  013504  001016                          BNE    3$                ;BR IF MANUAL ENTRY
1455  013506  005237  001556                 INC    CHANL             ;UPDATE CHANNEL NUMBER
1456  013512  022737  000003  001556         CMP    #3,CHANL          ;CHANNEL 3 (RAMP CHANNEL)?
1457  013520  001404                          BEQ    1$                ;;YES
1458  013522  022737  000007  001556         CMP    #7,CHANL          ;CHANNEL 7 (EDC INPUT CHANNEL)?
1459  013530  001002                          BNE    2$                ;;NO
1460  013532  005237  001556         1$:     INC    CHANL             ;CHANNELS 3 AND 7 ARE SKIPED
1461  013536  000137  012470         2$:     JMP    NOITS1            ;NO, CONTINUE TESTING
1462  013542  000207               3$:     RTS    PC                ;EXIT
1463  013544  000000               CHANIS: 0                        ;CURRENT CHANNEL TYPE
1464  013546  000000               TPVAL:  0                        ;VALUE OF THE MNCTP ''GAIN-TYPE'' REGISTER
```

```
1466
1467  013550                          NOIJMP:
1468                                   ;;***********************************************************
 (3)                                  ;;*TEST 67       MNCAD INTERCHANNEL SETTLING TEST, 1 EDGE
 (3)                                  ;;***********************************************************
 (2)  013550  000004                  TST67:  SCOPE
 (1)  013552  012737  000001  001160          MOV     #1,$TIMES        ;;DO 1 ITERATION
1469  013560  005737  001624                  TST     QUIET            ;TEST IF QUIET MODE
1470  013564  001012                          BNE     2$               ;BR IF YES
1471  013566  104401  041302                  TYPE    .SETMSG          ;TYPE ''SETTLING TEST''
1472  013572  004737  055712                  JSR     PC,WHICHU        ;DETERMINE THE UNIT #
1473  013576  013746  001622                  MOV     UNITBD,-(SP)     ;SAVE IT
1474  013602  104403                          TYPOS                    ;TYPE IT
1475  013604     001     000                  .BYTE   1,0
1476  013606  104401  001165                  TYPE    .$CRLF
1477  013612  012737  000001  001544  2$:     MOV     #1,CH1           ;LOAD INITIAL CHANNEL NUMBER
1478  013620  012737  000002  001546          MOV     #2,CH2           ;
1479  013626  004737  013662                  JSR     PC,SETTLE        ;RUN TEST ON CH 1-2
1480  013632  005737  022132                  TST     WFAM             ;RUNNING MNCAM ON TESTER ?
1481  013636  001410                          BEQ     1$               ;BR IF NOT
1482  013640  012737  000024  001544          MOV     #24,CH1          ;GET MUX CHANNEL INCASE TESTING MNCAM
1483  013646  012737  000025  001546          MOV     #25,CH2          ;GET NEXT CHANNEL
1484  013654  004737  013662                  JSR     PC,SETTLE        ;RUN TEST ON MNCAM CH 24-25
1485  013660                          1$:
 (2)  013660  000461                          BR      TST70            ;;
```

```
1487                                        ;SUBROUTINE TO DO THE SETTLING BETWEEN TWO CHANNELS
1488  013662  005037  014022       SETTLE: CLR    20$             ;CLEAR RETRY COUNT
1489  013666  005237  014022       1$:     INC    20$             ;INCREMENT COUNT
1490  013672  022737  000006  014022        CMP    #6,20$          ;IS COUNT = 6?
1491  013700  001444                        BEQ    3$              ;;YES
1492  013702  013737  001546  001556        MOV    CH2,CHANL
1493  013710  004537  036340                JSR    R5,CONVTC       ;GET EDGE VALUES
1494  013714  013737  001540  001574        MOV    TEMP,EDGE       ;SET UP EDGE VALUE
1495  013722  005002                        CLR    R2
1496  013724  004737  034002                JSR    PC,SET1A        ;SCALING = .02 LSB
1497  013730  000756                        BR     1$              ;ERROR RECOVERY JUMP
1498  013732  004737  034002                JSR    PC,SET1A        ;MAKE IT .01 LSB
1499  013736  000753                        BR     1$              ;ERROR RECOVERY JUMP
1500  013740  005702                        TST    R2              ;TEST RESULTS
1501  013742  100001                        BPL    2$
1502  013744  005402                        NEG    R2              ;MAKE IT POSITIVE
1503  013746  010204               2$:      MOV    R2,R4
1504  013750  012737  000001  034130        MOV    #1,EDGFLG
1505  013756  004737  033624                JSR    PC,TYPSET       ;TYPE SETTLING INFORMATION
1506  013762  023737  001546  001544        CMP    CH2,CH1         ;DONE?
1507  013770  103413                        BLO    4$              ;;YES
1508  013772  013702  001544                MOV    CH1,R2          ;SETTLE THE OTHER WAY
1509  013776  013737  001546  001544        MOV    CH2,CH1
1510  014004  010237  001546                MOV    R2,CH2
1511  014010  000724                        BR     SETTLE          ;;
1512  014012  012702  000377       3$:      MOV    #255.,R2               ;SET SETTLING TO MAX ERROR
1513  014016  000753                        BR     2$              ;;
1514  014020  000207               4$:      RTS    PC              ;EXIT
1515  014022  000000               20$:     0
```

I 5

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 31
CVMNAC.P11   10-JUL-81 14:30   T67   MNCAD INTERCHANNEL SETTLING TEST, 1 EDGE   .   SEQ 0060

```
 1517
 1518
  (3)            ;;*************************************************************
  (3)            ;*TEST 70        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST (CHANNEL 3 ONLY AFTER
  (2)   014024  000004        ;;*************************************************************
  (1)   014026  012737  000001  001160  TST70:  SCOPE
 1519  014034  105737  060161              MOV     #1,$TIMES           ;;DO 1 ITERATION
 1520  014040  100014                      TSTB    CHTABL+3            ;TESTING CHANNEL 3?
 1521  014042  022737  000001  001176      BPL     TST71              ;;BR IF NOT
 1522  014050  001010                      CMP     #1,$PASS           ;TEST IF THE SECOND PASS THRU THE PROGRAM
 1523  014052  005737  001624              BNE     TST71              ;;BR IF NOT THE SECOND PASS
 1524  014056  001005                      TST     QUIET              ;TEST IF QUIET MODE
 1525  014060  012737  000003  023176      BNE     TST71              ;;BR IF YES
 1526  014066  004737  034370              MOV     #3,CHA             ;LOAD CHANNEL TO RUN ON
                                           JSR     PC,DIFLIN          ;RUN DIF LIN AND REL ACC ON CH 3
 1527
 1528           ;;*************************************************************
  (3)           ;*TEST 71        END OF WRAPAROUND ANALOG TESTS
  (3)           ;;*************************************************************
  (2)   014072  000004        TST71:  SCOPE
  (1)   014074  012737  000001  001160      MOV     #1,$TIMES          ;;DO 1 ITERATION
 1529  014102  000207                      RTS     PC                 ;RETURN TO TEST SECTION
 1530
 1531           ;SUBROUTINE TO LOAD THE ''GAIN OR TP TYPE'' VALUE INTO ''CH1 AND CH2''
 1532  014104  112777  000077  165346  LDSETG: MOVB   #77,@ADST1       ;START ESCAPE SEQUENCE
 1533  014112  112577  165342              MOVB    (R5)+,@ADST1       ;LOAD ''GAIN OR TYPE''
 1534  014116  113777  001544  165334      MOVB    CH1,@ADST1         ;LOAD CHANNEL
 1535  014124  112777  000077  165326      MOVB    #77,@ADST1         ;START ESCAPE SEQUENCE
 1536  014132  112577  165322              MOVB    (R5)+,@ADST1       ;LOAD ''GAIN OR TYPE''
 1537  014136  113777  001546  165314      MOVB    CH2,@ADST1         ;LOAD CHANNEL
 1538  014144  000205                      RTS     R5                 ;EXIT
```

```
 1540
-1541                                    ;SUBROUTINE TO CHECK IF CHANNEL IN R0 IS AN "AG" CHANNEL
 1542  014146  116037  060156  014172  CHKAGC: MOVB    CHTABL(R0),10$   ;GET CHANNEL TYPE
 1543  014154  042737  177600  014172          BIC     #177600,10$     ;CLEAR OFF BITS
 1544  014162  122737  000003  014172          CMPB    #3,10$          ;TEST IF MNCAG CHANNEL
 1545  014170  000207                          RTS     PC              ;EXIT
 1546  014172  000000                  10$:    0
 1547                                    ;SUBROUTINE TO CHECK IF CHANNEL IN R0 IS AN "TP" CHANNEL
 1548  014174  116037  060156  014220  CHKTCC: MOVB    CHTABL(R0),10$   ;GET CHANNEL TYPE
 1549  014202  042737  177600  014220          BIC     #177600,10$     ;CLEAR OFF OTHER BITS
 1550  014210  122737  000004  014220          CMPB    #4,10$          ;TEST IF MNCTP CHANNEL
 1551  014216  000207                          RTS     PC              ;EXIT
 1552  014220  000000                  10$:    0
 1553                                    ;SUBROUTINE TO LOAD A GAIN OF "01" INTO EACH CHANNEL 10-77
 1554  014222  010146                  LD01CH: MOV     R1,-(SP)
 1555  014224  010246                          MOV     R2,-(SP)
 1556  014226  013702  001460                  MOV     ADST1,R2        ;LOAD ADDRESS POINTER
 1557  014232  012701  000010                  MOV     #10,R1          ;LOAD INITIAL CHANNEL
 1558  014236  112712  000077          1$:     MOVB    #77,(R2)        ;LOAD 'ESCAPE'
 1559  014242  112712  000001                  MOVB    #1,(R2)         ;LOAD GAIN = 01
 1560  014246  110112                          MOVB    R1,(R2)         ;LOAD CHANNEL #
 1561  014250  005201                          INC     R1              ;UPDATE CHANNEL #
 1562  014252  022701  000100                  CMP     #100,R1         ;TEST IF LAST CHANNEL
 1563  014256  001367                          BNE     1$              ;BR IF NOT LAST CHANNEL
 1564  014260  012602                          MOV     (SP)+,R2
 1565  014262  012601                          MOV     (SP)+,R1
 1566  014264  000207                          RTS     PC              ;EXIT
 1567
 1568                                    ;;SUBROUTINE FOR LOGIC TESTS
 1569  014266  013777  001124  165162  TESTIT: MOV     $GDDAT,@STREG    ;LOAD EXPECTED DATA INTO REGISTER
 1570  014274  017737  165156  001126  TEST:   MOV     @STREG,$BDDAT    ;READ ACTUAL REGISTER
 1571  014302  023737  001124  001126          CMP     $GDDAT,$BDDAT    ;COMPARE RESULTS
 1572  014310  001002                          BNE     RETERR          ;RETURN EXIT
 1573  014312  062716  000002                  ADD     #2,(SP)         ;CORRECT EXIT BUMPS ENTRY BY 2
 1574  014316  000002                  RETERR: RTI                     ;EXIT
```

K 5

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 33
CVMNAC.P11   10-JUL-81 14:30   T71   END OF WRAPAROUND ANALOG TESTS   SEQ 0062

```
1576                                      ;SUBROUTINE TO DO THE LOADING AND READING OF GAIN INFO
1577                                      ;         1ST.    LOAD CHANNEL 10-77 WITH GAIN = 01
1578                                      ;         2ND.    WRITE CHANNEL X GAIN TO = 10
1579                                      ;         3RD.    READ CHANNEL X GAIN AND EXPECT = 10
1580                                      ;         4TH.    READ CHANNEL 10-77 EXCEPT CH XX AND NON-PREAMP CHS.
1581
1582                                      ;DO 1ST STEP
1583    014320  004737  014222            CHKGAN: JSR     PC,LD01CH        ;LOAD GAIN BITS TO 01
1584    014324  012737  014332  001110            MOV     #1$,$LPERR       ;LOAD ERROR RETURN ADDRESS
1585
1586                                      ;DO 2ND STEP
1587    014332  112777  000077  165120    1$:      MOVB    #77,@ADST1       ;LOAD ''ESC''
1588    014340  112777  000002  165112             MOVB    #2,@ADST1        ;LOAD GAIN = 10
1589    014346  110077  165106                     MOVB    RO,@ADST1        ;LOAD CHANNEL XX
1590
1591                                      ;DO 3RD STEP
1592    014352  004737  014472                     JSR     PC,RDCHXY        ;READ CHANNEL IN RO
1593    014356  012737  020000  001124             MOV     #20000,$GDDAT    ;LOAD EXPECTED
1594    014364  023737  001124  001126             CMP     $GDDAT,$BDDAT    ;COMPARE TO EXPECTED
1595    014372  001403                             BEQ     2$               ;;BR IF SAME
1596    014374  010037  001556                     MOV     RO,CHANL         ;SAVE CHANNEL INFO
1597    014400  104012                             ERROR   12               ;GAIN ON CHANNEL FAILED TO LOAD
1598                                      ;NOW DO 4TH STEP
1599    014402  012700  000010            2$:      MOV     #10,RO           ;PRIME THE CHANNEL #
1600    014406  012737  014422  001110             MOV     #3$,$LPERR       ;LOAD ERROR RETURN ADDRESS
1601    014414  012737  010000  001124             MOV     #10000,$GDDAT    ;LOAD EXPECTED VALUE
1602    014422  020037  017764            3$:      CMP     RO,CHXX          ;TEST IF RO = CHXX
1603    014426  001414                             BEQ     4$               ;BR IF SAME
1604                                      ;TEST IF RO CHANNEL IS AN ''AG'' CHANNEL
1605    014430  004737  014146                     JSR     PC,CHKAGC
1606    014434  001011                             BNE     4$               ;BR IF NOT ''AG'' CHANNEL
1607    014436  004737  014472                     JSR     PC,RDCHXY        ;READ CHANNEL IN RO STATUS
1608    014442  023737  001124  001126             CMP     $GDDAT,$BDDAT    ;COMPARE
1609    014450  001403                             BEQ     4$               ;;BR IF SAME
1610    014452  010037  001556                     MOV     RO,CHANL         ;SAVE BAD CHANNEL INFO
1611    014456  104012                             ERROR   12               ;CHANNEL GAIN BITS CHANGED IN ERROR
1612    014460  005200                    4$:      INC     RO               ;UPDATE CHANNEL
1613    014462  022700  000100                     CMP     #100,RO          ;TEST IF MORE CHANNELS
1614    014466  001355                             BNE     3$               ;BR IF NONE
1615    014470  000207                             RTS     PC               ;EXIT
1616
1617                                      ;SUBROUTINE TO CONVERT CHANNEL IN RO
1618                                      ;RETURN STATUS IN $BDDAT
1619    014472  110077  164762            RDCHXY: MOVB    RO,@ADST1        ;LOAD MUX REG.
1620    014476  152777  000010  164752             BISB    #BIT3,@STREG     ;ENABLE STATUS INFO.
1621    014504  105277  164746                     INCB    @STREG           ;START CONVERSION
1622    014510  105777  164742            1$:      TSTB    @STREG           ;WAIT FOR DONE
1623    014514  100375                             BPL     1$
1624    014516  017737  164740  001126             MOV     @ADBUFF,$BDDAT   ;READ STATUS
1625    014524  042737  147777  001126             BIC     #147777,$BDDAT   ;MASK OFF A/D CONVERSION DATA
1626    014532  000207                             RTS     PC               ;EXIT
1627
```

```
1629                                     ;SUBROUTINE TO DO THE LOADING AND READING OF GAIN INFO
1630                                     ;          1ST.     LOAD CHANNEL 0-77 WITH GAIN = 0001
1631                                     ;          2ND.     WRITE CHANNEL X GAIN TO = 0010, 0100, OR 1000
1632                                     ;          3RD.     READ CHANNEL X GAIN AND EXPECT = 0010, 0100 OR 1000
1633                                     ;          4TH.     READ CHANNEL 0-77 EXCEPT CH XX AND NON-MNCTP CHS.
1634
1635                                     ;DO 1ST STEP
1636    014534  012737  000002  017770   CHKTCN: MOV    #2,GLD0          ;PRIME GAIN LOADING BIT ON CHXX
1637    014542  012737  020000  017772           MOV    #20000,GLD1      ;PRIME GAIN READ BACK EXPECTED VALUE
1638    014550  004737  014222           1$:     JSR    PC,LD01CH        ;LOAD GAIN BITS TO 01
1639    014554  013700  017764                   MOV    CHXX,R0          ;GET CHANNEL #
1640    014560  012737  014566  001110           MOV    #20$,$LPERR      ;LOAD ERROR RETURN
1641                                     ;DO 2ND STEP
1642    014566  112777  000077  164664   20$:    MOVB   #77,@ADST1       ;LOAD 'ESC'
1643    014574  113777  017770  164656           MOVB   GLD0,@ADST1      ;LOAD GAIN = 1000, 0100, OR 0010
1644    014602  110077  164652                   MOVB   R0,@ADST1        ;LOAD CHANNEL XX
1645                                     ;DO 3RD STEP
1646    014606  004737  014746                   JSR    PC,RDTCXY        ;READ CHANNEL IN R0
1647    014612  013737  017772  001124           MOV    GLD1,$GDDAT      ;LOAD EXPECTED
1648    014620  023737  001124  001126           CMP    $GDDAT,$BDDAT    ;COMPARE TO EXPECTED
1649    014626  001403                            BEQ    2$               ;;BR IF SAME
1650    014630  010037  001556                    MOV    R0,CHANL         ;SAVE CHANNEL INFO
1651    014634  104013                            ERROR  13               ;GAIN ON CHANNEL FAILED TO LCAD
1652                                     ;NOW DO 4TH STEP
1653    014636  012700  000010           2$:     MOV    #10,R0           ;PRIME THE CHANNEL #
1654    014642  012737  014656  001110           MOV    #3$,$LPERR       ;LOAD ERROR RETURN
1655    014650  012737  010000  001124           MOV    #10000,$GDDAT    ;LOAD EXPECTED VALUE
1656    014656  020037  017764           3$:     CMP    R0,CHXX          ;TEST IF R0 = CHXX
1657    014662  001414                            BEQ    4$               ;BR IF SAME
1658                                     ;TEST IF R0 CHANNEL IS AN 'TP' CHANNEL
1659    014664  004737  014174                    JSR    PC,CHKTCC
1660    014670  001011                            BNE    4$               ;BR IF NOT 'TP' CHANNEL
1661    014672  004737  014746                    JSR    PC,RDTCXY        ;READ CHANNEL IN R0 STATUS
1662    014676  023737  001124  001126           CMP    $GDDAT,$BDDAT    ;COMPARE
1663    014704  001403                            BEQ    4$               ;;BR IF SAME
1664    014706  010037  001556                    MOV    R0,CHANL         ;SAVE BAD CHANNEL INFO
1665    014712  104013                            ERROR  13               ;CHANNEL GAIN BITS CHANGED IN ERROR
1666    014714  005200                   4$:     INC    R0               ;UPDATE CHANNEL
1667    014716  022700  000100                    CMP    #100,R0          ;TEST IF MORE CHANNELS
1668    014722  001355                            BNE    3$               ;BR IF NONE
1669    014724  006337  017770                    ASL    GLD0             ;CHANGE GAIN DATA TO BE LOADED
1670    014730  006337  017772                    ASL    GLD1             ;CHANGE EXPECTED DATA B
1671    014734  022737  000020  017770           CMP    #20,GLD0         ;TEST IF DONE ALL BITS
1672    014742  001302                            BNE    1$               ;BR IF NOT FINISHED ALL BITS
1673    014744  000207                            RTS    PC               ;EXIT
1674                                     ;SUBROUTINE TO CONVERT CHANNEL IN R0
1675                                     ;RETURN MNCTP STATUS IN $BDDAT
1676    014746  110077  164506           RDTCXY: MOVB   R0,@ADST1        ;LOAD MUX
1677    014752  152777  000010  164476           BISB   #BIT3,@STREG     ;ENABLE STATUS BITS
1678    014760  105277  164472                    INCB   @STREG           ;CONVERT
1679    014764  105777  164466           1$:     TSTB   @STREG           ;WAIT FOR READY
1680    014770  100375                            BPL    1$
1681    014772  017737  164464  001126           MOV    @ADBUFF,$BDDAT   ;READ STATUS
1682    015000  042737  007777  001126           BIC    #7777,$BDDAT     ;MASK OFF A/D BITS
1683    015006  000207                            RTS    PC               ;EXIT
```

```
1685
1686                                        ;SUBROUTINE TO DO THE RMS AND PEAK NOISE TESTING
1687  015010  112537  015102    RMSPEK: MOVB   (R5)+,60$        ;GET 1 POINT
1688  015014  112537  015122            MOVB   (R5)+,61$        ;GET 2 POINT
1689  015020  012537  015166            MOV    (R5)+,62$        ;GET TEXT POINTER
1690  015024  013537  015244            MOV    @(R5)+,63$       ;GET TOLERANCE
1691  015030  012737  015042  015362    MOV    #1$,ERRADR       ;SET UP ERROR RETRY ADDRESS
1692  015036  005037  015250            CLR    65$              ;CLEAR RETRY COUNT
1693  015042  005237  015250    1$:     INC    65$              ;INCREMENT COUNT
1694  015046  022737  000006  015250    CMP    #6,65$           ;IS COUNT = 6?
1695  015054  001460                    BEQ    3$               ;;YES, CHANNEL TOO WIDE OR NOISY
1696  015056  013737  001556  001554    MOV    CHANL,DUMMY      ;LOAD DUMMY CHANNEL
1697  015064  004537  036340            JSR    R5,CONVTC        ;GET EDGE VALUE
1698  015070  013737  001540  001574    MOV    TEMP,EDGE        ;SET UP EDGE VALUE
1699  015076  004537  034170            JSR    R5,SARSUB        ;DO SAR ROUTINE AT 16%
1700  015102  000020            60$:    16.
1701  015104  004737  015336            JSR    PC,TSTDAC        ;CHECK VERNIER DAC SETTING
1702  015110  013737  001570  015246    MOV    DAC,64$          ;ADD RESULT TO RMS
1703  015116  004537  034170            JSR    R5,SARSUB        ;DO SAR ROUTINE AT 84%
1704  015122  000124            61$:    84.
1705  015124  004737  015336            JSR    PC,TSTDAC        ;CHECK VERNIER DAC SETTING
1706  015130  163737  001570  015246    SUB    DAC,64$          ;SUBTRACT RESULT FROM RMS
1707  015136  012737  000001  034130    MOV    #1,EDGFLG
1708  015144  005737  001624            TST    QUIET            ;TEST IF QUIET MODE
1709  015150  001405                    BEQ    2$               ;BR IF NOT
1710  015152  023737  015246  015244    CMP    64$,63$          ;TEST IF WITHIN LIMITS
1711  015160  003001                    BGT    2$               ;BR IF NOT
1712  015162  000427                    BR     5$               ;BR IF WITHIN
1713  015164  104401            2$:     TYPE
1714  015166  046551            62$:    RMSNOI                  ;TEXT POINTER
1715  015170  013702  015246            MOV    64$,R2
1716  015174  004737  036274            JSR    PC,TYPRP         ;TYPE RMS VALUES
1717  015200  023737  015246  015244    CMP    64$,63$          ;WITHIN LIMITS?
1718  015206  003007                    BGT    4$               ;NO
1719  015210  104401  046620            TYPE   ,OKMSG
1720  015214  000412                    BR     5$               ;;
1721  015216  012737  000377  015246  3$: MOV   #255.,64$        ;SET RMS TO MAX ERROR
1722  015224  000757                    BR     2$               ;;
1723  015226  104401  047505    4$:     TYPE   ,ERMSG
1724  015232  004737  055704            JSR    PC,WHICHV        ;INDICATE BAD UNIT
1725  015236  005237  001112            INC    $ERTTL           ;UPDATE ERROR TOTAL
1726  015242  000205            5$:     RTS    R5               ;EXIT
1727  015244  000000            63$:    0
1728  015246  000000            64$:    0
1729  015250  000000            65$:    0
```

```
1731
1732
1733                              ;SUBROUTINE TO FIND THE 50-50 EDGE OF THE INPUT SIGANL
1734
1735  015252  012737  004001  001574  OFFSET: MOV   #4001,EDGE                ;4000,4001 EDGE
1736  015260  004537  034170                  JSR   R5,SARSUB
1737  015264  000062                           50.
1738  015266  013737  001570  0C1540           MOV   DAC,TEMP
1739  015274  012737  004000  001574           MOV   #4000,EDGE              ;3777,4000 EDGE
1740  015302  004537  034170                   JSR   R5,SARSUB
1741  015306  000062                           50.
1742  015310  063737  001570  001540           ADD   DAC,TEMP
1743  015316  162737  000400  001540           SUB   #400,TEMP
1744  015324  013737  001540  015334           MOV   TEMP,OFFSAV            ;SAVE THE ANSWER
1745  015332  000207                           RTS   PC
1746  015334  000000          OFFSAV: 0
1747
1748
1749                              ; ROUTINE TO TEST DAC SETTING FROM SARSUB
1750                              ; JUMPS TO ADDRESS IN ERRADR IF DAC SETTING IS EITHER 0 OR 377
1751                              ; OTHERWISE RETURNS TO CALL+1
1752  015336  005737  001570  TSTDAC: TST   DAC                ;IS DAC = 0 ?
1753  015342  001405                   BEQ   1$                ;;YES
1754  015344  022737  000377  001570   CMP   #377,DAC          ;IS DAC = 377 ?
1755  015352  001401                   BEQ   1$                ;;YES
1756  015354  000207                   RTS   PC
1757  015356  005726          1$:      TST   (SP)+             ;POP CALL OFF STACK
1758  015360  000137                   JMP   @(PC)+            ;JUMP TO ADDRESS IN ERRADR
1759  015362  000000          ERRADR: 0
1760
1761                              ;SUBROUTINE TO HANDLE THE SINGLE ENDED-DIFFERENTIAL LOGIC TESTS
1762  015364  012537  015500  TSTSDF: MOV   (R5)+,10$          ;GET 1ST ARGUMENT
1763  015370  005737  001602          TST   WFTEST            ;USING THE TESTER ?
1764  015374  100014                   BPL   1$                ;BR IF NOT
1765  015376  005737  015500           TST   10$               ;TEST THE 1ST ARG.
1766  015402  001004                   BNE   23$               ;BR IF NON ZERO
1767  015404  000005                   RESET                   ;CLEAR THE BIT BY A BUS INIT
1768  015406  000240                   NOP
1769  015410  000240                   NOP
1770  015412  000403                   BR    24$
1771  015414  052777  000200  164102  23$: BIS  #BIT7,@DRVDOR ;SET THE BIT
1772  015422  004737  022144          24$: JSR  PC,STALL       ;ALLOW RELAY TO CHANGE
1773  015426  012537  001124          1$:  MOV  (R5)+,$GDDAT   ;GET 2ND ARG. <EXPECTED DATA>
1774  015432  012577  164020               MOV  (R5)+,@STREG   ;GET 3RD ARG. <CHANNEL TO USE>
1775  015436  105277  164014               INCB @STREG         ;START CONVERSION
1776  015442  105777  164010          2$:  TSTB @STREG         ;WAIT FOR DONE
1777  015446  100375                   BPL  2$
1778  015450  017737  164006  001126   MOV  @ADBUFF,$BDDAT     ;READ RESULT
1779  015456  042737  157777  001126   BIC  #157777,$BDDAT     ;MASK OFF OTHER BITS
1780  015464  023737  001124  001126   CMP  $GDDAT,$BDDAT      ;COMPARE
1781  015472  001401                   BEQ  3$                 ;;BR IF SAME
1782  015474  104001                   ERROR 1                 ;INCORRECT VALUE TO SINGLE ENDED-DIFFERENTIAL MODE
1783  015476  000205          3$:  RTS  R5                     ;EXIT
1784  015500  000000          10$: 0
```

```
 1786                                    ;SUBROUTINE TO INSERT ''+'' AND TYPE # ON THE STACK
 1787
 1788  015502  013702  001540    TOFF:   MOV     TEMP,R2
 1789  015506  100402                     BMI     1$              ;;IS THE NUMBER POSITIVE?
 1790  015510  104401  047272             TYPE    ,POSITV
 1791  015514  104416            1$:      TYPDC
 1792  015516  104401  050644             TYPE    ,MLSB           ;TYPE ASCIZ STRING
 1793  015522  000207                     RTS     PC
 1794
 1795                                    ;SUBROUTINE TO WAIT FOR OPERATOR'S ''RETURN'' THEN CHECK TOLERANCES
 1796
 1797  015524  005303            TCHK:   DEC     R3              ;DECREMENT COUNT
 1798  015526  001005                     BNE     1$              ;;
 1799  015530  012703  000005             MOV     #5,R3           ;RESET COUNT
 1800  015534  104401  001165             TYPE    ,$CRLF          ;TYPE A CARRIAGE RETURN AND LINE FEED
 1801  015540  000402                     BR      2$              ;;
 1802  015542  104401  046502    1$:      TYPE    ,SPACE          ;TYPE FOUR (4) SPACES
 1803  015546  005037  001572    2$:      CLR     DELAY           ;CLEAR DELAY
 1804  015552  005077  163366             CLR     a$TKS           ;CLEAR INTERRUPT ENABLE
 1805  015556  105777  163362    3$:      TSTB    a$TKS           ;IS KEYBOARD FLAG SET?
 1806  015562  100404                     BMI     4$              ;;YES
 1807  015564  005237  001572             INC     DELAY           ;IS DELAY ZERO?
 1808  015570  001372                     BNE     3$              ;;NO
 1809  015572  000416                     BR      6$              ;;
 1810  015574  005777  163346    4$:      TST     a$TKB           ;CLEAR FLAG
 1811  015600  012777  000100  163336     MOV     #100,a$TKS      ;SET INTERRUPT ENABLE
 1812  015606  004537  036462             JSR     R5,COMPAR       ;TEST LAST CONVERSION
 1813  015612. 000000                     0
 1814  015614  037370                     V10                     ;TOLERANCE .10 LSB
 1815  015616  000402                     BR      5$              ;;
 1816  015620  062716  000002             ADD     #2,(SP)         ;BUMP RETURN ADDRESS
 1817  015624  062716  000002    5$:      ADD     #2,(SP)         ;BUMP RETURN ADDRESS 2 WORDS
 1818  015630  000207            6$:      RTS     PC
```

```
1820                                    .SBTTL  MNCAD CALIBRATION SECTION
1821   015632  104401  046631   BEGINC: TYPE    ,CCHAN          ;ASK FOR CHANNEL
1822   015636  104413                   RDOCT                   ;READ CHANNEL NUMBER
1823   015640  012637  001556           MOV     (SP)+,CHANL     ;STORE CHANNEL NUMBER
1824   015644  013737  001556  001554   MOV     CHANL,DUMMY     ;LOAD DUMMY
1825   015652  104401  046717   1$:     TYPE    ,SEL            ;SELECT OFFSET OR GAIN ADJUST
1826   015656  104412                   RDLIN                   ;GET TEST
1827   015660  012600                   MOV     (SP)+,R0        ;MOVE POINTER TO R0
1828   015662  121027  000117           CMPB    (R0),#'O        ;IS IT 'O'?
1829   015666  001406                   BEQ     AJOFF           ;;YES, GO TO ADJUST OFFSET
1830   015670  121027  000107           CMPB    (R0),#'G        ;IS IT 'G'?
1831   015674  001430                   BEQ     AJGAIN          ;;YES, GO TO ADJUST GAIN
1832   015676  104401  001164           TYPE    ,$QUES          ;TYPE ''?''
1833   015702  000763                   BR      1$              ;;
1834
1835                                    ;SUBROUTINE TO CHECK OFFSET ADJUSTMENT VALUES
1836   015704  104401  047112   AJOFF:  TYPE    ,IGND           ;GROUND CHANNEL
1837   015710  104412                   RDLIN                   ;WAIT FOR CR
1838   015712  005726                   TST     (SP)+           ;POP 1 WORD OFF STACK
1839   015714  104401  047010   1$:     TYPE    ,XADJ           ;ADJUST MESSAGE
1840   015720  012703  000005           MOV     #5,R3           ;SET UP COUNT
1841   015724  004737  015252   2$:     JSR     PC,OFFSET       ;TEST AND TYPE OFFSET ERROR
1842   015730  004737  015502           JSR     PC,TOFF         ;TYPE OFFSET
1843   015734  004737  015524           JSR     PC,TCHK         ;CHECK FOR A CHARACTER AND DELAY
1844   015740  000771                   BR      2$              ;;
1845   015742  000402                   BR      3$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1846   015744  000137  001674           JMP     BEG2
1847   015750  104401  047505   3$:     TYPE    ,ERMSG          ;TELL OPER. 'ERROR'
1848   015754  000757                   BR      1$
1849                                    ;SUBROUTINE TO CHECK THE GAIN ADJUSTMENT
1850   015756  104401  047211   AJGAIN: TYPE    ,IVOLT          ;INPUT +5.115 VOLTS ON CHANNEL
1851   015762  104401  047151           TYPE    ,CRWR
1852   015766  104412                   RDLIN                   ;WAIT FOR CR
1853   015770  005726                   TST     (SP)+           ;POP 1 WORD OFF STACK
1854   015772  104401  047255   1$:     TYPE    ,YADJ           ;ADJUST MESSAGE
1855   015776  104401  047024           TYPE    ,MOLSB          ;TYPE '' FOR 0.00 LSB ERROR''
1856   016002  012703  000005           MOV     #5,R3           ;SET UP COUNT
1857   016006  012737  007777  001574 2$: MOV   #7777,EDGE      ;LOOK FOR 7776,7777 EDGE
1858   016014  004537  034170           JSR     R5,SARSUB
1859   016020  000062                   50.
1860   016022  013737  001570  001540   MOV     DAC,TEMP        ;SAVE DAC
1861   016030  012737  007776  001574   MOV     #7776,EDGE      ;LOOK FOR 7775,7776 EDGE
1862   016036  004537  034170           JSR     R5,SARSUB
1863   016042  000062                   50.
1864   016044  063737  001570  001540   ADD     DAC,TEMP        ;ADD RESULTS
1865   016052  162737  000400  001540   SUB     #400,TEMP       ;OFFSET RESULT
1866   016060  004737  015502           JSR     PC,TOFF         ;TYPE GAIN
1867   016064  004737  015524           JSR     PC,TCHK         ;CHECK FOR CHARACTER AND DELAY
1868   016070  000746                   BR      2$              ;;
1869   016072  000402                   BR      3$              ;;NOT WITHIN TOLLERANCE, TRY AGAIN
1870   016074  000137  001674           JMP     BEG2
1871   016100  104401  047505   3$:     TYPE    ,ERMSG          ;TELL OPER. 'ERROR'
1872   016104  000732                   BR      1$
```

```
1874                                    .SBTTL   SWITCH GAIN MANUAL INTERVENTION TEST
1875   016106   004737   033454  BEGINF: JSR     PC,FIXONE       ;ENSURE INITIAL BUS ADDRESS OF UNIT
1876   016112   104401   046631          TYPE    ,CCHAN          ;ASK FOR CHANNEL
1877   016116   104413                   RDOCT                   ;READ CHANNEL NUMBER
1878   016120   012600                   MOV     (SP)+,R0        ;GET CHANNEL NUMBER
1879   016122   010037   001556          MOV     R0,CHANL        ;LOAD CHANNEL FOR ERROR REPORT
1880   016126   000300                   SWAB    R0              ;PUT CHANNEL NUMBER IN HIGH BYTE
1881   016130   052700   000010          BIS     #BIT3,R0        ;SET STATUS ENABLE BIT
1882   016134   010077   163316          MOV     R0,@STREG       ;LOAD CHANNEL AND STATUS ENABLE
1883   016140   104401   044531          TYPE    ,SCM            ;ASK MODE BE SET TO CURRENT
1884   016144   104401   045232          TYPE    ,GHLF           ;ASK GAIN BE SET TO .5
1885   016150   012737   040000   001124 MOV     #BIT14,$GDDAT   ;SET UP EXPECTED
1886   016156   104417                   TESTID                  ;GO TEST FOR ID CODE
1887   016160   104011                   ERROR   11
1888   016162   104401   045253          TYPE    ,GAIN5          ;ASK GAIN BE SET TO 5
1889   016166   012737   050000   001124 MOV     #BIT14!BIT12,$GDDAT      ;LOAD EXPECTED
1890   016174   104417                   TESTID                  ;GO TEST ID CODE
1891   016176   104011                   ERROR   11
1892   016200   104401   045276          TYPE    ,GAIN50         ;ASK GAIN BE SET TO 50
1893   016204   012737   060000   001124 MOV     #BIT14!BIT13,$GDDAT      ;LOAD EXPECTED
1894   016212   104417                   TESTID                  ;GO TEST ID CODE
1895   016214   104011                   ERROR   11
1896   016216   104401   045321          TYPE    ,GAIN5M         ;ASK GAIN BE SET TO 500
1897   016222   012737   070000   001124 MOV     #BIT14!BIT13!BIT12,$GDDAT        ;LOAD EXPECTED
1898   016230   104417                   TESTID                  ;GO TEST ID CODE
1899   016232   104011                   ERROR   11
1900   016234   104401   045232          TYPE    ,GHLF           ;SET RANGE SWITCH
1901   016240   104401   044602          TYPE    ,SRM            ;ASK MODE BE SET TO RESISTANCE
1902   016244   012737   100000   001124 MOV     #100000,$GDDAT  ;LOAD EXPECTED VALUE
1903   016252   104417                   TESTID
1904   016254   104011                   ERROR   11              ;RESISTANCE MODE SWITCH VALUE IN ERROR
1905   016256   104401   044652          TYPE    ,SVM            ;ASK MODE BE SET TO VOLTS
1906   016262   012737   140000   001124 MOV     #140000,$GDDAT  ;LOAD EXPECTED VALUE
1907   016270   104417                   TESTID
1908   016272   104011                   ERROR   11              ;VOLTACE MODE SWITCH VALUE IN ERROR
1909   016274   104401   001165          TYPE    ,$CRLF
1910   016300   104401   044072          TYPE    ,SAGTST         ;TELL OPER. TO SET SWITCHES
1911   016304   104401   047151          TYPE    ,CRWR
1912   016310   104412                   RDLIN
1913   016312   005726                   TST     (SP)+           ;POP RETURN OFF STACK
1914   016314   104401   046423          TYPE    ,ENDTST         ;TELL OPER ''THATS ALL FOLKS''
1915   016320   000137   001674          JMP     BEG2
1916
1917   016324   104401   047151  TPRMP:  TYPE    ,CRWR           ;ASK FOR CR WHEN READY
1918   016330   104412                   RDLIN                   ;WAIT FOR CR
1919   016332   005726                   TST     (SP)+           ;POP 1 WORD OFF STACK
1920   016334   005277   163116          INC     @STREG          ;START A CONVERSION
1921   016340   105777   163112  1$:     TSTB    @STREG          ;WAIT TILL DONE
1922   016344   100375                   BPL     1$              ;
1923   016346   017737   163110   001126 MOV     @ADBUFF,$BDDAT  ;GET RESULTS
1924   016354   042737   007777   001126 BIC     #7777,$BDDAT    ;CLEAR CONVERTED VALUE
1925   016362   023737   001124   001126 CMP     $GDDAT,$BDDAT   ;IS ID RIGHT?
1926   016370   001002                   BNE     2$              ;;NO,TAKE ERROR RETURN
1927   016372   062716   000002          ADD     #2,(SP)         ;BUMP RETURN ADDRESS
1928   016376   000002          2$:      RTI
```

```
1930                                      .SBTTL  MNCAG TEST MODULE INTERACTIVE TESTS
1931  016400  004737  033454      BEGINT: JSR     PC,FIXONE            ;ENSURE CORRECT ADDRESSES
1932  016404  104401  001165              TYPE    ,$CRLF
1933  016410  104401  044072              TYPE    ,SAGTST              ;TELL OPER. TO SET AG TO 'P''
1934  016414  104401  046631              TYPE    ,CCHAN               ;GET CHANNEL NUMBER
1935  016420  104413                       RDOCT
1936  016422  012637  001544              MOV     (SP)+,CH1            ;GET CHANNEL # FROM OPER.
1937  016426  004737  017046              JSR     PC,CLRCHS            ;CONVERT EACH CHANNEL OF THIS MNCAG
1938                                ;FIRST - TEST MNCAG-TA HOLD LOGIC FOR THESE CHANNELS
1939  016432  004537  017300              JSR     R5,TSTHLD            ;TEST HOLD FOR 1ST CHANNEL OF THIS AG
1940  016436     000     005              .BYTE   0,5                  ;CHANNEL OFFSET, SWITCH NUMBER TO PUSH
1941  016440  004537  017300              JSR     R5,TSTHLD            ;              ..       2ND          ..
1942  016444     001     006              .BYTE   1,6                  ;              ..                    ..
1943  016446  004537  017300              JSR     R5,TSTHLD            ;              ..       3RD          ..
1944  016452     002     007              .BYTE   2,7
1945  016454  004537  017300              JSR     R5,TSTHLD            ;                       4TH
1946  016460     003     010              .BYTE   3,8.
1947
1948                                ;MNCAG PART 1
1949  016462  004537  016642              JSR     R5,TSETUP            ;GO DO THE WORK
1950  016466     002     003  002          .BYTE   2,3,2,3             ;FRONT PANEL EXPECTED CODE
      016471             003
1951  016472  045036                       .WORD   TXTP2               ;POS. OF TEST MODULE SWITCH
1952  016474     000     002              .BYTE   0,2                  ;GAIN, SPREAD
1953  016476  004002                       .WORD   4002                ;CHANNEL A - C EXPECTED VALUE
1954  016500     001     002              .BYTE   1,2                  ;GAIN, SPREAD
1955  016502  004024                       .WORD   4024                ;CHANNEL B - D EXPECTED VALUE
1956  016504     002     004              .BYTE   2,4                  ;GAIN, SPREAD
1957  016506  004310                       .WORD   4310                ;CHANNEL A - C EXPECTED VALUE
1958  016510     003     050              .BYTE   3,50                 ;GAIN, SPREAD
1959  016512  007720                       .WORD   7720                ;CHANNEL B - D EXPECTED VALUE
1960
1961                                ;MNCAG PART 2
1962  016514  004537  016642              JSR     R5,TSETUP            ;GO DO THE WORK
1963  016520     003     002  003          .BYTE   3,2,3,2             ;FRONT PANEL EXPECTED CODE
      016523             002
1964  016524  000000                       .WORD   0                   ;NO TEST MODULE CHANGES
1965  016526     000     002              .BYTE   0,2                  ;GAIN, SPREAD
1966  016530  004002                       .WORD   4002                ;CHANNEL A - C EXPECTED VALUE
1967  016532     001     002              .BYTE   1,2                  ;GAIN, SPREAD
1968  016534  004024                       .WORD   4024                ;CHANNEL B - D EXPECTED VALUE
1969  016536     002     004              .BYTE   2,4                  ;GAIN, SPREAD
1970  016540  004310                       .WORD   4310                ;CHANNEL A - C EXPECTED VALUE
1971  016542     003     050              .BYTE   3,50                 ;GAIN, SPREAD
1972  016544  007720                       .WORD   7720                ;CHANNEL B - D EXPECTED VALUE
1973
1974                                ;MNCAG PART 3
1975  016546  004537  016642              JSR     R5,TSETUP            ;GO DO THE WORK
1976  016552     001     002  001          .BYTE   1,2,1,2             ;FRONT PANEL EXPECTED CODE
      016555             002
1977  016556  045134                       .WORD   TXTP3               ;TEST MODULE SWITCH POS.
1978  016560     000     002              .BYTE   0,2                  ;GAIN, SPREAD
1979  016562  004024                       .WORD   4024                ;CHANNEL A - C EXPECTED VALUE
1980  016564     001     006              .BYTE   1,6                  ;GAIN, SPREAD
1981  016566  004310                       .WORD   4310                ;CHANNEL B - D EXPECTED VALUE
1982  016570     002     053              .BYTE   2,53                 ;GAIN SPREAD
```

F 6
CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 40-1
CVMNAC.P11   10-JUL-81 14:30   MNCAG TEST MODULE INTERACTIVE TESTS

SEQ 0070

```
1983  016572  00772C                       .WORD   7720
1984  016574   000     000                 .BYTE   0,0             ;NULL
1985  016576  000000                        .WORD   0               ;NULL CHANNEL B - D
1986
1987                              ;MNCAG PART 4
1988  016600  004537  016642               JSR     R5,TSETUP       ;GO DO THE WORK
1989  016604   002     001     002          .BYTE   2,1,2,1         ;FRONT PANEL EXPECTED CODE
      016607   001
1990  016610  000000                        .WORD   0               ;NO TEST MODULE CHANGES
1991  016612   000     002                  .BYTE   0,2             ;GAIN, SPREAD
1992  016614  004024                        .WORD   4024            ;CHANNEL A - C EXPECTED VALUE
1993  016616   001     006                  .BYTE   1,6             ;GAIN, SPREAD
1994  016620  004310                        .WORD   4310            ;CHANNEL B - D EXPECTED VALUE
1995  016622   002     053                  .BYTE   2,53            ;GAIN, SPREAD
1996  016624  007720                        .WORD   7720            ;CHANNEL A - C EXPECTED VALUE
1997  016626   000     000                  .BYTE   0,0             ;NULL
1998  016630  000000                        .WORD   0               ;CHANNEL B - D NULL
1999
2000  016632  104401  046423               TYPE    .ENDTST         ;TELL OPERATOR IT'S DONE
2001  016636  000137  001674               JMP     BEG2            ;EXIT
2002
2003                              ;SUBROUTINE TO DO MOST OF THE WORD FOR BEGINT
2004  016642  112500             TSETUP: MOVB    (R5)+,R0        ;GET 1ST ARG.
2005  016644  104401  044722               TYPE    .CHAPOS         ;TELL OPER ''A'' CHANNEL
2006  016650  004737  017416               JSR     PC,TYPITA       ;CONVERT AND TYPE IT
2007  016654  010037  017754               MOV     R0,CHANA        ;SAVE CHANNEL ''A'' EXPECTED VALUE
2008  016660  112500                        MOVB    (R5)+,R0        ;GET 2ND ARG.
2009  016662  104401  044745               TYPE    .CHBPOS         ;TELL OPER 'B' CHANNEL
2010  016666  004737  017416               JSR     PC,TYPITA       ;CONVERT AND TYPE IT
2011  016672  010037  017756               MOV     R0,CHANB        ;SAVE CHANNEL 'B' EXPECTED VALUE
2012  016676  112500                        MOVB    (R5)+,R0        ;GET 3RD ARG.
2013  016700  104401  044770               TYPE    .CHCPOS         ;TELL OPER ''C'' CHANNEL
2014  016704  004737  017416               JSR     PC,TYPITA       ;CONVERT AND TYPE IT
2015  016710  010037  017760               MOV     R0,CHANC        ;SAVE CHANNEL ''C'' EXPECTED VALUE
2016  016714  112500                        MOVB    (R5)+,R0        ;GET 4TH ARG.
2017  016716  104401  045013               TYPE    .CHDPOS         ;TELL OPER 'D'' CHANNEL
2018  016722  004737  017416               JSR     PC,TYPITA       ;CONVERT AND TYPE IT
2019  016726  010037  017762               MOV     R0,CHAND        ;SAVE CHANNEL 'D'' EXPECTED VALUE
2020                              ;NOW TELL OPERATOR ABOUT MNCAG (PREAMP) TEST MODULE POSITIONS
2021  016732  012537  016742               MOV     (R5)+,.60$      ;GET 5TH ARG.
2022  016736  001402                        BEQ     20$             ;BR IF NONE
2023  016740  104401                        TYPE                    ;TELL OPER
2024  016742  000000             60$:    0               .
2025                              ;NOW TELL OPER. TO TYPE ''RETURN'' KEY WHEN READY
2026  016744  104401  047151     20$:    TYPE    .CRWR           ;WAIT FOR 'RETURN''
2027  016750  104412                        RDLIN                   ;WAIT FOR OPERATOR
2028  016752  005726                        TST     (SP)+           ;POP STACK
```

```
2030                                           ;NOW CONVERT CHANNEL AND CHECK OPER SET CORRECT FRONT PANEL POS.
2031                                           ;IF FRONT PANEL SWITCH IS WRONG TELL THE OPERATOR
2032                                           ;IF OK, TEST THE VALUES
2033   016754  013737  001544  001546    MOV     CH1,CH2          ;REPRIME THE CHANNEL VALUE
2034   016762  004537  017460              JSR     R5,CONTA1        ;CONVERT AND CHECK CHANNEL ''A'' FRONT PANEL SWITCH
2035   016766  017754                       CHANA
2036   016770  005237  001546              INC     CH2              ;DO NEXT CHANNEL
2037   016774  004537  017460              JSR     R5,CONTA1        ;CONVERT AND CHECK CHANNEL 'B''
2038   017000  017756                       CHANB
2039   017002  005237  001546              INC     CH2              ;DO NEXT CHANNEL
2040   017006  004537  017460              JSR     R5,CONTA1        ;CONVERT AND CHECK CHANNEL ''C''
2041   017012  017760                       CHANC
2042   017014  005237  001546              INC     CH2              ;DO NEXT CHANNEL
2043   017020  004537  017460              JSR     R5,CONTA1        ;CONVERT AND CHECK CHANNEL 'D''
2044   017024  017762                       CHAND
2045   017026  004737  017124              JSR     PC,TSRT1         ;CONVERT CHANNELS AND VERIFY DATA
2046   017032  004737  017124              JSR     PC,TSRT1         ;SECOND SECTION
2047   017036  000205                       RTS     R5               ;EXIT
2048                                           ;SUBROUTINE TO DO A CONVERSION ON EACH MNCAG CHANNEL
2049   017040  012737  000010  001544  CLRCHT: MOV   #10,CH1        ;LOAD 1ST CHANNEL #
2050   017046  113777  001544  162404  CLRCHS: MOVB  CH1,@ADST1     ;SELECT CHANNEL
2051   017054  004737  017102              JSR     PC,21$           ;CONVERT CHANNEL
2054   017060  004737  017076              JSR     PC,20$           ;INCR. CHANN NUMBER AND CONVERT
(1)    017064  004737  017076              JSR     PC,20$           ;INCR. CHANN NUMBER AND CONVERT
(1)    017070  004737  017076              JSR     PC,20$           ;INCR. CHANN NUMBER AND CONVERT
2055   017074  000207                       RTS     PC               ;EXIT
2056   017076  105277  162356        20$:    INCB    @ADST1           ;UPDATE TO NEXT CHANNEL
2057   017102  112777  000001  162346  21$:  MOVB    #1,@STREG        ;CONVERT CHANNEL
2058   017110  105777  162342        22$:    TSTB    @STREG           ;WAIT FOR DONE
2059   017114  100375                       BPL     22$
2060   017116  005777  162340              TST     @ADBUFF          ;FALSE READ
2061   017122  000207                       RTS     PC               ;EXIT
```

```
2063                                            ;SUBROUTINE TO SETUP FOR CONVERTING DIFFERENT CHANNELS
2064  017124  112537  020002           TSRT1:  MOVB    (R5)+,PRIAC      ;GET INITIAL GAIN FOR A/C
2065  017130  112537  017776                   MOVB    (R5)+,SPRAC      ;GET INITIAL SPREAD FOR A/C
2066  017134  012537  017754                   MOV     (R5)+,CHANA      ;GET CHANNEL A/C EXPECTED VALUE
2067  017140  112537  020004                   MOVB    (R5)+,PRIBD      ;GET INITIAL GAIN FOR B/D
2068  017144  112537  020000                   MOVB    (R5)+,SPRBD      ;GET INITIAL SPREAD FOR B/D
2069  017150  012537  017756                   MOV     (R5)+,CHANB      ;GET CHANNEL B/D EXPECTED VALUE
2070
2071  017154  013737  001544  017764           MOV     CH1,CHXX         ;PRIME THE CHANNEL VALUE
2072  017162  013737  020002  017766           MOV     PRIAC,CHPRIM     ;PRIME THE A/C GAIN VALUE
2073  017170  013737  017776  001566           MOV     SPRAC,SPREAD     ;PRIME THE SPREAD TOLERANCE
2074  017176  013737  017754  001124           MOV     CHANA,$GDDAT     ;PRIME THE EXPECTED VALUE
2075  017204  004737  017634                   JSR     PC,CON4T         ;CONVERT CHANNEL AND TEST RESULT
2076
2077  017210  062737  000002  017764           ADD     #2,CHXX          ;UPDATE TO CHANNEL ''C''
2078  017216  004737  017634                   JSR     PC,CON4T         ;CONVERT CHANNEL AND TEST RESULT
2079                                            ;NOW DO CHANNEL B/D
2080  017222  013737  017756  001124           MOV     CHANB,$GDDAT     ;TEST IF ANY CHANNEL 'B/D' EXPECTED VALUE
2081  017230  001422                           BEQ     1$               ;BR IF NONE
2082  017232  013737  001544  017764           MOV     CH1,CHXX         ;PRIME INIT ''A'' CHANNEL
2083  017240  005237  017764                   INC     CHXX             ;MAKE IT ''CHANNEL B''
2084  017244  013737  020004  017766           MOV     PRIBD,CHPRIM     ;PRIME THE B/D GAIN VALUE
2085  017252  013737  020000  001566           MOV     SPRBD,SPREAD     ;PRIME THE SPREAD TOLERANCE
2086  017260  004737  017634                   JSR     PC,CON4T         ;CONVERT CHANNEL 'B'
2087
2088  017264  062737  000002  017764           ADD     #2,CHXX          ;UPDATE TO CHANNEL 'D'
2089  017272  004737  017634                   JSR     PC,CON4T         ;CONVERT CHANNEL AND TEST RESULT
2090  017276  000207                    1$:    RTS     PC               ;EXIT SUBROUTINE
2091                                            ;SUBROUTINE TO HANDLE THE MNCAG-TA HOLD TEST
2092  017300  112537  017414           TSTHLD: MOVB    (R5)+,10$        ;GET CHANNEL OFFSET FROM CH1
2093  017304  063737  001544  017414           ADD     CH1,10$          ;ADD CH1 VALUE
2094  017312  113777  017414  162140           MOVB    10$,@ADST1       ;LOAD MUX TO ENSURE THE LED IS ON
2095  017320  104401  042660                   TYPE    ,LEDON           ;TELL OPERATOR THE LED SHOULD BE ON
2096  017324  112537  042772                   MOVB    (R5)+,AGTASW     ;LOAD WHICH SWITCH TO PUSH NOW
2097  017330  152737  000060  042772           BISB    #60,AGTASW       ;MAKE CHARACTER AN ASCII NUMBER
2098  017336  104401  042731                   TYPE    ,PUSHAG          ;TELL OPERATOR TO PUSH SWITCH 5,6,7 OR 8
2099  017342  104401  047151                   TYPE    ,CRWR            ;AND DEPRESS 'RETURN'
2100  017346  104412                           RDLIN                    ;WAIT FOR OPERATOR
2101  017350  005726                           TST     (SP)+            ;CLEAN STACK
2102  017352  113777  017414  162100           MOVB    10$,@ADST1       ;LOAD MUX AGAIN, LED WHOULD GO OUT
2103  017360  104401  042704                   TYPE    ,LEDOFF          ;TELL OPERATOR LED SHOULD BE OUT
2104  017364  104401  047151                   TYPE    ,CRWR            ;AND DEPRESS 'RETURN'
2105  017370  104412                           RDLIN
2106  017372  005726                           TST     (SP)+            ;WAIT FOR OPER.
2107  017374  105277  162056                   INCB    @STREG           ;CONVERT THE SELECTED CHANNEL
2108  017400  105777  162052            1$:    TSTB    @STREG           ;WAIT FOR A/D DONE
2109  017404  100375                           BPL     1$               ;
2110  017406  017700  162050                   MOV     @ADBUFF,R0       ;READ VALUE TO CLEAR DONE FLAG
2111  017412  000205                           RTS     R5               ;EXIT
2112  017414  000000                    10$:   0
```

```
2114
2115                                          ;SUBROUTINE TO CONVERT FRONT PANEL VALUE AND TYPE OUT OPER. COMMANDS
2116   017416  010001              TYPITA: MOV     R0,R1              ;COPY R0
2117   017420  006301                      ASL     R1                 ;MAKE WORD VALUE
2118   017422  016137  017450  017432       MOV     FPANL(R1),10$      ;GET TEST POINTER
2119   017430  104401                      TYPE                       ;TELL OPERATOR THE CHANNEL POSITION
2120   017432  000000              10$:    0
2121   017434  006000                      ROR     R0                 ;CONVERT BITS
2122   017436  006000                      ROR     R0
2123   017440  006000                      ROR     R0
2124   017442  042700  037777             BIC     #37777,R0          ;MASK OFF OTHER BITS
2125   017446  000207                      RTS     PC                 ;EXIT
2126
2127   017450  000000              FPANL:  0
2128
2129   017452  044531                      SCM                        ;POINTER TO SET CURRENT MODE TEXT
2130   017454  044602                      SRM                        ;           ''         RESISTANCE        ''
2131   017456  044652                      SVM                        ;           ''         VOLTAGE           ''
2132
2133                                          ;SUBROUTINE TO CONVERT CHANNEL IN ''CH2''
2134
2135   017460  013537  001124      CONTA1: MOV     @(R5)+,$GDDAT      ;LOAD EXPECTED VALUE
2136   017464  012737  017472  001110       MOV     #10$,$LPERR        ;LOAD ERROR RETURN
2137   017472  113777  001546  161760  10$: MOVB    CH2,@ADST1         ;LOAD MUX CHANNEL
2138   017500  052777  000010  161750       BIS     #BIT3,@STREG       ;ENABLE STATUS
2139   017506  052777  000001  161742       BIS     #BIT0,@STREG       ;CONVERT CHANNEL
2140   017514  105777  161736      1$:     TSTB    @STREG             ;WAIT FOR READY
2141   017520  100375                      BPL     1$
2142   017522  017737  161734  001126       MOV     @ADBUFF,$BDDAT     ;READ CONVERSTION
2143   017530  042737  037777  001126       BIC     #37777,$BDDAT      ;MASK OFF DATA BITS
2144   017536  023737  001124  001126       CMP     $GDDAT,$BDDAT      ;COMPARE VALUES
2145   017544  001407                      BEQ     2$                 ;;BR IF SAME
2146   017546  013737  001546  001556       MOV     CH2,CHANL          ;GET CHANNEL VALUE
2147   017554  113737  017766  001557       MOVB    CHPRIM,CHANL+1     ;GET GAIN INFO
2148   017562  104011                      ERROR   11                 ;INCORRECT FRONT PANEL SWITCH POSITION
2149   017564  000205              2$:     RTS     R5                 ;EXIT
2150
2151                                          ;SUBROUTINE TO TEST CERTAIN SOFTWARE SWITCH REGISTER VALUES
2152   017566  032777  040000  161344  SRLOOP: BIT #SW14,@SWR         ;TEST 'LOOP ON THIS TEST'' SWITCH
2153   017574  001016                      BNE     1$                 ;BR IF SET
2154   017576  032777  001000  161334       BIT     #SW09,@SWR         ;TEST 'LOOP ON ERROR'' SWITCH
2155   017604  001012                      BNE     1$                 ;BR IF SET
2156   017606  032777  000400  161324       BIT     #SW08,@SWR         ;TEST IF 'LOOP ON TEST IN SR0-7'' IS SET
2157   017614  001404                      BEQ     2$                 ;BR IF NOT SET
2158   017616  123777  001102  161314       CMPB    $TSTNM,@SWR        ;TEST IF LOOPING ON THIS TEST
2159   017624  001402                      BEQ     1$                 ;BR IF YES
2160   017626  062716  000002      2$:     ADD     #2,(SP)            ;BUMP EXIT P.C.
2161   017632  000207              1$:     RTS     PC                 ;EXIT
2162
```

```
2164                                      ;SUBROUTINE TO CONVERT CHANNEL USING GAIN
2165
2166  017634  012737  017642  001110  CON4T:  MOV   #10$,$LPERR     ;LOAD ERROR RETURN
2167  017642  012700  000004          10$:    MOV   #4,R0          ;LOAD LOOP COUNTER
2168  017646  005001                          CLR   R1             ;CLEAR SUM VALUE
2169  017650  005077  161602                  CLR   @STREG         ;ENSURE CLEAR STATUS
2170  017654  112777  000077  161576          MOVB  #77,@ADST1     ;START ESCAPE
2171  017662  113777  017766  161570          MOVB  CHPRIM,@ADST1  ;LOAD GAIN DATA
2172  017670  113777  017764  161562          MOVB  CHXX,@ADST1    ;LOAD GAIN CHANNEL
2173  017676  105277  161554          1$:     INCB  @STREG         ;CONVERT CHANNEL
2174  017702  105777  161550          2$:     TSTB  @STREG         ;WAIT FOR READY
2175  017706  100375                          BPL   2$
2176  017710  067701  161546                  ADD   @ADBUFF,R1     ;UPDATE SUM
2177  017714  005300                          DEC   R0             ;FINISHED ?
2178  017716  001367                          BNE   1$             ;BR IF NOT
2179  017720  006201                          ASR   R1             ;RESTORE
2180  017722  006201                          ASR   R1
2181  017724  010137  001126                  MOV   R1,$BDDAT      ;LOAD ACTUAL CONVERTED VALUE
2182  017730  013737  017764  001556          MOV   CHXX,CHANL     ;LOAD CHANNEL VALUE IF ERROR
2183  017736  113737  017766  001557          MOVB  CHPRIM,CHANL+1 ;LOAD GAIN INFO IF ERROR
2184  017744  004537  036500                  JSR   R5,COMPRA      ;TEST AGAINST EXPECTED +- SPREAD
2185  017750  104004                          ERROR 4              ;INCORRECT VALUE FROM TEST MODULE
2186  017752  000207                          RTS   PC             ;EXIT
2187
2188
2189  017754  000000          CHANA:  0
2190  017756  000000          CHANB:  0
2191  017760  000000          CHANC:  0
2192  017762  000000          CHAND:  0
2193  017764  000000          CHXX:   0
2194  017766  000000          CHPRIM: 0
2195  017770  000000          GLD0:   0
2196  017772  000000          GLD1:   0
2197  017774  000000          GLD2:   0
2198  017776  000000          SPRAC:  0
2199  020000  000000          SPRBD:  0
2200  020002  000000          PRIAC:  0
2201  020004  000000          PRIBD:  0
```

```
2203
2204                                    .SBTTL   READ-WRITE GAIN BITS LOOP
2205   020006  004737  033454    BEGINX: JSR     PC,FIXONE              ;ENSURE PROPER ADDRESSES
2206   020012  004737  022200            JSR     PC,TCHANK             ;REPORT A/D CONFIG
2207   020016  017700  161440            MOV     @ADBUFF,R0            ;READ TO CLEAR 'READY'
2208   020022  104401  043224            TYPE    .RW1                  ;ASK WHICH CHANNEL
2209   020026  104413                    RDOCT
2210   020030  012637  001556            MOV     (SP)+,CHANL           ;SAVE CHANNEL #
2211   020034  104401  043124            TYPE    .GCHAN                ;ASK WHAT VALUE TO LOAD
2212   020040  104413                    RDOCT
2213   020042  012637  020210            MOV     (SP)+,10$             ;SAVE LOAD VALUE
2214   020046  104401  043250            TYPE    .RW3                  ;ASK WHAT VALUE TO READ
2215   020052  104413                    RDOCT
2216   020054  012600                    MOV     (SP)+,R0              ;GET VALUE
2217   020056  006000                    ROR     R0
2218   020060  006000                    ROR     R0
2219   020062  006000                    ROR     R0
2220   020064  006000                    ROR     R0
2221   020066  006000                    ROR     R0
2222   020070  042700  007777            BIC     #007777,R0            ;MASK OFF OTHER BITS
2223   020074  010037  001124            MOV     R0,$GDDAT             ;LOAD EXPECTED VALUE READ
2224   020100  042737  177700  001556    BIC     #177700,CHANL         ;MASK OFF EXTRA BITS
2225   020106  042737  177760  020210    BIC     #177760,10$           ;MASK OFF EXTRA BITS
2226   020114  112777  000077  161336 1$: MOVB   #77,@ADST1            ;LOAD 'ESCAPE' CODE
2227   020122  113777  020210  161330    MOVB    10$,@ADST1            ;LOAD CHANNEL NUMBER
2228   020130  113777  001556  161322    MOVB    CHANL,@ADST1          ;LOAD VALUE INTO GAIN BITS
2229   020136  112777  000010  161312    MOVB    #BIT3,@STREG          ;ENABLE STATUS
2230   020144  052777  000001  161304    BIS     #BIT0,@STREG          ;CONVERT CHANNEL
2231   020152  105777  161300         2$: TSTB   @STREG                ;WAIT FOR DONE
2232   020156  100375                    BPL     2$
2233   020160  017737  161276  001126    MOV     @ADBUFF,$BDDAT        ;READ GAIN INFO
2234   020166  042737  007777  001126    BIC     #7777,$BDDAT          ;MASK OFF CONVERTED VALUE
2235   020174  023737  001124  001126    CMP     $GDDAT,$BDDAT         ;COMPARE THE VALUES
2236   020202  001401                    BEQ     3$              ;;BR IF THE SAME
2237   020204  104014                    ERROR   14              ;GAIN REGISTER BITS ARE DIFFERENT
2238   020206  000742              3$:    BR      1$              ;TRY MORE
2239
2240   020210  000000             10$:    0
2241
```

```
2243                                    .SBTTL        PRINT VALUES ROUTINE
2244   020212  005077  161240           BEGINP: CLR   @STREG              ;CLEAR STATUS REGISTER
2245   020216  104401  046631                   TYPE  .CCHAN              ;ASK FOR CHANNEL NUMBER
2246   020222  104413                           RDOCT
2247   020224  012600                           MOV   (SP)+,R0            ;GET CHANNEL #
2248   020226  042700  177700                   BIC   #177700,R0          ;MASK OFF OTHER BITS
2249   020232  104401  043124                   TYPE  .GCHAN              ;ASK FOR CHANNEL GAIN/TP TYPE
2250   020236  104413                           RDOCT
2251   020240  012601                           MOV   (SP)+,R1
2254   020242  006101                           ROL   R1                  ;MOVE LEFT
 (1)   020244  006101                           ROL   R1                  ;MOVE LEFT
 (1)   020246  006101                           ROL   R1                  ;MOVE LEFT
 (1)   020250  006101                           ROL   R1                  ;MOVE LEFT
 (1)   020252  006101                           ROL   R1                  ;MOVE LEFT
 (1)   020254  006101                           ROL   R1                  ;MOVE LEFT
2255   020256  042701  176077                   BIC   #176077,R1          ;MASK OFF OTHER BITS
2256   020262  050100                           BIS   R1,R0               ;ADD TOGETHER
2257   020264  010077  160650                   MOV   R0,@SWR             ;LOAD SWITCH REGISTER
2258   020270  012777  001650  161166           MOV   #RETURN,@VECTOR     ;LOAD A/D INVERRUPT VECTOR
2259   020276  017700  160636           10$:    MOV   @SWR,R0             ;GET SWITCH VALUE
2260   020302  010001                           MOV   R0,R1               ;COPY R0
2261   020304  042700  177700                   BIC   #177700,R0          ;MASK TO ALL BUT CHANNEL VALUE
2264   020310  006001                           ROR   R1                  ;MOVE RIGHT
 (1)   020312  006001                           ROR   R1                  ;MOVE RIGHT
 (1)   020314  006001                           ROR   R1                  ;MOVE RIGHT
 (1)   020316  006001                           ROR   R1                  ;MOVE RIGHT
 (1)   020320  006001                           ROR   R1                  ;MOVE RIGHT
 (1)   020322  006001                           ROR   R1                  ;MOVE RIGHT
2265   020324  042701  177760                   BIC   #177760,R1          ;MASK TO ALL BUT GAIN BITS
2266   020330  112777  000077  161122           MOVB  #77,@ADST1          ;START SEQUENCE
2267   020336  110177  161116                   MOVB  R1,@ADST1           ;LOAD GAIN/TP TYPE
2268   020342  110077  161112                   MOVB  R0,@ADST1           ;LOAD SELECTED CHANNEL
2269   020346  005046                           CLR   -(SP)               ;CLEAR PSW
2270   020350  012746  020356                   MOV   #1$,-(SP)
2271   020354  000002                           RTI
2272   020356  032777  020000  160554   1$:     BIT   #BIT13,@SWR         ;IS BIT 13 SET?
2273   020364  001005                           BNE   2$                  ;;YES,SKIP TYPEOUT
2274   020366  104401  046477                   TYPE  .CH
2275   020372  010046                           MOV   R0,-(SP)            ;;SAVE R0 FOR TYPEOUT
 (1)                                                                      ;;TYPE CHANNEL
 (1)   020374  104403                           TYPOS                     ;;GO TYPE--OCTAL ASCII
 (1)   020376  002                              .BYTE 2                   ;;TYPE 2 DIGIT(S)
 (1)   020377  000                              .BYTE 0                   ;;SUPPRESS LEADING ZEROS
2276   020400  112777  000100  161050   2$:     MOVB  #100,@STREG         ;LOAD THE INTERRUPT ENABLE BIT
2277   020406  012702  000010                   MOV   #10,R2              ;TYPEOUT COUNTER
2278   020412  012701  000010           6$:     MOV   #8.,R1              ;LOAD LOOP COUNTER
2279   020416  005003                           CLR   R3                  ;CLEAR AVERAGE
2280   020420  105277  161032           3$:     INCB  @STREG              ;START CONVERSION
2281   020424  000001                           WAIT                      ;WAIT FOR INTRPT.
2282   020426  067703  161030                   ADD   @ADBUFF,R3          ;READ CONVERTED VALUE
2283   020432  005301                           DEC   R1                  ;FINISHED COUNTING
2284   020434  001371                           BNE   3$                  ;BR IF NOT
2285   020436  006203                           ASR   R3
2286   020440  006203                           ASR   R3
2287   020442  006203                           ASR   R3
2288   020444  005503                           ADC   R3
```

```
2289  020446  042703  170000                   BIC     #170000,R3            ;MASK OUT OTHER BITS
2290  020452  032777  020000  160460            BIT     #BIT13,@SWR           ;IS BIT 13 SET?
2291  020460  001403                            BEQ     4$                    ;NOT SET, TYPE OUT LIST
2292  020462  010377  160454                    MOV     R3,@DISPLAY           ;PUT VALUE IN DISPLAY FOR DISPLAY CONTROL
2293  020466  000703                            BR      10$                   ;REPEAT CONVERSION
2294  020470  104401  046502            4$:     TYPE    ,SPACE
2295  020474  010346                            MOV     R3,-(SP)              ;;SAVE R3 FOR TYPEOUT
 (1)                                                                          ;;PRINT OCTAL CONVERTED VALUE
 (1)  020476  104403                            TYPOS                         ;;GO TYPE--OCTAL ASCII
 (1)  020500     004                            .BYTE   4                     ;;TYPE 4 DIGIT(S)
 (1)  020501     001                            .BYTE   1                     ;;TYPE LEADING ZEROS
2296  020502  012701  010000                    MOV     #10000,R1
2297  020506  005301                    5$:     DEC     R1
2298  020510  001376                            BNE     5$
2299  020512  005302                            DEC     R2                    ;DECREMENT THE COUNTER
2300  020514  001336                            BNE     6$                    ;NO CARRIAGE RETURN
2301  020516  104401  001165                    TYPE    ,$CRLF                ;CARRIAGE RETURN
2302  020522  000665                            BR      10$                   ;REPEAT CONVERSION
```

```
2304                                    .SBTTL          LOGIC TEST SECTION START-UP
2305  020524  004737  022006            BEGL:   JSR     PC,WFCHK                 ;CHECK I D CODE IF WESTFIELD MODE
2306  020530  012737  020536  040352            MOV     #2$,AGTST               ;LOAD EOP RETURN IF NO A/D
2307  020536  004737  003452    2$:             JSR     PC,TESTAD               ;SIZE THE NUMBER OF MNCAD'S
2308  020542  004737  022200    1$:             JSR     PC,TCHANK               ;SIZE AND REPORT THE MNCAD CONFIGURATION
2309                                                                            ;ASK IF MNCXX-TA ARE AVAILABLE
2310  020546  004737  003746            JSR     PC,BEGINL                       ;LOGIC TESTS ON MNCAD, MNCAG AND MNCTP
2311  020552  004737  033354            JSR     PC,BUMPAD                       ;MORE TO TEST?
2312  020556  000771                    BR      1$                              ;TEST NEXT A/D
2313  020560  012737  020542  040352    MOV     #1$,AGTST                       ;ADDRESS FOR EOP
2314  020566  000137  040154            JMP     $EOP                            ;TYPE END OF PASS
2315                                    .SBTTL          AUTO TEST START-UP
2316  020572  004737  003452            BEGINA: JSR     PC,TESTAD               ;SIZE THE # OF MNCAD'S
2317  020576  004737  022006            JSR     PC,WFCHK                        ;CHECK I D CODE IF WESTFIELD MODE
2318  020602  004737  022212    1$:     JSR     PC,TCHANL                       ;SIZE AND REPORT THE MNCAD CONFIGURATION
2319                                                                            ;ASK IF MNCXX-TA ARE AVAILABLE
2320  020606  004737  003746            JSR     PC,BEGINL                       ;LOGIC TESTS ON MNCAD, MNCAG AND MNCTP
2321  020612  004737  010206            JSR     PC,WRAP                         ;RUN THE ANALOG TESTS
2322  020616  004737  033354            JSR     PC,BUMPAD                       ;BUMP THE ADDRESSES
2323  020622  000767                    BR      1$                              ;BR AND DO NEXT UNIT
2324  020624  012737  020602  040352    MOV     #1$,AGTST                       ;ADDRESS FOR EOP
2325  020632· 000137  040154            JMP     $EOP                            ;TYPE END OF PASS
2326                                    .SBTTL          WRAPAROUND TEST START-UP
2327  020636  004737  003452            BEGINW: JSR     PC,TESTAD               ;SIZE THE # OF MNCAD'S
2328  020642  004737  022006            JSR     PC,WFCHK                        ;CHECK I D CODE IF WESTFIELD MODE
2329  020646  004737  022212    1$:     JSR     PC,TCHANL                       ;SIZE AND REPORT THE A/D CONFIG.
2330                                                                            ;ASK IF MNCXX-TA ARE AVAILABLE
2331  020652  004737  010206            JSR     PC,WRAP                         ;WRAPAROUND TESTS
2332  020656  004737  033354            JSR     PC,BUMPAD                       ;UPDATE BUS ADDRESSES
2333  020662  000771                    BR      1$                              ;BR AND TEST NEXT UNIT
2334  020664  012737  020646  040352    MOV     #1$,AGTST
2335  020672  000137  040154            JMP     $EOP                            ;INCREMENTS $PASS
2336                                    .SBTTL          NOISE TEST START-UP
2337  020676  004737  033454            BEGINN: JSR     PC,FIXONE               ;ENSURE BASE AND VECTOR SETUP
2338  020702  004737  022200            JSR     PC,TCHANK                       ;SIZE AND REPORT THE MNCAD CONFIG.
2339  020706  005037  001552            CLR     NMBEXT                          ;CLEAR MULTIPLE UNIT FLAG
2340  020712  104401  041164            TYPE    ,SCHAN                          ;ASK FOR STARTING NOISE CHANNEL
2341  020716  104413                    RDOCT                                   ;GET OPER. CHANNEL INPUT
2342  020720  012637  001474            MOV     (SP)+,BASECH                    ;SAVE 1ST CHANNEL
2343  020724  104401  041220            TYPE    ,ECHAN                          ;ASK FOR END NOISE CHANNEL
2344  020730  104413                    RDOCT                                   ;GET OPER. CHANNEL INPUT
2345  020732  012637  001476            MOV     (SP)+,BASEND                    ;SAVE LAST CHANNEL
2346  020736  001003                    BNE     1$                              ;BR IF NON-ZERO
2347  020740  013737  001474  001476    MOV     BASECH,BASEND                   ;TAKE CARE IF ONLY 1 CHANNEL
2348  020746  013737  001474  001556 1$:MOV     BASECH,CHANL                    ;INIT THE STARTING CHANNEL
2349  020754  012737  000001  001530    MOV     #1,WIDE                         ;SET MANUAL ENTRY FLAG
2350  020762  004737  012410            JSR     PC,NOITST                       ;RUN NOISE TEST
2351  020766  023737  001556  001476 2$:CMP     CHANL,BASEND                    ;LAST CHANNEL
2352  020774  001405                    BEQ     3$                              ;BR IF FINISHED
2353  020776  005237  001556            INC     CHANL                           ;BUMP TO NEXT CHANNEL
2354  021002  004737  012470            JSR     PC,NOITS1                       ;RUN NOISE TEST AGAIN
2355  021006  000767                    BR      2$
2356  021010  012737  020746  040352 3$:MOV     #1$,AGTST                       ;LOAD RETRURN POINTER
2357  021016  000137  040154            JMP     $EOP                            ;AND REPORT END OF PASS
```

B 7

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 48
CVMNAC.P11   10-JUL-81 14:30   MNCAG COMMON MODE REJECTION TEST   SEQ 0079

```
 2359                                        .SBTTL  MNCAG  COMMON MODE REJECTION TEST
 2360   021022  104401  047544      BEGINM:  TYPE    ,COMOD1         ;TELL OPERATOR THE TEST NAME
 2361   021026  104401  046631               TYPE    ,CCHAN          ;ASK FOR CHANNEL TO USE
 2362   021032  104413                        RDOCT                   ;GET INPUT
 2363   021034  012600                        MOV     (SP)+,R0        ;GET HIS ANSWER
 2364   021036  010037  001556                MOV     R0,CHANL        ;SAVE CHANNEL TO TEST
 2365   021042  112777  000077  160410        MOVB    #77,@ADST1      ;ENSURE MNCAG GAIN OF .5
 2366   021050  112777  000000  160402        MOVB    #0,@ADST1       ;   FOR
 2367   021056  110077  160376                MOVB    R0,@ADST1       ;        THIS TEST
 2368   021062  010037  001554                MOV     R0,DUMMY        ;LOAD DUMMY CHANNEL
 2369   021066  104401  021074                TYPE    ,65$            ;;TYPE ASCIZ STRING
   (1)  021072  000424                        BR      64$             ;;GET OVER THE ASCIZ
   (1)                                 ;;65$:  .ASCIZ  <15><12>/SET COMMON MODE VOLTAGE TO + 10 VOLTS/
   (1)  021144                          64$:
 2370   021144  104401  047151                TYPE    ,CRWR           ;CRLF MESSAGE
 2371   021150  104412                        RDLIN                   ;WAIT FOR CARRIAGE RETURN
 2372   021152  005726                        TST     (SP)+           ;POP ADDRESS OFF STACK
 2373   021154  004537  036340                JSR     R5,CONVTC       ;GET CONVERSION VALUE
 2374   021160  013737  001540  001574        MOV     TEMP,EDGE       ;GET VALUE TO FIND EDGE OF
 2375   021166  004537  034170                JSR     R5,SARSUB       ;GET EDGE
 2376   021172  000062                        50.                     ;50% EDGE
 2377   021174  013737  001570  001540        MOV     DAC,TEMP        ;SAVE DAC SETTING IN TEMP
 2378   021202  104401  021210                TYPE    ,67$            ;;TYPE ASCIZ STRING
   (1)  021206  000424                        BR      66$             ;;GET OVER THE ASCIZ
   (1)                                 ;;67$:  .ASCIZ  <15><12>/SET COMMON MODE VOLTAGE TO - 10 VOLTS/
   (1)  021260                          66$:
 2379   021260  104401  047151                TYPE    ,CRWR           ;CRLF MESSAGE
 2380   021264  104412                        RDLIN                   ;WAIT FOR CARRIAGE RETURN
 2381   021266  005726                        TST     (SP)+           ;POP ADDRESS OFF STACK
 2382   021270  004537  034170                JSR     R5,SARSUB       ;GET EDGE
 2383   021274  000062                        50.                     ;50% EDGE
 2384   021276  163737  001540  001570        SUB     TEMP,DAC        ;GET DIFFERENCE
 2385   021304  104401  047544                TYPE    ,COMOD1         ;OUTPUT TEXT
 2386   021310  013702  001570                MOV     DAC,R2          ;GET NUMBER INTO R2
 2387   021314  104416                        TYPDC                   ;TYPE DECIMAL NUMBER
 2388   021316  104401  050644                TYPE    ,MLSB           ;ADD LSB TEXT
 2389   021322  013702  001570                MOV     DAC,R2          ;GET RESULT
 2390   021326  100001                        BPL     1$              ;BR IF POSITIVE
 2391   021330  005402                        NEG     R2              ;INVERT IF NEGATIVE
 2392   021332  020237  037700      1$:       CMP     R2,VCM          ;TEST AGAINST LIMIT
 2393   021336  003403                        BLE     2$              ;BR IF WITHIN LIMIT
 2394   021340  104401  047505                TYPE    ,ERMSG          ;TELL OPER. ERROR
 2395   021344  000402                        BR      3$
 2396   021346  104401  046620      2$:       TYPE    ,OKMSG          ;TELL OPER. OK
 2397   021352  104401  046423      3$:       TYPE    ,ENDTST
 2398   021356  000137  001674                JMP     BEG2            ;GO BACK TO SELECT TEST
```

```
2400                                               .SBTTL   DIFFERENTIAL LINEARITY AND REL. ACC. START-UP
2401   021362  004737  033454          BEGIND: JSR      PC,FIXONE        ;ENSURE BASE AND VECTOR SETUP
2402   021366  004737  022200                  JSR      PC,TCHANK        ;SIZE AND REPORT A/D CONFIG
2403   021372  005037  001552                  CLR      NMBEXT           ;ENSURE ONLY 1 MNCAD
2404   021376  104401  042363                  TYPE     ,RMPTXT          ;TELL OPERATOR ABOUT SETTING MNCAG-TA SWITCHES
2405   021402  104401  041164                  TYPE     ,SCHAN           ;ASK OPER. THE STARTING CHANNEL
2406   021406  104413                          RDOCT                     ;GET OPER INPUT
2407   021410  012637  001474                  MOV      (SP)+,BASECH     ;SAVE 1ST CHANNEL
2408   021414  104401  041220                  TYPE     ,ECHAN           ;ASK OPER. THE LAST CHANNEL
2409   021420  104413                          RDOCT                     ;GET OPER INPUT
2410   021422  012637  001476                  MOV      (SP)+,BASEND     ;SAVE LAST CHANNEL
2411   021426  001003                          BNE      1$               ;BR IF THERE WAS ONE
2412   021430  013737  001474  001476          MOV      BASECH,BASEND    ;ELSE ENSURE ONLY 1ST RUNS
2413   021436  013737  001474  023176  1$:     MOV      BASECH,CHA       ;LOAD CHANNEL TO RUN ON
2414   021444  112777  000077  160006  2$:     MOVB     #77,@ADST1       ;ENSURE MNCAG GAIN
2415   021452  112777  000000  160000          MOVB     #0,@ADST1        ;  OF .5
2416   021460  113777  023176  157772          MOVB     CHA,@ADST1       ;       ON THIS CHANNEL
2417   021466  004737  034370                  JSR      PC,DIFLIN        ;RUN DIF LIN AND REL ACC.
2418   021472  023737  023176  001476          CMP      CHA,BASEND       ;TEST IF LAST CHANNEL
2419   021500  001403                          BEQ      3$               ;BR IF FINISHED
2420   021502  005237  023176                  INC      CHA              ;UPDATE CHANNEL NUMBER
2421   021506  000756                          BR       2$               ;AND RUN ANOTHER TIME
2422   021510  012737  021436  040352  3$:     MOV      #1$,AGTST        ;LOAD RETURN ADDRESS
2423   021516  000137  040154                  JMP      $EOP             ;TYPE END OF PASS
2424                                               .SBTTL   SETTLING TEST START-UP
2425   021522  004737  033454          BEGINS: JSR      PC,FIXONE        ;ENSURE BASE AND VECTOR SETUP
2426   021526  004737  022200                  JSR      PC,TCHANK        ;SIZE AND REPORT A/D CONFIG
2427   021532  005037  001552                  CLR      NMBEXT           ;ENSURE ONLY 1 MNCAD
2428   021536  104401  021676                  TYPE     ,10$             ;ASK FOR 1ST CHANNEL
2429   021542  104413                          RDOCT                     ;GET OPER. INPUT
2430   021544  012637  021672                  MOV      (SP)+,2$         ;AND SAVE IT
2431   021550  104401  021750                  TYPE     ,12$             ;ASK FOR THE ''GAIN OR TC TYPE'' VALUE
2432   021554  104413                          RDOCT                     ;GET OPER. INPUT
2433   021556  112637  021652                  MOVB     (SP)+,13$        ;SAVE VALUE
2434   021562  105726                          TSTB     (SP)+            ;ADJUST STACK
2435   021564  104401  021731                  TYPE     ,11$             ;ASK FOR 2ND CHANNEL
2436   021570  104413                          RDOCT                     ;GET OPER INPUT
2437   021572  012637  021674                  MOV      (SP)+,3$         ;AND SAVE IT
2438   021576  104401  021750                  TYPE     ,12$             ;ASK FOR THE ''GAIN OR TC TYPE'' VALUE
2439   021602  104413                          RDOCT                     ;GET OPER INPUT
2440   021604  112637  021653                  MOVB     (SP)+,13$+1      ;SAVE VALUE
2441   021610  105726                          TSTB     (SP)+            ;ADJUST STACK
2442   021612  042737  177700  021672          BIC      #177700,2$       ;ENSURE GOOD CHANNEL VALUE
2443   021620  042737  177700  021674          BIC      #177700,3$       ;    ''         ''
2444   021626  104401  001165          1$:     TYPE     ,SCRLF           ;FRESH LINE
2445   021632  013737  021672  001544          MOV      2$,CH1           ;LOAD 1ST CHANNEL VALUE
2446   021640  013737  021674  001546          MOV      3$,CH2           ;LOAD 2ND CHANNEL VALUE
2447   021646  004537  014104                  JSR      R5,LDSETG        ;LOAD GAIN OR TC TYPE VALUE
2448   021652  000     000             13$:    .BYTE    0,0              ;VALUE TO BE LOADED
2449   021654  004737  013662                  JSR      PC,SETTLE        ;RUN SETTLING TEST
2450   021660  012737  021626  040352          MOV      #1$,AGTST        ;LOAD RETURN ADDRESS
2451   021666  000137  040154                  JMP      $EOP             ;AND REPORT END OF PASS
2452   021672  000000                  2$:     0
2453   021674  000000                  3$:     0
2462
```

```
2464                                 ;*ROUTINE TO CHECK FOR PROPER I D CODE IF TESTER MODE
2465
2466  022006  005037  022130   WFCHK:  CLR    WFAD            ;CLEAR TESTING MNCAD FLAG
2467  022012  005037  022132           CLR    WFAM            ;CLEAR TESTING MNCAM FLAG
2468  022016  005037  022134           CLR    WFAG            ;CLEAR TESTING MNCAG FLAG
2469  022022  005737  001602           TST    WFTEST          ;RUNNING  ON TESTER?
2470  022026  100037                    BPL    4$              ;;BR IF NOT
2471  022030  017700  157472           MOV    @DRVDIR,R0      ;READ TESTER (I.D. LINES)
2472  022034  042700  177417           BIC    #177417,R0      ;CLEAR OFF OTHER BITS
2473  022040  010037  001126           MOV    R0,$BDDAT       ;LOAD VALUE READ FROM TESTER
2474  022044  023700  022136           CMP    K60,R0          ;TEST IF VALID I.D. CODE
2475  022050  001005                    BNE    1$              ;BR IF NOT MNCAD CODE
2476  022052  005237  022130           INC    WFAD            ;SET TESTING MNCAD FLAG
2477  022056  104401  043725           TYPE   ,TSTAD          ;TYPE TESTING A/D MESSAGE
2478  022062  000421                    BR     4$
2479  022064  023700  022142   1$:     CMP    K340,R0         ;TEST IF VALID I.D. CODE FOR AM
2480  022070  001005                    BNE    2$              ;NR IF NOT MNCAM CODE
2481  022072  005237  022132           INC    WFAM            ;SET TESTING MNCAM FLAG
2482  022076  104401  043747           TYPE   ,TSTADM         ;TYPE TESTING A/D AND AM MESSAGE
2483  022102  000411                    BR     4$
2484  022104  023700  022140   2$:     CMP    K20,R0          ;TEST IF VALID I.D. CODE
2485  022110  001005                    BNE    3$              ;BR IF NOT MNCAG
2486  022112  005237  022134           INC    WFAG            ;SET TESTING MNCAG FLAG
2487  022116  104401  043771           TYPE   ,TSTAG          ;TYPE TESTING AG MESSAGE
2488  022122  000401                    BR     4$
2489  022124  104007           3$:     ERROR  7               ;INCORRECT I.D. CODE FOR MODULE
2490  022126  000207           4$:     RTS    PC              ;RETURN
2491
2492  022130  000000           WFAD:   0
2493  022132  000000           WFAM:   0
2494  022134  000000           WFAG:   0
2495
2496  022136  000060           K60:    60               ;MNCAD ID VALUE
2497  022140  000020           K20:    20               ;MNCAG  ''       ''
2498  022142  000340           K340:   340              ;MNCAM  ''       ''
2499
2500                                 ;SUBROUTINE TO DELAY A FIX AMOUNT OF TIME
2501  022144  013700  001434   STALL:  MOV    BARF0,R0        ;PRINE THE DELAY
2502  022150  005300           1$:     DEC    R0              ;DELAY
2503  022152  001376                    BNE    1$              ;
2504  022154  000207                    RTS    PC              ;EXIT
2505                                 ;SUBROUTINE TO TEST IF FIRST PASS OR AUTO MODE
2506                                 ; IF TRUE EXIT, IF NOT BUMP ENTRY BY 1 WORD AND THEN EXIT
2507  022156  005737  001176   AFIRST: TST    $PASS           ;TEST IF FIRST PASS
2508  022162  001005                    BNE    1$              ;BR IF NOT FIRST
2509  022164  105737  001134           TSTB   $AUTOB          ;TEST IF AUTO MODE
2510  022170  001002                    BNE    1$              ;BR IF AUTO MODE
2511  022172  062716  000002           ADD    #2,(SP)         ;ADJUST RETURN VALUE
2512  022176  000207           1$:     RTS    PC              ;EXIT
```

```
2514                                      ;PART 1 *ROUTINE TO TYPE OUT A/D CONFIGURATION
2515                                      ;PART 2 *IF RUNNING IN TEST MODULE MODE, ASK FOR CHANNELS TO TEST
2516   022200   005237   023176  TCHANK:  INC    CHA              ;SET LOGIC TEST ENTRY FLAG
2517   022204   000404                    BR     TCHANM           ;BR
2518   022206   000137   022652  TCHANN:  JMP    TCHANE           ;BR TO EXIT
2519   022212   005037   023176  TCHANL:  CLR    CHA              ;CLEAR LOGIC TEST ENTRY FLAG
2520   022216   004737   014222  TCHANM:  JSR    PC,LD01CH        ;PRESET MNCTC CHANNELS
2521   022222   005737   001176           TST    $PASS            ;TEST IF FIRST PASS
2522   022226   001367                    BNE    TCHANN           ;BR. AND EXIT IF NOT FIRST PASS
2523   022230   005077   157222           CLR    @STREG           ;CLEAR A/D STATUS
2524   022234   005037   023200           CLR    CHB              ;CLEAR MNCAG COUNTER
2525   022240   012700   060156           MOV    #CHTABL,R0       ;LOAD POINTER
2526   022244   005020           1$:      CLR    (R0)+            ;CLEAR CHANNEL TYPE TABLE
2527   022246   022700   060256           CMP    #CHTABL+100,R0   ;TEST IF FINISHED
2528   022252   001374                    BNE    1$               ;BR IF NOT DONE CLEARING BUFFER
2529   022254   005000                    CLR    R0               ;INIT R0
2530   022256   005001                    CLR    R1               ;INIT R1
2531   022260   004737   022156           JSR    PC,AFIRST        ;TEST IF FIRST PASS
2532   022264   000422                    BR     3$               ;BR IF NOT
2533   022266   104401   043454           TYPE   ,VTMSG           ;REPORT UNIT #
2534   022272   004737   055712           JSR    PC,WHICHU        ;DETERMINE ASCII UNIT #
2535   022276   013746   001622           MOV    UNITBD,-(SP)
2536   022302   104403                    TYPOS
2537   022304      001      000           .BYTE  1,0
2538   022306   104401   001165           TYPE   ,$CRLF           ;LEAVE A BLANK LINE
2539   022312   004737   022156  2$:      JSR    PC,AFIRST        ;TEST IF FIRST PASS
2540   022316   000405                    BR     3$               ;BR IF NOT
2541   022320   010146                    MOV    R1,-(SP)         ;:SAVE R1 FOR TYPEOUT
 (1)   022322   104403                    TYPOS                   ;:GO TYPE--OCTAL ASCII
 (1)   022324      002                    .BYTE  2                ;:TYPE 2 DIGIT(S)
 (1)   022325      000                    .BYTE  0                ;:SUPPRESS LEADING ZEROS
2542   022326   104401   042162           TYPE   ,MDASH           ;TYPE A DASH
2543   022332   005277   157120  3$:      INC    @STREG           ;START CONVERSION
2544   022336   105777   157114  4$:      TSTB   @STREG           ;WAIT FOR DONE
2545   022342   100375                    BPL    4$               ;BR IF NOT
2546   022344   017700   157112           MOV    @ADBUFF,R0       ;GET CONVERTED VALUE
2547   022350   042700   007777           BIC    #7777,R0         ;IS CHANNEL SINGLE ENDED
2548   022354   001007                    BNE    5$               ;:CHANNEL IS NOT SINGLE ENDED
2549   022356   012737   043040   022534  MOV    #MSE,12$         ;LOAD MESSAGE POINTER
2550   022364   004537   026404           JSR    R5,LODTAB
2551   022370      001      010           .BYTE  1,10             ;LOAD SINGLE ENDED CODE, LOAD NUMBER OF CHAN
2552   022372   000444                    BR     10$
2553   022374   032700   140000  5$:      BIT    #140000,R0       ;TEST IF MNCAG CHANNEL
2554   022400   001412                    BEQ    6$               ;BR IF NOT
2555   022402   062737   000004   023200  ADD    #4,CHB           ;UPDATE NUMBER OF MNCAG DETECTED
2556   022410   012737   043100   022534  MOV    #MPRMP,12$       ;LOAD MESSAGE POINTER
2557   022416   004537   026404           JSR    R5,LODTAB
2558   022422      003      004           .BYTE  3,4              ;LOAD PREAMP CODE, LOAD NUMBER OF CHAN'S
2559   022424   000427                    BR     10$
2560   022426   022700   010000  6$:      CMP    #10000,R0        ;TEST IF MNCTP CHANNEL
2561   022432   001016                    BNE    7$               ;BR IF NOT
2562   022434   012737   043112   022534  MOV    #MTCMP,12$       ;LOAD MESSAGE POINTER
2563   022442   022701   000074           CMP    #74,R1           ;TEST IF CHANNEL STARTS AT 74
2564   022446   001004                    BNE    60$              ;BR IF NOT
2565   022450   004537   026404           JSR    R5,LODTAB        ;LOAD MNCTP CODE, LOAD NUMBER OF CHAN'S
2566   022454      004      004           .BYTE  4,4              ;IF MNCTP STARTS ON CHAN 74
```

```
2567  022456  000412                        BR     10$
2568  022460  004537  026404         60$:   JSR    R5,LODTAB
2569  022464     004     010                .BYTE  4,10                ;LOAD MNCTP CODE, LOAD NUMBER OF CHAN'S
2570  022466  000406                        BR     10$
2571  022470  012737  043060  022534  7$:   MOV    #MDIF,12$           ;LOAD MESSAGE POINTER
2572  022476  004537  026404                JSR    R5,LODTAB
2573  022502     002     004                .BYTE  2,4                 ;LOAD DIFFERENTIAL CODE, LOAD NUMBER OF CHAN'S
2574  022504  022701  000100         10$:   CMP    #100,R1             ;IS CHANNEL > LAST POSSIBLE CHANNEL
2575  022510  101002                        BHI    11$                 ;;NO
2576  022512  012701  000077                MOV    #77,R1              ;YES, SET TO LAST CHANNEL
2577  022516  004737  022156         11$:   JSR    PC,AFIRST           ;TEST IF FIRST PASS
2578  022522  000405                        BR     13$                 ;BR IF NOT
2579  022524  010146                        MOV    R1,-(SP)            ;;SAVE R1 FOR TYPEOUT
 (1)  022526  104403                        TYPOS                      ;;GO TYPE--OCTAL ASCII
 (1)  022530     002                        .BYTE  2                   ;;TYPE 2 DIGIT(S)
 (1)  022531     000                        .BYTE  0                   ;;SUPPRESS LEADING ZEROS
2580  022532  104401                        TYPE                       ;REPORT THE CHANNEL TYPE
2581  022534  043040                 12$:   MSE                        ;POINTER TO MESSAGE
2582  022536  005201                 13$:   INC    R1                  ;SET CHANNEL TO NEXT SET OF CHANNELS
2583  022540  022701  000100                CMP    #100,R1             ;DONE?
2584  022544  001412                        BEQ    14$                 ;;YES
2585  022546  010100                        MOV    R1,R0               ;GET CHANNEL
2586  022550  000300                        SWAB   R0                  ;PUT CHANNEL NUMBER IN HIGH BYTE
2587  022552  052700  000010                BIS    #BIT3,R0            ;SET STATUS ENABLE BIT
2588  022556  010077  156674                MOV    R0,@STREG           ;LOAD INTO A/D STATUS REGISTER
2589  022562  032777  000002  156666        BIT    #BIT1,@STREG        ;IS NON-EXSISTENT CHANNEL BIT SET?
2590  022570  001650                        BEQ    2$                  ;;NO
2591                                   ;PART 2 IF USING TEST MODULE OR TESTER MODE, DO MORE TESTING
2592                                   ;        IF NOT THEN EXIT
2593  022572  023727  023200  000025  14$:  CMP    CHB,#25             ;TEST HOW MANY MNCAG FOUND
2594  022600  103402                        BLO    15$                 ;BR IF LESS THAN LIMIT
2595  022602  104401  042274                TYPE   .WOWAGS             ;TELL OPERATOR TOO MANY DETECTED
2596  022606  052737  100200  060156  15$:  BIS    #100200,CHTABL      ;ENSURE CH 0 + 1
2597  022614  052737  100200  060160        BIS    #100200,CHTABL+2         ; AND 2 + 3 ARE TESTED
2598  022622  005737  001422                TST    ADTA                ;TEST IF MNCAD-TA CONNECTED
2599  022626  001012                        BNE    ASKWHO              ;BR IF YES
2600  022630  005737  001424                TST    AMTA                ;  ''              AM        ''
2601  022634  001007                        BNE    ASKWHO              ;BR IF YES
2602  022636  005737  001426                TST    AGTA                ;  ''              AG        ''
2603  022642  001004                        BNE    ASKWHO              ;BR IF YES
2604  022644  005737  001430                TST    TPTA                ;  ''              TP        ''
2605  022650  001001                        BNE    ASKWHO              ;BR IF YES
2606  022652  000207                 TCHANE: RTS   PC                  ;EXIT IF DONE
2607                                   ;ROUTINE TO ASK OPERATOR ABOUT MNCXX-TA BEING CONNECTED
2608  022654  004737  022156         ASKWHO: JSR   PC,AFIRST           ;TEST IF FIRST PASS
2609  022660  000545                        BR     ASKDON              ;BR IF NOT
2610  022662  005737  023176                TST    CHA                 ;TEST IF LOGIC TEST ENTRY FLAG IS SET
2611  022666  001142                        BNE    ASKDON              ;BR IF IT WAS SET
2612  022670  012700  000004                MOV    #4,R0               ;LOAD INITIAL CHANNEL
2613  022674  005001                        CLR    R1                  ;INIT 2ND CHANNEL
2614                                   ;DETERMINE IF CHANNEL ( R0 ) IS SINGLE ENDED
2615  022676  126027  060156  000001  ASKSE: CMPB  CHTABL(R0),#1       ;TEST IF SE
2616  022704  001027                        BNE    ASKDIF              ;BR IF NOT
2617  022706  062701  000007                ADD    #7,R1               ;UPDATE END CHANNEL VALUE
2618  022712  120027  000004                CMPB   R0,#4               ;TEST IF CHANNEL 4
2619  022716  001004                        BNE    2$                  ;BR IF NOT
```

G 7
CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063)   10-JUL-81 14:41 PAGE 51-2
CVMNAC.P11   10-JUL-81 14:30   SETTLING TEST START-UP

SEQ 0084

```
2620  022720  105737  001422     1$:     TSTB    ADTA            ;TEST IF MNCAD-TA IS CONNECTED
2621  022724  001414                      BEQ     4$              ;BR IF NOT
2622  022726  000406                      BR      3$              ;
2623  022730  120027  000010     2$:     CMPB    R0,#10          ;TEST IF CHANNEL #10
2624  022734  001771                      BEQ     1$              ;BR IF YES
2625  022736  105737  001424             TSTB    AMTA            ;TEST IF MNCAM-TA IS CONNECTED
2626  022742  001405                      BEQ     4$              ;BR IF NOT
2627  022744  004737  026314     3$:     JSR     PC,ASKC         ;ASK OPERATOR
2628  022750  000402                      BR      4$              ;BR IF ANSWER WAS NO
2629  022752  004737  026446             JSR     PC,SETASK       ;GO AND SET "TEST THIS CHANNEL BIT"
2630  022756  005201             4$:     INC     R1              ;UPDATE TO NEXT CHANNEL
2631  022760  010100                      MOV     R1,R0           ;PRIME 1ST CHANNEL
2632  022762  000745                      BR      ASKSE           ;TEST NEXT CHANNEL
2633                            ;DETERMINE IF THE CHANNEL IS DIFFERENTIAL (DIF)
2634  022764  126027  060156  000002  ASKDIF: CMPB  CHTABL(R0),#2  ;TEST IF CHANNEL TYPE IS DIFF.
2635  022772  001024                      BNE     ASKAG           ;BR IF NOT
2636  022774  062701  000003             ADD     #3,R1           ;UPDATE TO LAST CHANNEL OF DIFF CHANNEL
2637  023000  120027  000010             CMPB    R0,#10          ;TEST IF CHANNEL #10
2638  023004  001004                      BNE     1$              ;BR IF NOT
2639  023006  105737  001422             TSTB    ADTA            ;TEST IF MNCAD-TA IS CONNECTED
2640  023012  001411                      BEQ     3$              ;BR IF NOT
2641  023014  000403                      BR      2$
2642  023016  105737  001424     1$:     TSTB    AMTA            ;TEST IF MNCAM-TA IS CONNECTED
2643  023022  001405                      BEQ     3$              ;BR IF NOT
2644  023024  004737  026314     2$:     JSR     PC,ASKC         ;ASK THE OPERATOR
2645  023030  000402                      BR      3$              ;BR IF ANSWER WAS NO
2646  023032  004737  026446             JSR     PC,SETASK       ;SET "TEST THIS CHANNEL BIT"
2647  023036  005201             3$:     INC     R1              ;UPDATE CHANNEL
2648  023040  010100                      MOV     R1,R0           ;UPDATE 1ST CHANNEL
2649  023042  000715                      BR      ASKSE           ;TEST NEXT CHANNEL
2650                            ;DETERMINE IF THE CHANNEL IS A MNCAG
2651  023044  126027  060156  000003  ASKAG: CMPB   CHTABL(R0),#3  ;TEST IF CHANNEL TYPE IS MNCAG
2652  023052  001015                      BNE     ASKTP           ;BR IF NOT
2653  023054  062701  000003             ADD     #3,R1           ;UPDATE TO LAST CHANNEL OF MNCAG CHANNEL
2654  023060  105737  001426             TSTB    AGTA            ;TEST IF MNCAG-TA IS CONNECTED
2655  023064  001405                      BEQ     1$              ;BR IF NOT
2656  023066  004737  026314             JSR     PC,ASKC         ;ASK THE OPERATOR
2657  023072  000402                      BR      1$              ;BR IF ANSWER WAS NO
2658  023074  004737  026446             JSR     PC,SETASK       ;SET "TEST THIS CHANNEL BITS"
2659  023100  005201             1$:     INC     R1              ;UPDATE CHANNEL
2660  023102  010100                      MOV     R1,R0           ;UPDATE 1ST CHANNEL
2661  023104  000674                      BR      ASKSE           ;TEST NEXT CHANNEL
2662                            ;DETERMINE IF THE CHANNEL IS A MNCTP
2663  023106  126027  060156  000004  ASKTP: CMPB   CHTABL(R0),#4  ;TEST IF CHANNEL TYPE IS MNCTP
2664  023114  001022                      BNE     ASKOOP          ;BR IF NOT
2665  023116  022700  000074             CMP     #74,R0          ;TEST IF CHANNEL STARTS AT 74
2666  023122  001402                      BEQ     2$              ;BR IF LAST MNCTP
2667  023124  062701  000004             ADD     #4,R1           ;USE THE FULL BANK OF CHANNELS
2668  023130  062701  000003     2$:     ADD     #3,R1           ;UPDATE TO LAST CHANNEL OF MNCTP CHANNEL
2669  023134  105737  001430             TSTB    TPTA            ;TEST IF MNCTP-TA IS CONNECTED
2670  023140  001405                      BEQ     1$              ;BR IF NOT
2671  023142  004737  026314             JSR     PC,ASKC         ;ASK THE OPERATOR
2672  023146  000402                      BR      1$              ;BR IF THE ANSWER WAS NO
2673  023150  004737  026446             JSR     PC,SETASK       ;SET "TEST THIS CHANNEL BITS"
2674  023154  005201             1$:     INC     R1              ;UPDATE CHANNEL
2675  023156  010100                      MOV     R1,R0           ;UPDATE 1ST CHANNEL
```

```
2676  023160  000646                   BR      ASKSE              ;TEST NEXT CHANNEL
2677                            ;OOPS THE CHANNEL TYPE WAS NOT #1, 2, 3, OR 4
2678  023162  005760  060156   ASKOOP: TST     CHTABL(R0)         ;TEST IF NON-EXISTANT CHANNEL
2679  023166  001402                   BEQ     ASKDON             ;BR IF NO MORE
2680  023170  104401  042166           TYPE    ,IDONTK            ;TELL OPERATOR SOME UNEXPECTED TYPE OF CHANNEL
2681  023174  000207           ASKDON: RTS     PC                 ;EXIT
2682  023176  000000           CHA:    0
2683  023200  000000           CHB:    0
```

```
2685
2686                                       ;SUBROUTINE TO DO THE NOISE TEST AT GAINS OF 50 AND 500 ON THE MNCAG
2687                        .              ;            ""      ""      ""      "" OF 00, 01, 10 AND 11 ON THE MNCTP
2688
2689                                       ;BUFFER IS AN AREA OF MEMORY WHERE THE COUNT OF THE CONVERSION VALUE IS TO BE
2690                                       ;        STORED.  THE INDEX INTO BUFFER IS THE VALUE OF THE A/D CONVERSION AND
2691                                       ;        THE COUNT AT THAT LOCATION IS THE NUMBER OF TIMES THAT A/D CONVERSION
2692                                       ;        VALUE WAS SEEN.
2693
2694                                       ;CLEAR THE BUFFER AREA IN PREPARATION FOR COLLECTION OF DATA FOR THE A/D
2695                                       ;NOISE CALCULATION.
2696
2697  023202  012700  061306     PRI4A:    MOV     #BUFFER,R0        ;CLEAR RESULT BUFFER AREA
2698  023206  005037  024700               CLR     BADCAL            ;CLEAR BAD CALCULATION FLAG
2699  023212  012701  010000               MOV     #4096.,R1 .
2700  023216  005020             1$:       CLR     (R0)+
2701  023220  005301                       DEC     R1
2702  023222  001375                       BNE     1$                ;BRANCH IF NOT DONE
2703
2704                                       ;SETUP THE A/D CHANNEL TO TEST AND ENABLE INTERRUPTS ON THE A/D MODULE.
2705                                       ;SETUP THE INTERRUPT VECTOR (@VECTOR) AND THEN DO 16384 DECIMAL CONVERSIONS
2706                                       ;LOGGING THE RESULTS IN BUFFER.
2707                                       ;
2708  023224  013700  001556               MOV     CHANL,R0          ;SETUP TO DO A CONVERSION
2709  023230  000300                       SWAB    R0
2710  023232  052700  000100               BIS     #100,R0
2711  023236  010077  156214               MOV     R0,@STREG
2712  023242  012777  001650  156214       MOV     #RETURN,@VECTOR   ;SETUP INTERRUPT VECTORS
2713  023250  012777  000200  156210       MOV     #200,@VECTR1
2714  023256  012700  040000               MOV     #16384.,R0        ;DO 16384(10) CONVERSIONS
2715  023262  005277  156170     COLECT:   INC     @STREG            ;START CONVERSION
2716  023266  000001                       WAIT                      ;WAIT TILL CONVERSION IS DONE
2717  023270  017701  156166               MOV     @ADBUFF,R1        ;READ RESULT
2718  023274  006301                       ASL     R1                ;GET INDEX
2719  023276  005261  061306               INC     BUFFER(R1)        ;BUILD HISTORY TABLE
2720  023302  005300                       DEC     R0                ;DECREMENT NUMBER OF SAMPLES
2721  023304  001366                       BNE     COLECT            ;BRANCH IF NOT DONE
2722
2723                                       ;THIS SECTION PERFORMS THE FOLLOWING CALCULATION:
2724
2725                                       ;TEMP = ((BUFFER) * 0) + ((BUFFER+2) * 1) + ((BUFFER+4) * 2) + ...
2726                                       ;             ((BUFFER+8188) * 4094) + ((BUFFER+8190) * 4095)
2727                                       ;
2728  023306  005005                       CLR     R5                ;SETUP INDEX
2729  023310  005037  024702               CLR     TEMPL             ;SETUP TO MULTIPLY
2730  023314  005037  024704               CLR     TEMPH
2731  023320  005037  024714     XBAR:     CLR     VMULH
2732  023324  016537  061306  023360       MOV     BUFFER(R5),1$
2733  023332  001431                       BEQ     2$
2734  023334  022737  040000  023360       CMP     #16384.,1$        ;TEST IF ALL COUNTS WERE THE SAME
2735  023342  001423                       BEQ     4$                ;BR IF YES AND TELL OPERATOR
2736  023344  010537  024712               MOV     R5,VMULL          ;GET CONVERTED VALUE
2737  023350  006237  024712               ASR     VMULL             ;   INTO VMULL
2738  023354  004537  025224               JSR     R5,MULTI          ;MULTIPLY THE VALUE IN VMUL(H/L) BY THE
2739                                                                 ;WORD FOLLOWING THIS INSTRUCTION AND
2740                                                                 ;RETURN THE RESULT IN R0 AND R1 WHERE
```

J 7

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 52-1
CVMNAC.P11   10-JUL-81 14:30   SETTLING TEST START-UP   SEQ 0087

```
2741                                                    ;R0 IS THE LEAST SIGNIFICANT WORD OF THE
2742                                                    ;DOUBLE PRECISION RESULT.
2743   023360  000000          1$:    0
2744   023362  060037  024702         ADD     R0,TEMPL
2745   023366  005537  024704         ADC     TEMPH
2746   023372  060137  024704         ADD     R1,TEMPH
2747   023376  100007                 BPL     2$               ;BRANCH IF NO OVERFLOW
2748   023400  004537  024416         JSR     R5,TOOBIG        ;CALC. OVERFLOW - TEMP(H/L) OVERFLOWED DURING
2749                                                            ;   CALCULATION.
2750   023404  046274                 EROVF
2751   023406  000137  024332         JMP     TOOBAD
2752   023412  000137  024334  4$:    JMP     TOGOOD           ;REPORT NO NOISE
2753   023416  005725          2$:    TST     (R5)+            ;BUMP INDEX
2754   023420  032705  020000         BIT     #BIT13,R5        ;DONE?
2755   023424  001735                 BEQ     XBAR
2756                             ;
2757                             ;DIVIDE RESULT BY 16384 LEAVING WHOLE PORTION IN TEMPH AND DECIMAL PART
2758                             ;IN TEMPL.  IN OTHER WORDS, THE NUMBER IS A DOUBLE PRECISION NUMBER WITH
2759                             ;A BINARY DECIMAL POINT BETWEEN TEMPH AND TEMPL.
2760                             ;
2761   023426  012700  000002         MOV     #2,R0            ;DIVIDE BY 16384(10)
2762   023432  006337  024702  3$:    ASL     TEMPL
2763   023436  006137  024704         ROL     TEMPH
2764   023442  005300                 DEC     R0
2765   023444  001372                 BNE     3$
2766
2767                             ;PERFORM CALCUALTION OF RMS**2
2768
2769                             ;THE FOLLOWING FORMULA IS USED:
2770                             ;  RMS2 = ((BUFFER) * (TEMP**2)) + ((BUFFER+2) * ((TEMP - 1) ** 2)) + ...
2771                             ;    ...  ((BUFFER+8190) * ((TEMP - 4095) ** 2))
2772                             ;
2773                             ;CLEAR INDEX AND RESULT FIELDS
2774                             ;
2775   023446  005005                 CLR     R5               ;SETUP INDEX
2776   023450  005037  024716         CLR     V1L              ;SETUP TO MULTIPLY
2777   023454  005037  024720         CLR     V1H
2778   023460  005037  024722         CLR  ,, V2L
2779   023464  005037  024724         CLR     V2H
2780                             ;
2781                             ;PROCESS ALL THE DATA BY THE FORMULA FOR RMS2 ABOVE
2782                             ;
2783   023470  016537 061306 023560 RMS2:  MOV  BUFFER(R5),2$  ;GET NUMBER OF HITS
2784   023476  001461                 BEQ     3$               ;BRANCH IF NO HITS - NOTHING TO PROCESS
2785   023500  010501                 MOV     R5,R1            ;GET A/D CONVERTER VALUE FOR HITS
2786   023502  006201                 ASR     R1
2787   023504  013737  024702 024712  MOV     TEMPL,VMULL      ;MOVE AVERAGE CONVERTER VALUE TO WORK AREA
2788   023512  013737  024704 024714  MOV     TEMPH,VMULH
2789   023520  160137  024714         SUB     R1,VMULH         ;SUBTRACT A/D CONVERTER VALUE FROM AVERAGE
2790   023524  100011                 BPL     1$               ;BRANCH IF RESULT POSITIVE -
2791   023526  005137  024712         COM     VMULL            ;  OTHERWISE, TAKE THE ABSOLUTE VALUE OF RESULT
2792   023532  005137  024714         COM     VMULH
2793   023536  062737 000001 024712   ADD     #1,VMULL
2794   023544  005537  024714         ADC     VMULH
2795   023550  004737  025654  1$:    JSR     PC,SQUARE        ;SQUARE NUMBER - PUTTING RESULT INTO SQR0,SQR1,
2796                                                            ;  AND SQR3
```

```
2797   023554  004537  025404              JSR     R5,XMULT            ;EXTENDED MULTIPLICATION
2798                                        ;   MULTIPLY TRIPLE PRECISION NUMBER IN SQRX BY
2799                                        ;   THE WORD ARGUMENT FOLLOWING THIS INSTRUCTION
2800                                        ;
2801                                        ;   RETURNS A QUADRUPLE PRECISION NUMBER IN
2802                                        ;   XMULY.   WHERE Y CAN VARY FROM 0 TO 3.
2803   023560  000000              2$:      0
2804   023562  063737  024734  024716      ADD     XMUL0,V1L           ;ADD IN RESULT
2805   023570  005537  024720              ADC     V1H
2806   023574  063737  024736  024720      ADD     XMUL1,V1H
2807   023602  005537  024722              ADC     V2L
2808   023606  063737  024740  024722      ADD     XMUL2,V2L
2809   023614  005537  024724              ADC     V2H
2810   023620  063737  024742  024724      ADD     XMUL3,V2H
2811   023626  100005                      BPL     3$                  ;BRANCH IF NO OVERFLOW
2812   023630  004537  024416              JSR     R5,TOOBIG           ;CALC. OVERFLOW
2813   023634  046274                      EROVF
2814   023636  000137  024332              JMP     TOOBAD
2815   023642  005725              3$:      TST     (R5)+               ;BUMP INDEX TO NEXT ENTRY IN BUFFER
2816   023644  032705  020000              BIT     #BIT13,R5           ;DONE?
2817   023650  001707                      BEQ     RMS2                ; NO
2818                                        ;
2819                                        ;DIVIDE RESULT FROM RMS2 CALCULATION BY 16384.
2820                                        ;
2821                                        ;V2H WILL CONTAIN THE INTEGER PORTION OF THE RESULT AND V1L, V1H, AND
2822                                        ;V2L WILL CONTAIN THE FRACTION PARTS WHERE V1L IS THE LEAST SIGNIFICANT
2823                                        ;WORD.
2824                                        ;
2825   023652  012700  000002              MOV     #2,R0               ;DIVIDE BY 16384(10)
2826   023656  006337  024716      4$:      ASL     V1L
2827   023662  006137  024720              ROL     V1H
2828   023666  006137  024722              ROL     V2L
2829   023672  006137  024724              ROL     V2H
2830   023676  100005                      BPL     5$
2831   023700  004537  024416              JSR     R5,TOOBIG           ;REPORT ERROR
2832   023704  046274                      EROVF
2833   023706  000137  024416              JMP     TOOBIG
2834   023712  005300              5$:      DEC     R0
2835   023714  001360                      BNE     4$
2836                                        ;
2837                                        ;ROUND OFF THE RESULT TO A TRIPLE PRECISION NUMBER WITH V2H CONTAINING THE
2838                                        ;INTEGER PART.
2839                                        ;
2840   023716  062737  100000  024716      ADD     #BIT15,V1L          ;ROUND OFF NUMBER
2841   023724  005537  024720              ADC     V1H
2842   023730  005537  024722              ADC     V2L
2843   023734  005537  024724              ADC     V2H
2844                                        ;
2845                                        ;MOVE RMS2 TO SQRX IN PREPARATION FOR SQUARE ROOT.
2846                                        ;CHECK RMS2 TO SEE IF THE RESULT IS ZERO IF SO, REPORT ERROR.
2847                                        ;
2848   023740  013737  024720  024726      MOV     V1H,SQR0            ;SET UP TO FIND SQUARE ROOT
2849   023746  013737  024722  024730      MOV     V2L,SQR1
2850   023754  013737  024724  024732      MOV     V2H,SQR2
2851   023762  013700  024726              MOV     SQR0,R0             ;CHECK FOR ZERO
2852   023766  053700  024730              BIS     SQR1,R0
```

```
2853  023772  053700  024732          ~       BIS    SQR2,R0
2854  023776  001005                          BNE    6$              ;BR IF NON-ZERO
2855  024000  004537  024416                  JSR    R5,TOOBIG       ;REPORT ERROR
2856  024004  046152                          ERDIV
2857  024006  000137  024332                  JMP    TOOBAD
2858
2859                                    ;FIND THE SQUARE ROOT OF THE NUMBER IN SQRX USING NEWTONS METHOD.
2860
2861                                    ;USING THE FORMULA AS FOLLOWS:
2862
2863                                    ; RESULT = NUMBER / GUESS
2864                                    ; NEXT-GUESS = (RESULT + GUESS) / 2
2865                                    ; IF GUESS <> NEXT-GUESS THEN GUESS = NEXT-GUESS \
2866                                    ;                                  REPEAT UNTIL GUESS = NEXT-GUESS
2867
2868                                    ;WHEN FINISHED, THE RESULT IS IN VMULL, VMULH WHERE VMULH CONTAINS THE INTEGER
2869                                    ;PART AND VMULL CONTAINS THE FRACTIONAL PART.
2870
2871  024012  005002          6$:          CLR    R2              ;GET FIRST GUESS
2872  024014  012703  004000                  MOV    #2048.,R3
2873  024020  010237  024716      SQRR:       MOV    R2,V1L          ;SETUP FOR DIVISION
2874  024024  010337  024720                  MOV    R3,V1H
2875  024030  004737  025554                  JSR    PC,XDIVI        ;GO DO DIVISION
2876                                    ;   DIVIDE TRIPLE PRECISION NUMBER SQRX
2877                                    ;   BY DOUBLE PRECISION NUMBER V1L,V1H GIVING
2878                                    ;   DOUBLE PRECISION RESULT VMULL,VMULH.
2879  024034  060237  024712                  ADD    R2,VMULL        ;GET NEXT GUESS - BY ADDING THE LAST GUESS
2880                                    ;   TO THE RESULT OF THE DIVIDE IN VMULL,VMULH
2881                                    ;   AND DIVIDING BY TWO (2)
2882  024040  005537  024714                  ADC    VMULH
2883  024044  060337  024714                  ADD    R3,VMULH
2884  024050  006237  024714                  ASR    VMULH
2885  024054  006037  024712                  ROR    VMULL
2886  024060  163702  024712                  SUB    VMULL,R2        ;FIND ABSOLUTE DIFFERENCE BETWEEN GUESS
2887                                    ;   AND NEXT GUESS
2888  024064  005603                          SBC    R3              ;REMOVE BORROW FROM NEXT GUESS
2889  024066  163703  024714                  SUB    VMULH,R3
2890  024072  102413                          BVS    1$              ;BRANCH IF TOO FAR OFF
2891  024074  100005                          BPL    2$              ;BR IF ALREADY PLUS
2892  024076  005102                          COM    R2              ;MAKE POSITIVE
2893  024100  005103                          COM    R3              ; NEXT VALUE
2894  024102  062702  000001                  ADD    #1,R2           ;  CORRECTION
2895  024106  005503                          ADC    R3
2896  024110  001004          2$:          BNE    1$              ;BR IF MORE
2897  024112  005702                          TST    R2              ;TEST IF CLOSE
2898  024114  100402                          BMI    1$              ;BR
2899  024116  005302                          DEC    R2
2900  024120  003414                          BLE    PRMS            ;BR IF DONE
2901  024122  013702  024712      1$:          MOV    VMULL,R2        ;SETUP FOR NEXT GUESS CALCULATION
2902  024126  013703  024714                  MOV    VMULH,R3
2903  024132  010200                          MOV    R2,R0           ;TEST FOR DIVISION BY ZERO
2904  024134  050300                          BIS    R3,R0
2905  024136  001330                          BNE    SQRR
2906  024140  004537  024416                  JSR    R5,TOOBIG       ;CALC. ERROR
2907  024144  046152                          ERDIV
2908  024146  000137  024332                  JMP    TOOBAD
```

```
2909
2910                                      ;NOW THAT THE RMS NUMBER CRUNCHING AND COLLECTION IS DONE, TEST THE RESULTS
2911                                      ;
2912                                      ;CHECK TO SEE IF THERE WAS AN ERROR ENCOUNTERED DURING THE RMS NOISE
2913                                      ;CALCULATION.  IF THERE WAS AN ERROR, BADCAL WILL BE NON-ZERO.
2914                                      ;
2915   024152  005737  024700     PRMS:   TST    BADCAL              ;TEST IF A BAD CALCULATION OCCURRED
2916   024156  001402                     BEQ    1$                  ;BR IF NOT
2917   024160  000137  024332             JMP    TOOBAD              ;DONT TEST IF WITHIN LIMITS
2918                                      ;
2919                                      ; CHECK TO SEE IF RMS NOISE IS WITHIN LIMITS AND REPORT TO OPERATOR
2920                                      ;
2921   024164  004537  024524     1$:     JSR    R5,ERCHKG           ;CHECK IF WITHIN LIMITS
2922   024170  000000             AGCHRA: 0                          ;MSW OF RMS LIMIT
2923   024172  000000             AGCHRB: 0          :               ;LSW OF RMS LIMIT
2924   024174  046551                     RMSNOI                     ;TEXT POINTER FOR RMS NOISE
```

```
2926
2927                                         ;NOW TAKE THE COLLECTED DATA AND DETERMINE THE PEAK NUMBERS.
2928                                         ;DIVIDE ALL ENTRIES IN 'BUFFER' BY 64 (DECIMAL).
2929                                         ;
2930   024176  012700  017776                       MOV      #<4095.*2>,R0     ;GET OFFSET TO LAST ENTRY
2931   024202  006260  061306        1$:             ASR      BUFFER(R0)        ;DIVIDE COUNT BY 64
2932   024206  006260  061306                       ASR      BUFFER(R0)
2933   024212  006260  061306                       ASR      BUFFER(R0)
2934   024216  006260  061306                       ASR      BUFFER(R0)
2935   024222  006260  061306                       ASR      BUFFER(R0)
2936   024226  006260  061306                       ASR      BUFFER(R0)
2937   024232  005300                               DEC      R0
2938   024234  005300                               DEC      R0
2939   024236  100361                               BPL      1$
2940
2941                                         ;NOW CALULATE PEAK NOISE
2942                                         ;
2943                                         ;DISCOUNT ONE (1) HIT AT THE ENDS.
2944                                         ;
2945   024240  004737  024360        PEAKNN: JSR      PC,PEAKN          ;FIND THE FIRST AND LAST BUFFER ENTRY
2946   024244  005361  061306                DEC      BUFFER(R1)        ;REMOVE 1 FROM THE FIRST ENTRY
2947   024250  005360  061306                DEC      BUFFER(R0)        ;REMOVE 1 FROM THE LAST ENTRY
2948                                         ;
2949                                         ;FIND NEW PEAKS AND SUBTRACT LOWEST A/D CONVERTER VALUE FROM HIGHEST
2950                                         ;AND DIVIDE THE RESULT BY TWO (2) GIVING PEAK NOISE.
2951                                         ;
2952   024254  004737  024360                JSR      PC,PEAKN          ;FIND THE NEW FIRST AND LAST BUFFER ENTRY
2953   024260  160100                3$:      SUB      R1,R0             ;GET PEAK NOISE
2954   024262  006200                         ASR      R0
2955   024264  010037  024714                 MOV      R0,VMULH
2956   024270  005037  024712                 CLR      VMULL
2957   024274  006237  024714                 ASR      VMULH
2958   024300  006037  024712                 ROR      VMULL
2959   024304  005737  024700                 TST      BADCAL            ;TEST IF BAD CALCULATION OCCURRED
2960   024310  001402                          BEQ      4$                ;BR IF NONE
2961   024312  000137  024332                  JMP      TOOBAD            ;IF SOME DONT TEST AGAINST LIMITS
2962                                         ;
2963                                         ;CHECK TO SEE IF PEAK NOISE IS WITHIN LIMITS AND REPORT TO OPERATOR
2964                                         ;
2965   024316  004537  024524        4$:      JSR      R5,ERCHKG         ;CHECK IF NOISE IS WITHIN LIMITS
2966   024322  000000              AGCHPA: 0                           ;MSW OF PEAK LIMIT
2967   024324  000000              AGCHPB: 0                           ;LSW OF PEAK LIMIT
2968   024326  046565                        PKNOI                     ;TEXT POINTER FOR PEAK NOISE
2969   024330  000207                        RTS      PC                ;NORMAL AMOUNT OF NOISE EXIT
2970                                         ;COME HERE IF TOO MUCH NOISE WAS DETECTED
2971   024332  000207              TOOBAD: RTS      PC                ;TOO MUCH NOISE EXIT
2972                                         ;COME HERE IF ALL COUNTS ENDED UP IN ONE BIN
2973   024334  005737  001624        TOGOOD: TST      QUIET             ;TEST IF QUIET MODE
2974   024340  001006                        BNE      1$                ;BR IF QUIET
2975   024342  104401  040640                TYPE     ,NONOIS           ;TELL OPERATOR NO NOISE DETECTED
2976   024346  004737  036314                JSR      PC,PSONOI         ;REPORT CHANNEL INFO
2977   024352  104401  046620                TYPE     ,OKMSG            ;ADD ''OK'' JUST FOR SHOW
2978   024356  000207              1$:      RTS      PC                ;NO NOISE DETECTED EXIT
2979
2980                                         ;SUBROUTINE TO FIND THE FIRST AND LAST NON ZERO BUFFER ENTRIES
2981                                         ;FIRST NON-ZERO ENTRY IN 'BUFFER' IS RETURNED IN R1 AND LAST
```

```
2982                                    ;NON-ZERO ENTRY IN 'BUFFER' IS RETURNED IN R0.
2983                                    ;
2984                                    ; FIND FIRST NON-ZERO ENTRY IN 'BUFFER' AND SAVE IN R1
2985                                    ;
2986    024360  005000           PEAKN:  CLR     R0              ;LOAD POINTER TO THE START OF THE BUFFER
2987    024362  005760  061306   1$:TST  BUFFER(R0)      ;WAS THERE A HIT HERE?
2988    024366  001002                   BNE     2$              ;BR IF YES
2989    024370  005720                   TST     (R0)+           ;GO TO NEXT STATE AND TRY AGAIN
2990    024372  000773                   BR      1$              ;
2991    024374  010001           2$:     MOV     R0,R1           ;COPY R0
2992                                    ;
2993                                    ; FIND LAST NON-ZERO ENTRY IN 'BUFFER'
2994                                    ;
2995    024376  012700  017776           MOV     #17776,R0       ;LOAD POINTER TO END OF THE BUFFER
2996    024402  005760  061306   3$:     TST     BUFFER(R0)      ;WAS THERE A HIT HERE?
2997    024406  001002                   BNE     4$              ;BR IF YES
2998    024410  005740                   TST     -(R0)           ;GO TO PREVIOUS STATE AND TRY AGAIN
2999    024412  000773                   BR      3$
3000    024414  000207           4$:     RTS     PC              ;EXIT
3001
```

```
3003                              ;SUBROUTINE TO HANDLE CALCULATION ERRORS
3004                              ; CODE COMES HERE WHEN THERE WAS A CALCULATION ERROR - IE UNEXPECTED
3005                              ; ARITHMETIC OVERFLOW OCCURED - IN THE NOISE COMPUTATION.
3006                              ;
3007   024416  010537  024522    TOOBIG: MOV    R5,11$            ;SAVE CALLING ADDRESS
3008   024422  162737  000004  024522   SUB    #4,11$            ;CORRECT THE VALUE
3009   024430  013737  024522  024700   MOV    11$,BADCAL        ;LOAD LOCATION OF ERROR INTO FLAG
3010   024436  012537  024460          MOV    (R5)+,10$         ;SAVE TRAILING ARGUMENT
3011   024442  032777  020000  154470   BIT    #SW13,@SWR        ;TEST IF INHIBIT REPORT IS SET
3012   024450  001017                   BNE    1$                ;BR IF SET
3013   024452  104401  046103           TYPE   ,EXCNOI           ;REPORT EXCESSIVE NOISE CAUSED FATAL MATH ERROR
3014   024456  104401                   TYPE                     ;TELL OPER THE BAD NEWS
3015   024460  000000            10$:   0                        ;POINTER TO ASCII TEXT MESSAGE
3016   024462  013746  024522           MOV    11$,-(SP)         ;MOVE BAD PC TO STACK
3017   024466  104402                   TYPOC                    ;AND ADD TO ERROR TYPEOUT
3018   024470  104401  046601           TYPE   ,CHAN             ;ADD CHANNEL TEXT
3019   024474  013746  001556           MOV    CHANL,-(SP)       ;AND CHANNEL NUMBER
3020   024500  104403                   TYPOS
3021   024502    002     000            .BYTE  2,0
3022   024504  104401  001165           TYPE   ,$CRLF            ;ADD CRLF
3023   024510  004737  055704    1$:    JSR    PC,WHICHV         ;DETERMINE THE FAILING UNIT MASK
3024   024514  005237  001112           INC    $ERTTL            ;UPDATE ERROR TOTAL
3025   024520  000205                   RTS    R5                ;EXIT
3026   024522  000000            11$:   0
3027                              ;SUBROUTINE TO CHECK WITHIN LIMITS
3028   024524  012537  024674    ERCHKG: MOV    (R5)+,10$         ;GET MSW VALUE
3029   024530  012537  024676           MOV    (R5)+,11$         ;GET LSW VALUE
3030   024534  012537  024644           MOV    (R5)+,72$         ;GET NOISE TEXT POINTER
3031   024540  013737  024712  024706   MOV    VMULL,VMULLS      ;SAVE RESULTS FOR LATER
3032   024546  013737  024714  024710   MOV    VMULH,VMULHS      ;
3033   024554  012737  046620  024664   MOV    #OKMSG,77$        ;PRIME THE PASS/FAIL TEXT
3034   024562  023737  024674  024710   CMP    10$,VMULHS        ;COMPARE MSW
3035   024570  100406                   BMI    1$                ;BR IF EXCESSIVE
3036   024572  001004                   BNE    3$                ;BR IF OK
3037   024574  023737  024676  024706   CMP    11$,VMULLS        ;COMPARE LSW
3038   024602  101401                   BLOS   1$                ;BR IF EXCESSIVE
3039   024604  000407            3$:    BR     70$
3040   024606  012737  047505  024664  1$:    MOV    #ERMSG,77$        ;LOAD ERROR TEXT POINTER
3041   024614  004737  055704           JSR    PC,WHICHV         ;DETERMINE UNIT
3042   024620  005237  001112           INC    $ERTTL            ;UPDATE ERROR COUNT
3043   024624  005737  001624    70$:   TST    QUIET             ;TEST QUIET FLAG
3044   024630  001404                   BEQ    71$               ;BR IF CLEARED
3045   024632  022737  046620  024664   CMP    #OKMSG,77$        ;TEST IF ANY ERROR
3046   024640  001414                   BEQ    2$                ;BR IF NOT
3047   024642  104401            71$:   TYPE                     ;TELL OPERATOR THE RMS/PEAK NEWS
3048   024644  046551            72$:   RMSNOI ;PKNOI            ;RMS/PEAK NOISE TEXT POINTER
3049   024646  004737  026050           JSR    PC,PRGAIN         ;REPORT RESULTS
3050   024652  104401  050644           TYPE   ,MLSB             ;ADD LSB TEXT
3051   024656  004737  036314           JSR    PC,PSONOI         ;ADD CHANNEL REPORT
3052   024662  104401                   TYPE                     ;TELL OPER THE PASS/FAIL NEWS
3053   024664  046620            77$:   OKMSG  ;ERMSG            ;PASS/FAIL TEXT POINTER
3054   024666  000240                   NOP
3055   024670  000240                   NOP
3056   024672  000205            2$:    RTS    R5                ;EXIT
3057   024674  000000            10$:   0
3058   024676  000000            11$:   0
```

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 54-1
CVMNAC.P11   10-JUL-81 14:30   SETTLING TEST START-UP   SEQ 0094

D 8

```
3059  024700  000000          BADCAL:  0            ;BAD CALC. FLAG
3060  024702  000000          TEMPL:   0            ;TEMPORARY STORAGE USED IN RMS CALCULATION
3061  024704  000000          TEMPH:   0            ;TEMPORARY STORAGE USED IN RMS CALCULATION
3062  024706  000000          VMULLS:  0            ;TEMP LOC. OF VMULL
3063  024710  000000          VMULHS:  0            ;TEMP LOC. OF VMULH
3064  024712  000000          VMULL:   0
3065  024714  000000          VMULH:   0
3066  024716  000000          V1L:     0
3067  024720  000000          V1H:     0
3068  024722  000000          V2L:     0
3069  024724  000000          V2H:     0
3070  024726  000000          SQR0:    0            ;USED FOR SQUARE ROUTINE
3071  024730  000000          SQR1:    0            ;USED FOR SQUARE ROUTINE
3072  024732  000000          SQR2:    0            ;USED FOR SQUARE ROUTINE
3073  024734  000000          XMUL0:   0
3074  024736  000000          XMUL1:   0
3075  024740  000000          XMUL2:   0
3076  024742  000000          XMUL3:   0
```

```
3078
3079                                    ;DOUBLE PRECISION DIVIDER FOR DECIMAL DIVISION OF TWO DOUBLE
3080                                    ;PRECISION NUMBERS.
3081                                    ;      ENTER WITH DIVIDEND IN V2 DIVISOR IN V1
3082                                    ;      RETURNS WHOLE NUMBER IN VMULH, DECIMAL PART IN VMULL
3083                                    ;      REMAINDER IN V2
3084                                    ;NOTE: NUMBER RETURNED IN V2 IS SHIFTED LEFT 16 PLACES FROM THE ORIGINAL.
3085                                    ;      THEREFORE, IF NUMBERS WERE INTEGER, V2L CONTAINS FRACTIONAL PART
3086                                    ;      OF REMAINDER ON RETURN.
3087                                    ;CALLING SEQUENCE:
3088                                    ;      CALL DIVI
3089                                    ;NOTE: V2H, AND V1H IS THE HIGH ORDER OR MOST SIGNIFIGANT BITS.
3090                                    ;      V2L, AND V1L IS THE LOW ORDER OR LEAST SIGNIFIGANT BITS.
3091                                    ;
3092                                    ; SETUP RESULT SIGN FLAG ON TOP OF STACK AND MAKE DIVIDEND AND DIVISOR
3093                                    ; POSITIVE.
3094
3095    024744  012700  000020   DIVI:  MOV    #16.,R0        ;SET UP DECIMAL COUNT
3096    024750  005037  024714          CLR    VMULH          ;CLEAR WHOLE PART OF RESULT
3097    024754  005037  024712          CLR    VMULL          ;CLEAR DECIMAL PART OF RESULT
3098    024760  005046                  CLR    -(SP)          ;CLEAR SIGN OF RESULT
3099    024762  005737  024720          TST    V1H            ;IS V1 NEGATIVE?
3100    024766  100012                  BPL    1$             ;NO
3101    024770  005216                  INC    (SP)           ;INCREMENT SIGN FLAG
3102    024772  005137  024716          COM    V1L            ;TWO'S COMPLEMENT V1
3103    024776  005137  024720          COM    V1H
3104    025002  062737  000001  024716  ADD    #1,V1L
3105    025010  005537  024720          ADC    V1H
3106    025014  005737  024724   1$:    TST    V2H            ;IS V2 NEGATIVE?
3107    025020  100012                  BPL    2$             ;NO
3108    025022  005316                  DEC    (SP)           ;DECREMENT SIGN FLAG
3109    025024  005137  024716          COM    V1L            ;TWO'S COMPLEMENT V2
3110    025030  005137  024720          COM    V1H
3111    025034  062737  000001  024716  ADD    #1,V1L
3112    025042  005537  024720          ADC    V1H
3113                                    ;
3114                                    ; DIVIDE WHOLE NUMBER BY REPEATED SUBTRACTION UNTIL V2 GOES NEGATIVE.
3115                                    ;
3116    025046  163737  024716  024722  2$:  SUB    V1L,V2L    ;SUBTRACT V1 FROM V2
3117    025054  005637  024724          SBC    V2H
3118    025060  163737  024720  024724  SUB    V1H,V2H
3119    025066  100406                  BMI    3$             ;BRANCH IF SUBTRACT FAILED
3120    025070  005237  024714          INC    VMULH          ;ADD ONE TO WHOLE NUMBER RESULT
3121    025074  100364                  BPL    2$             ;TRY ANOTHER SUBTRACTION
3122                                    ;
3123                                    ; ERROR OCCURED WHILE ATTEMPTING TO DIVIDE V2 BY V1
3124                                    ;
3125    025076  004537  024416          JSR    R5,TOOBIG      ;CALC. OVERFLOW
3126    025102  046152                  ERDIV
3127                                    ;
3128                                    ; ADD V1 TO V2 TO RETURN V2 TO A POSITIVE NUMBER
3129                                    ;
3130    025104  063737  024716  024722  3$:  ADD    V1L,V2L    ;ADD V1 TO V2
3131    025112  005537  024724          ADC    V2H
3132    025116  063737  024720  024724  ADD    V1H,V2H
3133                                    ;
```

```
3134                                    ; CALCULATE 16 BIT FRACTIONAL PART (VMULL) BY RESTORING DIVISION.
3135                                    ; IE IF SUBTRACTION CAUSES A CHANGE OF SIGN, UNDO THE OPERATION.
3136                                    ;
3137    025124  005300          4$:     DEC     RO                      ;DECREMENT DECIMAL COUNT
3138    025126  100422                  BMI     5$                      ;BRANCH IF DONE
3139    025130  006337  024722          ASL     V2L                     ;MULTIPLY V2 BY 2
3140    025134  006137  024724          ROL     V2H
3141    025140  006337  024712          ASL     VMULL                   ;MULTIPY VMULL BY 2
3142    025144  163737  024716  024722  SUB     V1L,V2L                 ;SUBTRACT V1 FROM V2
3143    025152  005637  024724          SBC     V2H
3144    025156  163737  024720  024724  SUB     V1H,V2H
3145    025164  100747                  BMI     3$                      ;BRANCH IF SUBTRACTION FAILED
3146    025166  005237  024712          INC     VMULL                   ;INCREMENT DECIMAL RESULT
3147    025172  000754                  BR      4$                      ;TRY AGAIN
3148                                    ;
3149                                    ; CHECK THE SIGN OF THE RESULT - IF TOP OF STACK NON-ZERO, COMPLEMENT
3150                                    ; THE RESULT MAKING IT NEGATIVE.
3151                                    ;
3152    025174  005726          5$:     TST     (SP)+                   ;TEST SIGN FLAG
3153    025176  001411                  BEQ     6$                      ;NUMBER IS POSITIVE
3154    025200  005137  024712          COM     VMULL                   ;TWO'S COMPLEMENT RESULT
3155    025204  005137  024714          COM     VMULH
3156    025210  062737  000001  024712  ADD     #1,VMULL
3157    025216  005537  024714          ADC     VMULH
3158    025222  000207          6$:     RTS     PC                      ;RETURN FROM DIVI
```

```
3160                                  ;ROUTINE TO MULTIPLY TWO NUMBERS
3161                                  ;CALL:  JSR     R5,MULTI
3162                                  ;       MULTIPLIER
3163
3164                                  ;MULTIPLIES VMUL BY MULTIPLIER, RESULT IN R0 & R1 WITH THE LOW BYTE   ı
3165                                  ;       IN R0 HIGH BYTE IN R1
3166                                  ;THE MULTIPLIER IS ALWAYS AN UNSIGNED NUMBER.
3167
3168   025224  005046                 MULTI:  CLR     -(SP)            ;CLEAR SIGN FLAG
3169   025226  005000                         CLR     R0               ;CLEAR WORK REGISTERS
3170   025230  005001                         CLR     R1
3171   025232  012702  100000                 MOV     #BIT15,R2        ;SETUP TEST BIT, MULTIPLIER IS UNSIGNED
3172                                  ;
3173                                  ; TEST SIGN OF MULTIPLICAND AND MAKE TOP OF STACK NON-ZERO IF NEGATIVE.
3174                                  ; TAKE ABSOLUTE VALUE OF MULTIPLICAND.
3175                                  ;
3176   025236  005737  024714                 TST     VMULH            ;TEST SIGN
3177   025242  100012                         BPL     1$               ;BRANCH IF POSITIVE
3178   025244  005216                         INC     (SP)             ;INCREMENT SIGN FLAG
3179   025246  005137  024712                 COM     VMULL            ;TWO'S COMPLEMENT NUMBER
3180   025252  005137  024714                 COM     VMULH
3181   025256  062737  000001  024712         ADD     #1,VMULL
3182   025264  005537  024714                 ADC     VMULH
3183
3184   025270  006300                 1$:     ASL     R0               ;MULTIPLY RESULT BY 2
3185   025272  006101                         ROL     R1
3186   025274  103003                         BCC     4$               ;BR IF NO **MULTIPLICATION ERROR**
3187
3188                                  ; ERROR OCCURED DURING MULTIPLICATION OPERATION
3189                                  ; WHILE ATTEMPTING TO MULTIPLY RESULT BY 2.
3190                                  ;
3191   025276  004537  024416                 JSR     R5,TOOBIG        ;REPORT ERROR
3192   025302  046220                         ERMUL
3193                                  ;
3194                                  ; TEST BIT IN MULTIPLIER AND ADD MULTIPLICAND TO RESULT IF NON-ZERO.
3195
3196   025304  030215                 4$:     BIT     R2,(R5)          ;TEST MULTIPLIER BIT
3197   025306  001411                         BEQ     2$               ;BRANCH IF BIT IS CLEAR
3198   025310  063700  024712                 ADD     VMULL,R0         ;ADD NUMBER TO RESULT
3199   025314  005501                         ADC     R1
3200   025316  063701  024714                 ADD     VMULH,R1
3201   025322  103003                         BCC     2$               ;BR IF NO **MULTIPLICATION ERROR**
3202
3203                                  ; ERROR OCCURED DURING MULTIPLICATION OPERATION
3204                                  ; WHILE ATTEMPTING TO ADD VMUL(H/L) TO RESULT.
3205                                  ;
3206   025324  004537  024416                 JSR     R5,TOOBIG        ;REPORT ERROR
3207   025330  046220                         ERMUL
3208
3209                                  ; SHIFT TEST BIT RIGHT AND LOOP IF NON-ZERO (MULTIPLICATION IS DONE
3210                                  ; WHEN R2 GOES TO ZERO).
3211
3212   025332  000241                 2$:     CLC                      ;SHIFT TEST BIT RIGHT
3213   025334  006002                         ROR     R2
3214   025336  001354                         BNE     1$               ;BRANCH IF NOT DONE
3215                                  ;
```

```
3216                                 ; BUMP RETURN POINTER AND ADJUST SIGN OF RESULT IN VMUL(H/L).
3217                                 ;
3218  025340  005725                         TST     (R5)+              ;BUMP RETURN POINTER
3219  025342  005726                         TST     (SP)+              ;TEST SIGN FLAG
3220  025344  001416                         BEQ     3$                 ;BRANCH IF POSITIVE
3221  025346  005100                         COM     R0                 ;TWO'S COMPLEMENT THE RESULT
3222  025350  005101                         COM     R1
3223  025352  062700  000001                 ADD     #1,R0
3224  025356  005501                         ADC     R1
3225  025360  005137  024712                 COM     VMULL              ;TWO'S COMPLEMENT NUMBER
3226  025364  005137  024714                 COM     VMULH
3227  025370  062737  000001  024712         ADD     #1,VMULL
3228  025376  005537  024714                 ADC     VMULH
3229  025402  000205                  3$:     RTS     R5                 ;RETURN FROM MULTI
3230
```

```
3232                                          ;EXTENDED MULTIPLY
3233                                          ;ROUTINE TO MULTIPLY A TRIPLE PRECISION NUMBER
3234                                          ;BY A SINGLE PRECISION NUMBER GIVING A QUADUPLE PRECISION RESULT
3235                                          ;
3236                                          ;CALLED BY:
3237                                          ;        CALL    XMULT
3238                                          ;        MULTIPLIER
3239                                          ;
3240                                          ;MULTIPLIES THE CONTENTS OF SQR(2/1/0) BY MULTIPLIER GIVING XMUL(3/2/1/0)
3241                                          ;       NUMBER IN "()" ARE MOST TO LEAST SIGNIFICAND - LEFT TO RIGHT
3242                                          ;
3243                                          ; CLEAR RESULT AREA
3244                                          ;
3245    025404  005037  024734       XMULT:   CLR     XMUL0           ;CLEAR RESULT
3246    025410  005037  024736                CLR     XMUL1
3247    025414  005037  024740                CLR     XMUL2
3248    025420  005037  024742                CLR     XMUL3
3249
3250                                          ; MULTIPLY SQR0 BY MULTIPLIER PUTING RESULT INTO XMUL(1/0)
3251                                          ;
3252    025424  012537  024712                MOV     (R5)+,VMULL     ;SETUP FOR MULTIPLICATION
3253    025430  005037  024714                CLR     VMULH
3254    025434  013737  024726  025446        MOV     SQR0,1$
3255    025442  004537  025224                JSR     R5,MULTI        ;GET FIRST TERM
3256    025446  000000                1$:     0
3257    025450  010037  024734                MOV     R0,XMUL0        ;SAVE FIRST RESULT
3258    025454  010137  024736                MOV     R1,XMUL1
3259
3260                                          ; MULTIPLY SQR1 BY MULTIPLIER ADDING RESULT TO XMUL(2/1)
3261                                          ;
3262    025460  013737  024730  025472        MOV     SQR1,2$         ;PREPARE FOR SECOND MULTIPLICATION
3263    025466  004537  025224                JSR     R5,MULTI        ;GET SECOND TERM
3264    025472  000000                2$:     0
3265    025474  060037  024736                ADD     R0,XMUL1        ;ADD TO FIRST RESULT (SHIFTED)
3266    025500  005537  024740                ADC     XMUL2
3267    025504  060137  024740                ADD     R1,XMUL2
3268
3269                                          ; MULTIPLY SQR2 BY MULTIPLIER ADDING RESULT TO XMUL(3/2)
3270                                          ;
3271    025510  013737  024732  025524        MOV     SQR2,3$         ;PREPARE FOR THIRD MULTIPLICATION
3272    025516  000240                        NOP                     ;**FOR DEBUG**
3273    025520  004537  025224                JSR     R5,MULTI        ;GET THIRD TERM
3274    025524  000000                3$:     0
3275    025526  060037  024740                ADD     R0,XMUL2        ;ADD TO FIRST & SECOND (SHIFTED)
3276    025532  005537  024742                ADC     XMUL3
3277    025536  060137  024742                ADD     R1,XMUL3
3278    025542  100003                        BPL     4$              ;BR IF NO ERROR IN MULTIPLICATION
3279
3280                                          ; ERROR OCCURED IN EXTENDED MULTIPLY OPERATION
3281                                          ; RESULT XMUL(3/2/1/0) OVERFLOWED.
3282                                          ;
3283    025544  004537  024416                JSR     R5,TOOBIG       ;REPORT ERROR
3284    025550  046220                        ERMUL
3285    025552  000205                4$:     RTS     R5              ;BACK TO WHERE WE CAME
3286
3287                                          ;ROUTINE TO DIVIDE A TRIPLE PRECISION NUMBER
```

J 8

```
3288                                    ;BY A DOUBLE PRECISION NUMBER GIVING A DOUBLE PRECISION RESULT
3289                                    ;
3290                                    ;CALLING SEQUENCE:
3291                                    ;      CALL    XDIVI
3292                                    ;
3293                                    ;DIVIDES SQR(2/1/0) BY V1(H/L) RETURNING RESULT IN VMUL(H/L)
3294                                    ;      NUMBER/LETTERS IN "()" ARE MOST TO LEAST SIGNIFICAND - LEFT TO RIGHT
3295                                    ;
3296                                    ; DIVIDE SQR(2/1) BY V1(H/L)
3297
3298   025554  013737  024730  024722   XDIVI:  MOV    SQR1,V2L         ;SETUP FOR FIRST DIVIDE
3299   025562  013737  024732  024724           MOV    SQR2,V2H
3300   025570  004737  024744                    JSR    PC,DIVI         ;GET FIRST RESULT
3301
3302                                    ; CHECK FOR ARITHMETIC OVERFLOW
3303                                    ;
3304   025574  005737  024714                    TST    VMULH           ;DID OVERFLOW OCCUR?
3305   025600  001403                             BEQ    1$              ;NO
3306
3307                                    ; ERROR OCCURED WHILE ATTEMTING TO DIVIDE SQR(2/1) BY V1(H/L)
3308                                    ;
3309   025602  004537  024416                    JSR    R5,TOOBIG       ;REPORT ERROR
3310   025606  046152                             ERDIV
3311
3312                                    ; SAVE RESULT OF FIRST DIVISION IN TEMPH
3313                                    ;
3314   025610  013737  024712  024704   1$:     MOV    VMULL,TEMPH     ;SAVE FIRST RESULT
3315
3316                                    ; ADD SQR0 TO V2L AND ANY CARRY RESULTING TO V2H.  NOTE:
3317                                    ; V2(H/L) WAS SHIFTED 16 PLACES LEFT BY THE PRECEDING CALL TO DIVI.
3318                                    ;
3319   025616  063737  024726  024722           ADD    SQR0,V2L        ;SETUP FOR SECOND DIVIDE
3320   025624  005537  024724                    ADC    V2H
3321
3322                                    ; DIVIDE (REMAINDER*(2^16))+SQR0) BY V1(H/L)
3323                                    ;
3324   025630  004737  024744                    JSR    PC,DIVI         ;GET SECOND RESULT
3325
3326                                    ; ADD TEMPH TO RESULT IN VMULH AND CHECK FOR ARITHMETIC OVERFLOW
3327                                    ;
3328   025634  063737  02470.  024714           ADD    TEMPH,VMULH     ;ADD IN FIRST RESULT
3329   025642  100003                             BPL    2$              ;BRANCH IF NO OVERFLOW
3330
3331                                    ; ERROR ARITHMETIC OVERFLOW OCCURED DURING DIVISION OPERATION
3332                                    ; OF (REMAINDER*(2^16))+SQR0 BY V1(H/L).
3333                                    ;
3334   025644  004537  024416                    JSR    R5,TOOBIG       ;REPORT ERROR
3335   025650  046152                             ERDIV
3336   025652  000207                    2$:     RTS    PC
```

```
3338                                        ;ROUTINE TO SQUARE A 32 BIT NUMBER WITH 16 BITS AFTER POINT
3339                                        ;RETURNS A 48 BIT NUMBER WITH 16 BITS AFTER POINT
3340                                        ;
3341                                        ;CALLING SEQUENCE:
3342                                        ;       CALL    SQUARE
3343                                        ;
3344                                        ;SQUARES THE 32 BIT NUMBER CONTAINED IN VMUL(H/L) AND RETURNS THE RESULT IN
3345                                        ;SQR(2/1/0).  NUMBERS/LETTERS IN "()" ARE MOST TO LEAST SIGNIFICANT LEFT TO
3346                                        ;RIGHT.
3347                                        ;
3348                                        ;NOTE: THE ORIGINAL CONTENTS OF VMUL(H/L) WILL BE LOST.
3349                                        ;
3350                                        ; SAVE VMUL(H/L) ON STACK
3351                                        ;
3352   025654  013746  024714     SQUARE: MOV     VMULH,-(SP)     ;SAVE 32 BIT NUMBER ON STACK
3353   025660  013746  024712             MOV     VMULL,-(SP)
3354                                        ;
3355                                        ; SQUARE NUMBER IN VMULL
3356                                        ;
3357   025664  005037  024714             CLR     VMULH           ;SETUP FOR FIRST MULTIPLICATION
3358   025670  013737  024712  025702     MOV     VMULL,1$
3359   025676  004537  025224             JSR     R5,MULTI
3360   025702  000000             1$:     0
3361                                        ;
3362                                        ; ROUND RESULT TO 16 BITS OF FRACTIONAL PART
3363                                        ;
3364   025704  062700  100000             ADD     #BIT15,R0       ;ROUND OFF DECIMAL PART
3365   025710  005501                      ADC     R1
3366                                        ;
3367                                        ; SAVE PART OF FRACTION PART IN SQR0 AND CLEAR SQR1, AND SQR2.
3368                                        ;
3369   025712  010137  024726     2$:     MOV     R1,SQR0         ;SAVE RESULT
3370   025716  005037  024730             CLR     SQR1
3371   025722  005037  024732             CLR     SQR2
3372                                        ;
3373                                        ; MULTIPLY VMULL BY VMULH
3374                                        ;
3375   025726  012637  024712             MOV     (SP)+,VMULL     ;SETUP FOR SECOND MULTIPLICATION
3376   025732  005037  024714             CLR     VMULH
3377   025736  011637  025746             MOV     (SP),3$
3378   025742  004537  025224             JSR     R5,MULTI
3379   025746  000000             3$:     0
3380                                        ;
3381                                        ; MULTIPLY THE RESULT BY 2 - THIS IS QUICKER THAN DOING THE SAME MULTIPICATION
3382                                        ; TWICE.
3383                                        ;
3384   025750  006300                      ASL     R0              ;MULTIPLY RESULT BY 2
3385   025752  006101                      ROL     R1
3386                                        ;
3387                                        ; ADD RESULT TO SQR(2/1/0) WHERE SQR2 GET ANY CARRY THAT RESULTED.
3388                                        ;
3389   025754  060037  024726             ADD     R0,SQR0
3390   025760  005537  024730             ADC     SQR1
3391   025764  060137  024730             ADD     R1,SQR1         ;ADD TO PREVIOUS RESULT
3392   025770  005537  024732             ADC     SQR2
3393                                        ;
```

```
3394                                    ; CHECK FOR ARITHMETIC OVERFLOW
3395                                    ;
3396    025774  100003                          BPL     4$
3397                                    ;
3398                                    ; ERROR OCCURED WHILE ATTEMPTING TO SQUARE VMUL(H/L)
3399                                    ;
3400    025776  004537  024416                  JSR     R5,TOOBIG           ;REPORT ERROR
3401    026002  046337                          ERSQR
3402                                    ;
3403                                    ; MULTIPLY VMULH BY VMULH
3404                                    ;
3405    026004  011637  024712          4$:     MOV     (SP),VMULL          ;SETUP FOR LAST MULTIPLICATION
3406    026010  012637  026020                  MOV     (SP)+,5$
3407    026014  004537  025224                  JSR     R5,MULTI
3408    026020  000000                  5$:     0
3409                                    ;
3410                                    ; ADD RESULT TO SQR(2/1)
3411                                    ;
3412    026022  060037  024730                  ADD     R0,SQR1             ;ADD IN LAST FIGURE
3413    026026  005537  024732                  ADC     SQR2
3414    026032  060137  024732                  ADD     R1,SQR2
3415                                    ;
3416                                    ; CHECK FOR ARITHMETIC OVERFLOW
3417                                    ;
3418    026036  100003                          BPL     6$
3419                                    ;
3420                                    ; ARITHMETIC OVERFLOW OCCURED WHEN ADDING IN VMULH * VMULH TO SQR(2/1/0)
3421                                    ;
3422    026040  004537  024416                  JSR     R5,TOOBIG           ;REPORT ERROR
3423    026044  046337                          ERSQR
3424    026046  000207                  6$:     RTS     PC                  ;RETURN
3425
3426                                    ;SUBROUTINE TO PRINT THE VOLTAGE GAIN
3427    026050  062737  000510  024712  PRGAIN: ADD     #510,VMULL          ;ADD .005 LSB FOR ROUNDING REASONS
3428    026056  004737  026132                  JSR     PC,TYPDEC           ;TYPE OUT DECIMAL NUMBER
3429    026062  104401  026070                  TYPE    ,65$                ;;TYPE ASCIZ STRING
 (1)    026066  000401                          BR      64$                 ;;GET OVER THE ASCIZ
 (1)                                    ;;65$:   .ASCIZ  /./
 (1)    026072                          64$:
3430    026072  012737  000002  026130          MOV     #2,10$              ;SET UP # OF DECIMAL PLACES
3431    026100  004537  025224          1$:     JSR     R5,MULTI            ;MULTIPLY DECIMAL FRACTION BY 10(10)
3432    026104  000012                          10.
3433    026106  010037  024712                  MOV     R0,VMULL            ;SAVE DECIMAL PART
3434    026112  010100                          MOV     R1,R0               ;PUT NUMBER IN R0
3435    026114  004737  026276                  JSR     PC,TYPDIG           ;TYPE OUT DIGIT
3436    026120  005337  026130                  DEC     10$                 ;DECREMENT DIGIT COUNT
3437    026124  001365                          BNE     1$                  ;BRANCH IF NOT DONE
3438    026126  000207                          RTS     PC                  ;RETURN FROM PRGAIN
3439    026130  000000                  10$:    0
3440
3441                                    ;SUBROUTINE TO TYPE OUT A DECIMAL NUMBER
3442    026132  005737  024714          TYPDEC: TST     VMULH               ;TEST NUMBER
3443    026136  001005                          BNE     1$                  ;BRANCH IF NUMBER NOT ZERO
3444    026140  104401  026146                  TYPE    ,65$                ;;TYPE ASCIZ STRING
 (1)    026144  000401                          BR      64$                 ;;GET OVER THE ASCIZ
 (1)                                    ;;65$:   .ASCIZ  /0/
```

```
  (1)   026150                      64$:
 3445   026150   000207                      RTS     PC              ;RETURN FROM TYPDEC
 3446   026152   100015              1$:     BPL     DECPRT          ;BRANCH IF NUMBER POSITIVE
 3447   026154   104401   026162             TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)   026160   000401                      BR      66$             ;;GET OVER THE ASCIZ
  (1)                        ;;67$:  .ASCIZ  /-/
  (1)   026164                      66$:
 3448   026164   005137   024712             COM     VMULL           ;TWO'S COMPLEMENT NUMBER
 3449   026170   005137   024714             COM     VMULH
 3450   026174   062737   000001   024712    ADD     #1,VMULL
 3451   026202   005537   024714             ADC     VMULH
 3452   026206   005737   024714    DECPRT:  TST     VMULH           ;TEST NUMBER
 3453   026212   001001                      BNE     1$              ;BRANCH IF NUMBER NOT ZERO
 3454   026214   000207                      RTS     PC              ;RETURN
 3455   026216   010046              1$:     MOV     R0,-(SP)        ;SAVE WORK REGISTER
 3456   026220   012701   050000             MOV     #50000,R1       ;GET TEST NUMBER
 3457   026224   013700   024714             MOV     VMULH,R0        ;GET DIVIDEND
 3458   026230   005037   024714             CLR     VMULH           ;CLEAR RESULT
 3459   026234   006337   024714    2$:      ASL     VMULH           ;DIVIDE R0 BY 10
 3460   026240   020001                      CMP     R0,R1           ;RESULT IN VMULH
 3461   026242   100403                      BMI     3$              ;REMAINDER IN R0
 3462   026244   160100                      SUB     R1,R0
 3463   026246   005237   024714             INC     VMULH
 3464   026252   006201              3$:     ASR     R1
 3465   026254   022701   000005             CMP     #5,R1           ;TEST FOR DONE
 3466   026260   001365                      BNE     2$              ;BRANCH IF NOT DONE
 3467   026262   004737   026206             JSR     PC,DECPRT       ;DO DIVISION AGAIN TILL VMULH = 0
 3468   026266   004737   026276             JSR     PC,TYPDIG       ;TYPE OUT DIGIT
 3469   026272   012600                      MOV     (SP)+,R0        ;RESTORE WORK REGISTER
 3470   026274   000207                      RTS     PC              ;RETURN
 3471   026276   062700   000060    TYPDIG:  ADD     #60,R0          ;MAKE NUMBER ASCII
 3472   026302   110037   053612             MOVB    R0,ONES         ;SAVE FOR TYPEOUT
 3473   026306   104401   053612             TYPE    ,ONES           ;TYPE OUT NUMBER
 3474   026312   000207                      RTS     PC              ;RETURN FROM TYPDIG
```

```
3476
3477                              ;*SUB-ROUTINE TO ASK CHANNELS TO TEST
3478
3479   026314  104401  043651     ASKC:   TYPE    ,TCHAN           ;TYPE "TEST CHANNELS "
3480   026320  010046                      MOV     R0,-(SP)         ;:SAVE R0 FOR TYPEOUT
 (1)   026322  104403                      TYPOS                    ;:GO TYPE--OCTAL ASCII
 (1)   026324     002                      .BYTE   2                ;:TYPE 2 DIGIT(S)
 (1)   026325     000                      .BYTE   0                ;:SUPPRESS LEADING ZEROS
3481   026326  104401  042162             TYPE    ,MDASH           ;TYPE " - "
3482   026332  010146                      MOV     R1,-(SP)         ;:SAVE R1 FOR TYPEOUT
 (1)   026334  104403                      TYPOS                    ;:GO TYPE--OCTAL ASCII
 (1)   026336     002                      .BYTE   2                ;:TYPE 2 DIGIT(S)
 (1)   026337     000                      .BYTE   0                ;:SUPPRESS LEADING ZEROS
3483   026340  104401  042156             TYPE    ,QUEST           ;TYPE " ? "
3484   026344  104412                      RDLIN                    ;GET RESPONSE
3485   026346  012602                      MOV     (SP)+,R2         ;GET ADDRESS OF RESPONSE TEXT
3486   026350  142712  000040             BICB    #40,(R2)         ;MAKE CHARACTER UPPER CASE
3487   026354  122712  000131             CMPB    #'Y,(R2)         ;IS IT A Y?
3488   026360  001406                      BEQ     1$               ;:YES
3489   026362  122712  000116             CMPB    #'N,(R2)         ;IS IT AN N?
3490   026366  001405                      BEQ     2$               ;:YES
3491   026370  104401  043672             TYPE    ,YESNO           ;TYPE "TYPE Y FOR YES, N FOR NO"
3492   026374  000747                      BR      ASKC             ;:
3493   026376  062716  000002     1$:     ADD     #2,(SP)          ;SKIP OVER BRANCH
3494   026402  000207             2$:     RTS     PC               ;RETURN
3495
3496                              ;SUBROUTINE TO LOAD THE TYPE OF CHANNEL CODE INTO "CHTABL" BUFFER
3497   026404  112537  026442     LODTAB: MOVB    (R5)+,10$        ;GET CODE VALUE
3498   026410  112537  026444             MOVB    (R5)+,11$        ;GET NUMBER OF CHANNELS
3499   026414  113761  026442 060156 1$:  MOVB    10$,CHTABL(R1)   ;SAVE THIS CHANNELS TYPE
3500   026422  105337  026444             DECB    11$              ;MORE CHANNELS ?
3501   026426  001402                      BEQ     2$               ;BR IF DONE
3502   026430  005201                      INC     R1               ;UPDATE CHANNEL NUMBER
3503   026432  000770                      BR      1$               ;LOAD NEXT CHANNEL TYPE
3504   026434  000240             2$:     NOP
3505   026436  000240                      NOP
3506   026440  000205                      RTS     R5               ;EXIT
3507   026442  000000             10$:    0
3508   026444  000000             11$:    0
3509
3510                              ;SUBROUTINE TO SET THE "TEST THIS CHANNEL" BIT
3511   026446  152760  000200 060156 SETASK: BISB  #BIT7,CHTABL(R0)         ;SET THE BIT
3512   026454  020001                      CMP     R0,R1            ;FINISHED LOADING
3513   026456  001402                      BEQ     1$               ;BR IF DONE
3514   026460  005200                      INC     R0               ;UPDATE CHANNEL NUMBER
3515   026462  000771                      BR      SETASK           ;BR BACK
3516   026464  000207             1$:     RTS     PC               ;EXIT
```

```
3518                                    ;         TEST THE MNCTP USING THE MNCTP TEST MODULE
3519   026466  000240           AMNCTP: NOP
3520   026470  005037  001112           CLR      $ERTTL              ;PRIME ERROR COUNT
3521   026474  000240                   NOP
3522   026476  012737  040746  027216   MOV      #NOTSIE,72$         ;LOAD TEXT POINTER
3523   026504  013746  000004           MOV      @#ERRVEC,-(SP)      ;SAVE BUS TRAP POINTER VALUE
3524   026510  012737  027200  000004   MOV      #70$,@#ERRVEC       ;LOAD NEW RETURN POINTER
3525   026516  000240                   NOP
3526   026520  005777  001462           TST      @MNCTMO             ;ADDRESS THE MNCTP INCOMMING TESTER MODULE
3527   026524  000240                   NOP
3528   026526  000240                   NOP
3529   026530  000240                   NOP
3530   026532  004537  032720           JSR      R5,DVMIEE           ;LOAD DEVICE #2 <DVM> TO RESPOND
3531   026536  031310                   INIEE                        ;INITILIZE DEVICE ADDRESS #2
3532   026540  000000                   0
3533   026542  000240                   NOP
3534   026544  000240                   NOP
3535   026546  000240                   NOP
3536   026550  000240                   NOP
3537   026552  000240                   NOP
3538   026554  000240                   NOP
3539   026556  000240                   NOP
3540   026560  000240                   NOP
3541   026562  000240                   NOP
3542   026564  000240                   NOP
3543   026566  000240                   NOP
3544   026570  000240                   NOP
3545   026572  000240                   NOP
3546   026574  012637  000004           MOV      (SP)+,@#ERRVEC      ;RESTORE BUS TRAP POINTER
3547   026600  032777  040000  004420   BIT      #BIT14,@IBCSR       ;TEST IF IEEE DEVICE ADDRESS #2 IS PRESENT
3548   026606  001405                   BEQ      1$                  ;BR IF ERROR BIT IS CLEARED
3549   026610  012737  041023  027216   MOV      #NOLSTN,72$         ;LOAD ERROR MESSAGE TEXT POINTER
3550   026616  000137  027206           JMP      71$                 ;AND REPORT THAT DEVICE #2 ISN'T THERE
3551   026622  104401  041164   1$:     TYPE     ,SCHAN              ;ASK OPERATOR WHAT CHANNEL TO START WITH
3552   026626  104413                   RDOCT
3553   026630  012637  027224           MOV      (SP)+,TPSCHN        ;GET ANSWER
3554   026634  001003                   BNE      2$                  ;BR IF ANSWER WAS GIVEN
3555   026636  012737  000014  027224   MOV      #14,TPSCHN          ;USE CH14 AS THE DEFAULT
3556   026644  042737  177700  027224 2$: BIC    #177700,TPSCHN      ;ENSURE ONLY 2 DIGIT NUMBER
3557
3558   026652  005037  001112           CLR      $ERTTL              ;ENSURE CLEAR ERROR COUNT AFTER EACH RUN
3559
3560   026656  000240                   NOP
3561   026660  000240                   NOP
3562
3563   026662  004537  003002           JSR      R5,ASKTA            ;ASK OPERATOR IF HE WANTS TO SELECT GAIN TYPE
3564   026666  047274                   SELGT                        ;TEXT POINTER
3565   026670  001540                   TEMP
3566   026672  000471                   BR       60$                 ;BR IF ANSWER WAS NO
3567                               ;OPERATOR WANTS TO SELECT ONE
3568   026674  104401  047336           TYPE     ,SELGTA             ;TELL OPER THE CHOICES
3569   026700  104413                   RDOCT
3570   026702  012600                   MOV      (SP)+,R0            ;GET VALUE
3571   026704  042700  177774           BIC      #177774,R0          ;ENSURE 0-3 ANSWER
3572   026710  001013                   BNE      21$                 ;BR IF NOT CODE 0
3573
```

```
3574   026712   004537   027226            JSR     R5,MNCTM      ;TEST MNCTP CODE 0
3575   026716   030530                      GS00
3576   026720      000      001             .BYTE   0,1
3577   026722   100060                      .WORD   100060        ;
3578   026724   100063                      .WORD   100063        ;+- 03.5 MV
3579   026726   100065                      .WORD   100065        ;
3580   026730   000240                      NOP
3581   026732   000240                      NOP
3582   026734   000137   027166            JMP     3$
3583                              ;ONLY CODE 01
3584   026740   022700   000001   21$:      CMP     #1,R0         ;TEST IF CODE 1
3585   026744   001013                      BNE     22$           ;BR IF NOT
3586
3587   026746   004537   027226            JSR     R5,MNCTM      ;TEST MNCTP CODE 1
3588   026752   030570                      GS01
3589   026754      001      002             .BYTE   1,2
3590   026756   100060                      .WORD   100060
3591   026760   100066                      .WORD   100066        ;+- 06.5 MV
3592   026762   100065                      .WORD   100065        ;
3593   026764   000240                      NOP
3594   026766   000240                      NOP
3595   026770   000137   027166            JMP     3$
3596                              ;ONLY CODE 2
3597   026774   022700   000002   22$:      CMP     #2,R0         ;TEST IF CODE 2
3598   027000   001013                      BNE     23$           ;BR IF NOT
3599
3600   027002   004537   027226            JSR     R5,MNCTM      ;TEST MNCTP CODE 2
3601   027006   030630                      GS10
3602   027010      010      004             .BYTE   10,4
3603   027012   100061                      .WORD   100061
3604   027014   100060                      .WORD   100060        ;+- 10.0 MV
3605   027016   100060                      .WORD   100060
3606   027020   000240                      NOP
3607   027022   000240                      NOP
3608   027024   000137   027166            JMP     3$            ;END OF PASS
3609
3610                              ;ONLY CODE 3 IS LEFT
3611   027030   004537   027226   23$:      JSR     R5,MNCTM      ;TEST MNCTP CODE 3
3612   027034   030670                      GS11
3613   027036      011      010             .BYTE   11,10
3614   027040   100061                      .WORD   100061
3615   027042   100063                      .WORD   100063        ;+- 13.5 MV
3616   027044   100065                      .WORD   100065
3617   027046   000240                      NOP
3618   027050   000240                      NOP
3619   027052   000137   027166            JMP     3$            ;END OF PASS
3620
3621                              ;ROUTINE TO CYCLE THRU ALL 4 GAIN/TYPES
3622
3623   027056   004537   027226   60$:      JSR     R5,MNCTM      ;TEST MNCTP USING GAIN SELECT 0000
3624   027062   030530                      GS00
3625   027064      000      001             .BYTE   0,1
3626   027066   100060                      .WORD   100060
3627   027070   100063                      .WORD   100063        ;+- 03.5 MV TOLERANCE
3628   027072   100065                      .WORD   100065
3629   027074   000240                      NOP
```

```
3630   027076  00024C   .                    NOP
3631
3632   027100  004537  027226               JSR      R5,MNCTM        ;TEST MNCTP USING GAIN SELECT 0001
3633   027104  030570                        GS01
3634   027106    001     002                 .BYTE    1,2
3635   027110  100060                        .WORD    100060
3636   027112  100066                        .WORD    100066          ;+- 06.5 MV TOLERANCE
3637   027114  100065                        .WORD    100065
3638   027116  000240                        NOP
3639   027120  000240                        NOP
3640
3641   027122  004537  027226               JSR      R5,MNCTM        ;TEST MNCTP USING GAIN SELECT 1000
3642   027126  030630                        GS10
3643   027130    010     004                 .BYTE    10,4
3644   027132  100061                        .WORD    100061
3645   027134  100060                        .WORD    100060          ;+- 10.0 MV TOLERANCE
3646   027136  100060                        .WORD    100060
3647   027140  000240                        NOP
3648   027142  000240                        NOP
3649
3650   027144  004537  027226               JSR      R5,MNCTM        ;TEST MNCTP USING GAIN SELECT 1001
3651   027150  030670                        GS11
3652   027152    011     010                 .BYTE    11,10
3653   027154  100061                        .WORD    100061
3654   027156  100063                        .WORD    100063          ;+- 13.5 MV TOLERANCE
3655   027160  100065                        .WORD    100065
3656   027162  000240                        NOP
3657   027164  000240                        NOP
3658
3659   027166  012737  026622  040352  3$:   MOV      #1$,AGTST                 ;RETURN POINTER AFTER EOP
3660   027174  000137  040154               JMP      $EOP            ;REPORT AND OF PASS AND ERROR COUNT
3661
3662                              ;ROUTINE TO HANDLE BUS TIME-OUT REFERENCE TO THE MNCTP TESTER OR IEEE INTER.
3663   027200  022626           70$:   CMP      (SP)+,(SP)+     ;CLEAN OFF STACK
3664   027202  012637  000004          MOV      (SP)+,@#ERRVEC  ;RESTORE BUS TRAP POINTER
3665   027206  042737  040000  001602  71$:   BIC      #BIT14,WFTEST   ;REMOVE MNCTP TESTER FLAG
3666   027214  104401                   TYPE                     ;TELL OPERATOR THE BAD NEWS
3667   027216  040746           72$:   NOTSIE                   ;NO TEST MODULE/IEEE OR DEVICE #2 NOT PRESENT
3668   027220  000137  002634          JMP      MTESTA          ;RETURN TO MAIN PROGRAM SECTION
3669   027224  000000           TPSCHN: 0
3670
3671                              ;SUBROUTINE FOR THE MNCTP TESTER HANDLER
3672                              ;       JSR      R5,MNCTM        ;CALL IS FOLLOWED BY:
3673                              ;       GS00             ;GAIN SELECT TEXT POINTER
3674                              ;       .BYTE    X       ;MNCTP GAIN CODE
3675                              ;       .BYTE    Y       ;MNCTM GAIN CODE
3676                              ;       .WORD    100060
3677                              ;       .WORD    100063  ;+- 03.5 MV TOLERANCE
3678                              ;       .WORD    100065  ;
3679
3680   027226  012537  027370    MNCTM: MOV      (R5)+,70$       ;GET ASCII TEXT POINTER
3681   027232  005077  000750           CLR      @MNCTM0         ;ENSURE NOT IN MNCTP TEST MODE
3682                              ;LOAD MNCTP CHANNELS TO SELECTED GAIN CODE VALUE
3683   027236  112500                   MOVB     (R5)+,R0        ;GET MNCTP GAIN CODE
3684   027240  110037  030212           MOVB     R0,TPGVAL       ;SAVE FOR LATER USE
3685   027244  013701  027224           MOV      TPSCHN,R1       ;START WITH SELECTED MNCTP CHANNEL
```

```
3686  027250  013702  001460                 MOV    ADST1,R2          ;GET MNCAD BUS ADDRESS
3687  027254  012703  000010                 MOV    #10,R3            ;LOAD LOOP COUNTER
3688  027260  112712  000077         1$:     MOVB   #77,(R2)          ;LOAD ''ESCAPE''
3689  027264  110012                         MOVB   R0,(R2)           ;LOAD SELECTED GAIN
3690  027266  110112                         MOVB   R1,(R2)           ;LOAD SELECTED CHANNEL
3691  027270  005201                         INC    R1                ;UPDATE CHANNEL VALUE
3692  027272  005303                         DEC    R3                ;FINISHED ALL CHANNELS ?
3693  027274  001371                         BNE    1$                ;BR IF MORE
3694  027276  012777  000020  000702         MOV    #20,@MNCTM0       ;SET ''TEST MODE'' TESTER BIT
3695  027304  152577  000676                 BISB   (R5)+,@MNCTM0     ;SET TESTER GAIN VALUE
3696  027310  012700  000000                 MOV    #0,R0             ;INIT THE SCALE BEING USED
3697  027314  011537  032424                 MOV    (R5),LTOL2        ;LOAD LOW LIMIT HSD TOLERANCE VALUE
3698  027320  012537  032542                 MOV    (R5)+,HTOL2       ;LOAD HIGH LIMIT HSD TOLERANCE VALUE
3699  027324  011537  032426                 MOV    (R5),LTOL0        ;LOAD LOW LIMIT MSD TOLERANCE VALUE
3700  027330  012537  032544                 MOV    (R5)+,HTOL0       ;LOAD HIGH LIMIT MSD TOLERANCE VALUE
3701  027334  011537  032430                 MOV    (R5),LTOL1        ;LOAD LOW LIMIT LSD TOLERANCE VALUE
3702  027340  012537  032546                 MOV    (R5)+,HTOL1       ;LOAD HIGH LIMIT LSD TOLERANCE VALUE
3703  027344  113737  032424  030516         MOVB   LTOL2,GSTOL2      ;LOAD TYPE-OUT TOLERANCE VALUE
3704  027352  113737  032426  030517         MOVB   LTOL0,GSTOL0      ;           ''         ''        ''
3705  027360  113737  032430  030521         MOVB   LTOL1,GSTOL1      ;           ''         ''        ''
3706  027366  104401                         TYPE
3707  027370  000000         70$:            0                       ;TELL OPERATOR TP TYPE
3708  027372  104401  030507                 TYPE   ,GSTOL            ;TELL OPERATOR TOLERANCE VALUES
3709
3710  027376  016001  030022         2$:     MOV    SCLDAC(R0),R1     ;GET TESTER DAC VALUE
3711  027402  016037  027776  030166         MOV    SCLTXT(R0),SCLMSG       ;GET THE SCALE TEXT POINTER
3712  027410  032737  000010  030212         BIT    #10,TPGVAL        ;TEST IF GAIN TYPE 10 OR 11
3713  027416  001407                         BEQ    5$                ;BR IF GAIN TYPE 00 OR 01
3714  027420  016001  030034                 MOV    SCLCAD(R0),R1     ;GET DAC DATA VALUE
3715  027424  016037  030010  030166         MOV    SCLXTX(R0),SCLMSG       ;GET DAC TEXT POINTER
3716  027432  000240                         NOP
3717  027434  000240                         NOP
3718  027436  000240         5$:             NOP
3719  027440  000240                         NOP
3720  027442  000240                         NOP
3721  027444  000240                         NOP
3722  027446  110177  000536                 MOVB   R1,@MNCTM1        ;LOAD TESTER DAC VALUE
3723
3724  027452  142777  000340  000526         BICB   #340,@MNCTM0      ;CLEAR 'READING' BITS
3725  027460  152777  000040  000520         BISB   #40,@MNCTM0       ;SELECT 'READING TEMP' BITS
3726
3727  027466  004537  032720                 JSR    R5,DVMIEE         ;READ DVM - FOR THE TEMP AND SAVE VALUE READ
3728  027472  031372                         RDTEMP                   ;IEEE COMMAND POINTER
3729  027474  031173                         DVMTMP                   ;POINTER TO ANSWER SPACE
3730
3731  027476  142777  000340  000502         BICB   #340,@MNCTM0      ;CLEAR 'READING' BITS
3732  027504  152777  000200  000474         BISB   #200,@MNCTM0      ;SELECT 'READING V-IN' BITS
3733
3734  027512  004537  032720                 JSR    R5,DVMIEE         ;READ DVM - FOR THE V-IN AND SAVE VALUE READ
3735  027516  031540                         RDVIN                    ;IEEE COMMAND POINTER
3736  027520  031121                         DVMVI                    ;POINTER TO ANSWER SPACE
3737
3738                                  ;NOW LOAD THE + OFFSET VALUE
3739  027522  004537  032720                 JSR    R5,DVMIEE         ;LOAD + OFFSET TOLERANCE INTO DVM ''Z'' REG.
3740  027526  032410                         RDVINP                   ;IEEE COMMAND POINTER FOR PLUS
3741  027530  000000                         0                        ;NO ANSWER EXPECTED
```

```
3742
3743  027532  004537  032720              JSR    R5,DVMIEE    ;LOAD ADJUSTED VALUE INTO LOWER LIMIT REG.
3744  027536  032520                       RDVINL             ;IEEE COMMAND POINTER FOR LOWER + NEG
3745  027540  000000                       0                  ;NO ANSWER EXPECTED
3746
3747  027542  004537  032720              JSR    R5,DVMIEE    ;LOAD ADJUSTED VALUE INTO UPPER LIMIT REG.
3748  027546  032636                       RDVINU             ;IEEE COMMAND POINTER FOR UPPER
3749  027550  000000                       0                  ;NO ANSWER EXPECTED
3750
3751                                  ;THE UPPER AND LOWER LIMIT REGISTERS HAVE NOW BEEN LOADED
3752                                  ; ''V-IN'' - THE OFFSET IN LOWER
3753                                  ; ''V-IN'' + THE OFFSET IN UPPER
3754
3755  027552  142777  000340  000426    BICB   #340,@MNCTM0   ;CLEAR ''READING'' BITS
3756  027560  152777  000100  000420    BISB   #100,@MNCTM0   ;SELECT ''READING V-OUT'' BITS
3757  027566  005001                    CLR    R1             ;INIT THE CHANNEL INDEX
3758
3759  027570  013702  027224    3$:     MOV    TPSCHN,R2      ;GET 1ST MNCTP CHANNEL
3760  027574  060102                    ADD    R1,R2          ;ADD CURRENT CHANNEL INDEX VALUE
3761  027576  110277  151656            MOVB   R2,@ADST1      ;SELECT THE MNCTP CHANNEL
3762  027602  012737  001165  030202    MOV    #$CRLF,DECRPC  ;PRIME NO ERROR MESSAGE
3763  027610  105277  151642            INCB   @STREG         ;START AN CONVERSION
3764  027614  110102                    MOVB   R1,R2          ;COPY CHANNEL POINTER
3765  027616  062702  000060            ADD    #60,R2         ;MAKE IT ASCII VALUE
3766  027622  110237  030223            MOVB   R2,TMCH        ;SAVE FOR TYPE-OUT
3767
3768  027626  052777  000400  000352    BIS    #400,@MNCTM0   ;SET (AZT-L)
3769  027634  004537  032720            JSR    R5,DVMIEE      ;TELL DVM TO DO 1ST SAMPLE
3770  027640  031642                    RDVOUA               ;IEEE COMMAND POINTER
3771  027642  000000                    0                    ; ''NO ANSWER EXPECTED''
3772
3773  027644  000240                    NOP
3774  027646  000240                    NOP
3775  027650  004537  032720            JSR    R5,DVMIEE      ;TELL DVM TO STORE SAMPLE IN ''Z'' REGISTER
3776  027654  031766                    RDVOUB               ;IEEE COMMAND POINTER
3777  027656  000000                    0                    ;''NO ANSWER EXPECTED''
3778
3779  027660  000240                    NOP
3780  027662  000240                    NOP
3781  027664  042777  000400  000314    BIC    #400,@MNCTM0   ;REMOVE (AZT-L)
3782  027672  000240                    NOP
3783  027674  000240                    NOP
3784  027676  004537  032720            JSR    R5,DVMIEE      ;TELL DVM TO SCALE RESULT AND SEND RESULT TO CPU
3785  027702  032104                    RDVOUC               ;IEEE COMMAND POINTER
3786  027704  031047                    DVMVO                ;POINTER TO ANSWER SPACE
3787
3788  027706  000240                    NOP
3789  027710  000240                    NOP
3790  027712  000240                    NOP
3791  027714  000240                    NOP
3792
3793  027716  004537  032720            JSR    R5,DVMIEE      ;TELL DVM TO INDICATE PASS/FAIL
3794  027722  032176                    RDVOUD               ;IEEE COMMAND POINTER
3795  027724  000000                    0                    ;NO ANSWER EXPECTED
3796
3797  027726  000240                    NOP
```

```
3798   027730   000240                          NOP
3799   027732   000240                          NOP
3800   027734   000240                          NOP
3801
3802   027736   105701                          TSTB    R1                  ;TEST IF FIRST CHANNEL
3803   027740   001002                          BNE     4$                  ;BR IF NOT
3804   027742   104401   030331                 TYPE    ,TMHEAD             ;REPORT THE OUTPUT COLUMN HEADER
3805
3806   027746   004737   030046         4$:     JSR     PC,DECRPT           ;REPORT READINGS
3807
3808   027752   005201                          INC     R1                  ;UPDATE CHANNEL BEING USED
3809   027754   022701   000010                 CMP     #10,R1              ;TEST IF LAST MNCTP CHANNEL
3810   027760   001303                          BNE     3$                  ;BR IF MORE CHANNELS AT THIS SCALE
3811
3812   027762   062700   000002                 ADD     #2,R0               ;UPDATE SCALE BEING USED
3813   027766   022700   000012                 CMP     #12,R0              ;TEST IF COMPLETED LAST SCALE
3814   027772   001201                          BNE     2$                  ;BR IF MORE SCALES TO DO
3815
3816   027774   000205                          RTS     R5                  ;EXIT
3817
3818   027776   030730   030740   030751 SCLTXT: SCAL0,SCAL1,SCAL2,SCAL3,SCAL4    ;SCALE TEXT POINTERS FOR + SIDE
       030004   030762   030773
3819   030010   031037   031026   031015 SCLXTX: SCAL8,SCAL7,SCAL6,SCAL5,SCAL4    ;SCALE TEXT POINTERS FOR - SIDE
       030016   031004   030773
3820
3821   030022   000376   000340   000300 SCLDAC: .WORD   376, 340, 300, 240, 200 ;TESTER DAC VALUE FOR + SIDE
       030030   000240   000200
3822   030034   000000   000040   000100 SCLCAD: .WORD   0,40,100,140,200        ;TESTER DAC VALUE FOR - SIDE
       030042   000140   000200
3823
3824                                     ;SUBROUTINE TO DECODE THE GAIN SELECTED AND
3825                                     ;CH0     REPORT THE GAIN, CHANNEL INDEX, V-OUT, V-IN, SCALE, AND TEMP
3826                                     ;CH1-7   REPORT THE GAIN, CHANNEL INDEX, V-OUT
3827   030046   112737   000060   030220 DECRPT: MOVB    #60,TMG1            ;PRIME THE GAIN OUTPUT TEXT
3828   030054   112737   000060   030221         MOVB    #60,TMG0            ;   FOR REPORTING
3829   030062   123727   030212   000000         CMPB    TPGVAL,#0           ;TEST IF CODE 0000
3830   030070   001417                          BEQ     3$                  ;BR IF CORRECT
3831   030072   123727   030212   000001         CMPB    TPGVAL,#1           ;TEST IF CODE 0001
3832   030100   001411                          BEQ     2$                  ;BR IF CORRECT
3833   030102   123727   030212   000010         CMPB    TPGVAL,#10          ;TEST IF CODE 1000
3834   030110   001003                          BNE     1$
3835   030112   105237   030220                 INCB    TMG1                ;ADJUST TO CODE 1000
3836   030116   000404                          BR      3$
3837   030120   105237   030220         1$:     INCB    TMG1                ;UPDATE GAIN TEXT
3838   030124   105237   030221         2$:     INCB    TMG0                ;  OUTPUT REPORT
3839   030130   104401   030220         3$:     TYPE    ,TMG1               ;TELL OPERATOR THE CHANNEL AND GAIN SELECTED
3840   030134   104401   046502                 TYPE    ,SPACE
3841   030140   104401   031047                 TYPE    ,DVMVO              ;TELL OPERATOR THE V-OUT VALUE
3842   030144   105701                          TSTB    R1                  ;TEST IF FIRST MNCTP CHANNEL
3843   030146   001014                          BNE     DECRPB              ;BR IF CHAN 1-7
3844   030150   104401   046502                 TYPE    ,SPACE
3845   030154   104401   031121                 TYPE    ,DVMVI              ;TELL OPERATOR THE V-IN VALUE
3846   030160   104401   046502                 TYPE    ,SPACE
3847   030164   104401                          TYPE
3848   030166   000000                  SCLMSG: 0                           ;TELL OPERATOR THE SCALE BEING USED
3849   030170   104401   046502                 TYPE    ,SPACE
```

```
3850  030174  104401   031173                    TYPE    ,DVMTMP           ;TELL OPERATOR THE TEMP SENSED
3851  030200  104401                     DECRPB:  TYPE                      ;GIVE A FRESH OUTPUT LINE
3852  030202  001165                     DECRPC:  $CRLF            ;OR POINTER TO ERROR TEXT
3853  030204  000207                              RTS     PC                ;EXIT TO THE CALLING ROUTINE
3854
3855  030206  171010                     MNCTM0:  171010                    ;MNCTP TESTER ADDRESS
3856  030210  171011                     MNCTM1:  171011
3857
3858  030212  000000                     TPGVAL:  0                         ;TEMP LOC OF MNCTP GAIN VALUE
3859  030214  000000                     TMFLAG:  0                         ;TESTER USER FLAG
3860  030216     015     012             TMOUT:   .BYTE   15,12             ;''CR-LF''
3861  030220     060                     TMG1:    .BYTE   60
3862  030221     060                     TMG0:    .BYTE   60
3863  030222     056                              .BYTE   56                ;''*''
3864  030223     060     000             TMCH:    .BYTE   60,0              ;CHANNEL CODE ''0-7''
3865  030225     200  051511   052040    DWTSTP:  .ASCIZ  <200>\IS THE MNCTP INCOMMING INSPECTION TEST MODULE' AND DVM CONNECTED ?
      030232  042510  046440   041516                                                                        :
      030240  050124  044440   041516
      030246  046517  044515   043516
      030254  044440  051516   042520
      030262  052103  047511   020116
      030270  042524  052123   046440
      030276  042117  046125   020105
      030304  047101  020104   053104
      030312  020115  047503   047116
      030320  041505  042524   020104
      030326  020077     000
3866  030331     015     012             TMHEAD:  .BYTE   15,12
3867  030333     107  027116   044103             .ASCII  \GN.CH    V-OUT       V-IN       SCALE      TEMP\
      030340  020040  020040   053040
      030346  047455  052125   020040
      030354  020040  020040   053040
      030362  044455  020116   020040
      030370  020040  020040   041523
      030376  046101  020105   020040
      030404  020040  020040   042524
      030412  050115
3868  030414     015     012                      .BYTE   15,12
3869  030416  026455  026455   026455             .ASCII  \-----------------------------------------------\
      030424  026455  026455   026455
      030432  026455  026455   026455
      030440  026455  026455   026455
      030446  026455  026455   026455
      030454  026455  026455   026455
      030462  026455  026455   026455
      030470  026455  026455   026455
      030476  026455  026455   026455
3870  030504     015     012      000             .BYTE   15,12,0
3871  030507     050  047524   020114    GSTOL:   .ASCII  \(TOL = \
      030514  020075
3872  030516     060                     GSTOL2:  .BYTE   60
3873  030517     060                     GSTOL0:  .BYTE   60
3874  030520     056                              .BYTE   56
3875  030521     060                     GSTOL1:  .BYTE   60
3876  030522  046440  024526   000200             .ASCIZ  \ MV)\<200>
3877  030530  052600  044523   043516    GS00:    .ASCIZ  <200>\USING GAIN SELECT CODE = 0000 \
```

```
        030536  043440  044501  020116
        030544  042523  042514  052103
        030552  041440  042117  020105
        030560  020075  030060  030060
        030566  000040
3878    030570  052600  044523  043516  GS01:   .ASCIZ  <200>\USING GAIN SELECT CODE = 0001 \
        030576  043440  044501  020116
        030604  042523  042514  052103
        030612  041440  042117  020105
        030620  020075  030060  030460
        030626  000040
3879    030630  052600  044523  043516  GS10:   .ASCIZ  <200>\USING GAIN SELECT CODE = 1000 \
        030636  043440  044501  020116
        030644  042523  042514  052103
        030652  041440  042117  020105
        030660  020075  030061  030060
        030666  000040
3880    030670  052600  044523  043516  GS11:   .ASCIZ  <200>\USING GAIN SELECT CODE = 1001 \
        030676  043440  044501  020116
        030704  042523  042514  052103
        030712  041440  042117  020105
        030720  020075  030061  030460
        030726  000040
3881    030730  025440  043040  051456  SCAL0:  .ASCIZ  \ + F.S.\
        030736  000056
3882    030740  020053  027463  020064  SCAL1:  .ASCIZ  \+ 3/4 FS\
        030746  051506     000
3883    030751     053  030440  031057  SCAL2:  .ASCIZ  \+ 1/2 FS\
        030756  043040  000123
3884    030762  020053  027461  020064  SCAL3:  .ASCIZ  \+ 1/4 FS\
        030770  051506     000
3885    030773     040  020040  020060  SCAL4:  .ASCIZ  \   0  FS\
        031000  043040  000123
3886    031004  020055  027461  020064  SCAL5:  .ASCIZ  \- 1/4 FS\
        031012  051506     000
3887    031015     055  030440  031057  SCAL6:  .ASCIZ  \- 1/2 FS\
        031022  043040  000123
3888    031026  020055  027463  020064  SCAL7:  .ASCIZ  \- 3/4 FS\
        031034  051506     000
3889    031037     040  020055  027106  SCAL8:  .ASCIZ  \ - F.S.\
        031044  027123     000
3890    031047     053  027065  031061  DVMV0:  .ASCIZ  \+5.123456\
        031054  032063  033065     000
3891    031061  000020                          .BLKW   20
3892    031121     053  027065  032466  DVMV1:  .ASCIZ  \+5.654321\
        031126  031464  030462     000
3893    031133  000020                          .BLKW   20
3894    031173     053  034471  034456  DVMTMP: .ASCIZ  \+99.99C\
        031200  041471     000
3895    031203  000020                          .BLKW   20
3896    031243  000020                          .BLKW   20
3897            031304                          .EVEN
3898    031304  000000                          .WORD   0
3899    031306  000000                          .WORD   0
3900
3901                                    ;LIST OF COMMANDS AND DATA TO INITILIZE DEVICE #2
```

```
3902  031310  000110    INIEE:  .WORD   110       ;COMMAND - CLEAR BUS
3903  031312  100077            .WORD   100077    ;DATA - UNIVERSAL UNLISTEN
3904  031314  100042            .WORD   100042    ;DATA - DEVICE 2 LISTEN <DVM>
3905  031316  000144            .WORD   144       ;COMMAND - TALKER ON
3906  031320  100123            .WORD   100123    ;DATA - 'S'
3907  031322  100117            .WORD   100117    ;DATA - 'O'
3908  031324  100061            .WORD   100061    ;DATA - '1' (SYSTEM OUTPUT MODE #1)
3909  031326  100000            .WORD   100000    ;DATA - 'NOP'
3910  031330  000000            .WORD   0         ;TERMINATOR
3911  031332  000020            .BLKW   20
3912
3913                    ;LIST OF COMMANDS AND DATA TO TAKE THE MNCTP TEMP.
3914  031372  100077    RDTEMP: .WORD   100077    ;DATA - UNIVERSAL UNLISTEN
3915  031374  100042            .WORD   100042    ;DATA - DEVICE 2 LISTEN <DVM>
3916  031376  000144            .WORD   144       ;COMMAND - TALKER ON
3917  031400  100122            .WORD   100122    ;DATA - 'R'
3918  031402  100123            .WORD   100123    ;DATA - 'S'
3919  031404  100060            .WORD   100060    ;DATA - '0' (READING STORE OFF)
3920  031406  100122            .WORD   100122    ;DATA - 'R'
3921  031410  100064            .WORD   100064    ;DATA - '4' (10 VOLT RANGE)
3922  031412  100115            .WORD   100115    ;DATA - 'M'
3923  031414  100060            .WORD   100060    ;DATA - '0' (MATH OFF)
3924  031416  100106            .WORD   100106    ;DATA - 'F'
3925  031420  100065            .WORD   100065    ;DATA - '5' (4 WIRE)
3926  031422  100115            .WORD   100115    ;DATA - 'M'
3927  031424  100066            .WORD   100066    ;DATA - '6' (MATH C TEMP)
3928  031426  100124            .WORD   100124    ;DATA - 'T'
3929  031430  100061            .WORD   100061    ;DATA - '1' (INTERNAL TRIGGER)
3930  031432  000105            .WORD   105       ;COMMAND - TAKE CONTROL + REM
3931  031434  100102            .WORD   100102    ;DATA - DEVICE #2 TALK ADDRESS
3932  031436  000000            .WORD   0         ;MESSAGE TERMINATOR
3933  031440  000020            .BLKW   20
3934  031500  000020            .BLKW   20
3935
3936                    ;LIST OF COMMANDS AND DATA TO READ THE MNCTP 'V-IN'
3937  031540  100077    RDVIN:  .WORD   100077    ;DATA - UNIVERSAL UNLISTEN
3938  031542  100042            .WORD   100042    ;DATA - DEVICE #2 LISTEN
3939  031544  000144            .WORD   144       ;COMMAND - TALKER ON
3940  031546  100122            .WORD   100122    ;DATA - 'R'
3941  031550  100123            .WORD   100123    ;DATA - 'S'
3942  031552  100060            .WORD   100060    ;DATA - '0' (READING STORE OFF)
3943  031554  100122            .WORD   100122    ;DATA - 'R'
3944  031556  100064            .WORD   100064    ;DATA - '4' (10 VOLT RANGE)
3945  031560  100106            .WORD   100106    ;DATA - 'F'
3946  031562  100061            .WORD   100061    ;DATA - '1' (DC VOLTAGE)
3947  031564  100115            .WORD   100115    ;DATA - 'M'
3948  031566  100060            .WORD   100060    ;DATA '0' (MATH OFF)
3949  031570  100124            .WORD   100124    ;DATA 'T'
3950  031572  100061            .WORD   100061    ;DATA '1' (INTERNAL TRIGGER)
3951  031574  000105            .WORD   105       ;COMMAND - TAKE CONTROL + REM
3952  031576  100102            .WORD   100102    ;DATA - DEVICE #2 TALK ADDRESS
3953  031600  000000            .WORD   0         ;MESSAGE TERMINATOR
3954  031602  000020            .BLKW   20
3955
3956                    ;LIST OF COMMANDS AND DATA TO PREPARE TO TAKE 1 PHASE OF 'V-OUT'
3957  031642  100077    RDVOUA: .WORD   100077    ;DATA - UNIVERSAL UNLISTEN
```

```
3958  031644  100042              .WORD     100042          ;DATA - LISNER #2
3959  031646  000144              .WORD     144             ;COMMAND - TALKER ON
3960  031650  100115              .WORD     100115          ;DATA - 'M'
3961  031652  100060              .WORD     100060          ;DATA '0' (MATH OFF)
3962  031654  100106              .WORD     100106          ;DATA - 'F'
3963  031656  100061              .WORD     100061          ;DATA - '1' (DC VOLTAGE)
3964  031660  100124              .WORD     100124          ;DATA 'T'
3965  031662  100061              .WORD     100061          ;DATA '1' (INTERNAL TRIGGER)
3966  031664  000000              .WORD     0               ;MESSAGE TERMINATOR
3967  031666  000020              .BLKW     20
3968  031726  000020              .BLKW     20
3969                          ;LIST OF COMMANDS AND DATA TO TAKE THE 2ND PHASE OF 'V-OUT'
3970  031766  100077      RDVOUB: .WORD     100077          ;DATA - UNIVERSAL UNLISTEN
3971  031770  100042              .WORD     100042          ;DATA - LISTNER #2
3972  031772  000144              .WORD     144             ;COMMAND - TALKER ON
3973  031774  100123              .WORD     100123          ;DATA 'S'
3974  031776  100124              .WORD     100124          ;DATA 'T'
3975  032000  100132              .WORD     100132          ;DATA - 'Z' (STORE DATA IN REG. Z)
3976  032002  000000              .WORD     0               ;MESSAGE TERMINATOR
3977  032004  000020              .BLKW     20
3978  032044  000020              .BLKW     20
3979                          ;LIST OF COMMANDS AND DATA TO TAKE THE 3RD PHASE OF 'V-OUT'
3980  032104  100077      RDVOUC: .WORD     100077          ;DATA - UNIVERSAL UNLISTEN
3981  032106  100042              .WORD     100042          ;DATA - LISTNER #2
3982  032110  000144              .WORD     144             ;DATA - TALKER ON
3983  032112  100115              .WORD     100115          ;DATA 'M'
3984  032114  100067              .WORD     100067          ;DATA '7' (SCALE RESULT)
3985  032116  100122              .WORD     100122          ;DATA - 'R'
3986  032120  100123              .WORD     100123          ;DATA - 'S'
3987  032122  100061              .WORD     100061          ;DATA - '1' (ENABLE READING STORE)
3988  032124  100124              .WORD     100124          ;DATA - 'T'
3989  032126  100063              .WORD     100063          ;DATA '3' (SINGLE TRIGGER)
3990  032130  000105              .WORD     105             ;COMMAND - TAKE CONTROL
3991  032132  100102              .WORD     100102          ;DATA - TALKER #2
3992  032134  000000              .WORD     0               ;TERMINATOR
3993  032136  000020              .BLKW     20
3994
3995                          ;LIST OF COMMANDS AND DATA TO READ 'V-OUT' PASS/FAIL STATUS
3996  032176  100077      RDVOUD: .WORD     100077          ;DATA - UNIVERSAL UNLISTEN
3997  032200  100042              .WORD     100042          ;DATA - LISTNER #2
3998  032202  000144              .WORD     144             ;COMMAND - TALK ON
3999  032204  100061              .WORD     100061          ;DATA - '1'
4000  032206  100123              .WORD     100123          ;DATA - 'S'
4001  032210  100124              .WORD     100124          ;DATA - 'T'
4002  032212  100122              .WORD     100122          ;DATA - 'R' (STORE 1 IN 'R' REGISTER)
4003  032214  100123              .WORD     100123          ;DATA - 'S'
4004  032216  100115              .WORD     100115          ;DATA - 'M'
4005  032220  100062              .WORD     100062          ;DATA - '2'
4006  032222  100060              .WORD     100060          ;DATA - '0'
4007  032224  100060              .WORD     100060          ;DATA - '0'
4008  032226  100115              .WORD     100115          ;DATA - 'M'
4009  032230  100061              .WORD     100061          ;DATA - '1' (PASS/FAIL)
4010  032232  100122              .WORD     100122          ;DATA 'R'
4011  032234  100105              .WORD     100105          ;DATA 'E'
4012  032236  100122              .WORD     100122          ;DATA 'R' (RECALL R REGISTER)
4013                          ;THE 'SRQ' REQUEST LINE WILL SET IF 'FAIL' CONDITION
```

```
4014   032240   100122                        .WORD    100122             ;DATA - 'R'
4015   032242   100123                        .WORD    100123             ;DATA - 'S'
4016   032244   100060                        .WORD    100060             ;DATA - '0' (READING STORE OFF)
4017   032246   000000                        .WORD    0                  ;TERMINATOR
4018   032250   000020                        .BLKW    20
4019                             ;LIST OF COMMANDS AND DATA TO HANDLE THE ''SRQ'' REQUEST
4020   032310   100077           RDVOUE: .WORD    100077             ;DATA - UNIVERSAL UNLISTEN
4021   032312   100042                        .WORD    100042             ;DATA - LISTEN #2
4022   032314   100177                        .WORD    100177             ;DATA - UNIVERSAL UNTALK
4023   032316   100102                        .WORD    100102             ;DATA - TALKER #2
4024   032320   100030                        .WORD    100030             ;DATA - SERIAL POLE REQUEST
4025   032322   000140                        .WORD    140                ;COMMAND - TALK
4026   032324   000105                        .WORD    105                ;COMMAND - TAKE CONTROL
4027   032326   100031                        .WORD    100031             ;DATA - SERIAL POLE DISABLE
4028   032330   100042                        .WORD    100042             ;DATA - LISTEN #2
4029   032332   100077                        .WORD    100077             ;DATA - UNIVERSAL UNLISTEN
4030   032334   100042                        .WORD    100042             ;DATA - LISTEN #2
4031   032336   000144                        .WORD    144                ;COMMAND - TALK
4032   032340   100122                        .WORD    100122             ;DATA - 'R'
4033   032342   100123                        .WORD    100123             ;DATA - 'S'
4034   032344   100060                        .WORD    100060             ;DATA - '0' (READING STORE OFF)
4035   032346   000000                        .WORD    0                  ;TEMINATOR
4036   032350   000020                        .BLKW    20
4037
4038                             ;LIST OF COMMANDS AND DATA TO PRIME THE LOWER LIMIT REGISTER
4039   032410   100077           RDVINP: .WORD    100077             ;DATA - UNIVERSAL UNLISTEN
4040   032412   100042                        .WORD    100042             ;DATA - LISTNER #2
4041   032414   000144                        .WORD    144                ;COMMAND - TALK ON
4042   032416   100115                        .WORD    100115             ;DATA - 'M'
4043   032420   100060                        .WORD    100060             ;DATA - '0' (MATH OFF)
4044   032422   100053                        .WORD    100053             ;DATA - '+'
4045   032424   100060           LTOL2:  .WORD    100060             ;DATA - '0' (ADJUSTED VALUE ON TP CODE)
4046   032426   100062           LTOL0:  .WORD    100062             ;DATA - '2' (ADJUSTED VALUE ON TP CODE)
4047   032430   100065           LTOL1:  .WORD    100065             ;DATA - '5' (ADJUSTED VALUE ON TP CODE)
4048   032432   100105                        .WORD    100105             ;DATA - 'E'
4049   032434   100055                        .WORD    100055             ;DATA - '_'
4050   032436   100064                        .WORD    100064             ;DATA - '4' (+ <LTOL2!LTOL0.LTOL1> MV)
4051   032440   100123                        .WORD    100123             ;DATA - 'S'
4052   032442   100124                        .WORD    100124             ;DATA - 'T'
4053   032444   100132                        .WORD    100132             ;DATA - 'Z' (STORE IN Z)
4054   032446   100115                        .WORD    100115             ;DATA - 'M'
4055   032450   100067                        .WORD    100067             ;DATA - '7' (SCALE)
4056   032452   100124                        .WORD    100124             ;DATA - 'T'
4057   032454   100061                        .WORD    100061             ;DATA - '1' (INTERNAL TRIGGER)
4058   032456   000000                        .WORD    0                  ;TERMINATOR
4059   032460   000020                        .BLKW    20
4060
4061                             ;LIST OF COMMANDS AND DATA  TO STORE RESULT IN LOWER LIMIT REG
4062                             ;    AND PREPARE TO LOAD UPPER LIMIT REG.
4063   032520   100077           RDVINL: .WORD    100077             ;UNIVERSAL UNLISTEN
4064   032522   100042                        .WORD    100042             ;DATA - LISTNER #2
4065   032524   000144                        .WORD    144                ;COMMAND - TALK ON
4066   032526   100123                        .WORD    100123             ;DATA - 'S'
4067   032530   100124                        .WORD    100124             ;DATA - 'T'
4068   032532   100114                        .WORD    100114             ;DATA - 'L' (STORE RESULT IN LOWER LIMIT)
4069   032534   100115                        .WORD    100115             ;DATA - 'M'
```

```
4070   032536  100060                      .WORD    100060          ;DATA - '0'' (MATH OFF)
4071   032540  100055                      .WORD    100055          ;DATA - ''-''
4072   032542  100060.         HTOL2:      .WORD    100060          ;DATA - ''0'' (ADJUSTED WITH MNCTP CODE)
4073   032544  100062          HTOL0:      .WORD    100062          ;DATA - ''2'' (ADJUSTED WITH MNCTP CODE)
4074   032546  100065          HTOL1:      .WORD    100065          ;DATA - ''5'' (ADJUSTED WITH MNCTP CODE)
4075   032550  100105                      .WORD    100105          ;DATA - ''E''
4076   032552  100055                      .WORD    100055          ;DATA - ''-''
4077   032554  100064                      .WORD    100064          ;DATA - ''4'' (- <HTOL2!HTOL0.HTOL1> MV)
4078   032556  100123                      .WORD    100123          ;DATA - ''S''
4079   032560  100124                      .WORD    100124          ;DATA - ''T''
4080   032562  100132                      .WORD    100132          ;DATA - ''Z'' (SAVE IN Z REG)
4081   032564  100115                      .WORD    100115          ;DATA - ''M''
4082   032566  100067                      .WORD    100067          ;DATA - ''7'' (SCALE)
4083   032570  100124                      .WORD    100124          ;DATA - ''T''
4084   032572  100061                      .WORD    100061          ;DATA - ''1'' (INTERNAL TRIGGER)
4085   032574  000000                      .WORD    0               ;TERMINATOR
4086   032576  000020                      .BLKW    20
4087                           ;LIST OF COMMANDS AND DATA TO STORE RESULT IN UPPER LIMIT REGISTER
4088   032636  100077          RDVINU: .WORD    100077          ;UNIVERSAL UNLISTEN
4089   032640  100042                      .WORD    100042          ;DATA - LISTEN #2
4090   032642  000144 .                    .WORD    144             ;COMMAND - TALK ON
4091   032644  100123                      .WORD    100123          ;DATA - ''S''
4092   032646  100124                      .WORD    100124          ;DATA - ''T''
4093   032650  100125                      .WORD    100125          ;DATA - ''U''
4094   032652  100115                      .WORD    100115          ;DATA - ''M''
4095   032654  100060                      .WORD    100060          ;DATA - ''0'' (MATH OFF)
4096   032656  000000                      .WORD    0               ;TERMINATOR
4097   032660  000020                      .BLKW    20
4098                                        .EVEN
4099
```

```
4101                                    ;SUBROUTINE TO HANDLE IEEE DVM
4102                                    ;    ENTER WITH TRAILING ARG.  - DESTROYS R2 AND R3
4103   032720  012777  033152  000304  DVMIEE: MOV     #30$,@IBV0      ;LOAD IEEE ERROR VECTOR
4104   032726  012777  000340  000300          MOV     #340,@IBV0A
4105   032734  012777  033174  000274          MOV     #40$,@IBV1      ;LOAD IEEE SRQ VECTOR
4106   032742  012777  000340  000270          MOV     #340,@IBV1A
4107   032750  012777  033032  000264          MOV     #10$,@IBV2      ;LOAD COMMAND INTER. VECTOR
4108   032756  012777  000340  000260          MOV     #340,@IBV2A
4109   032764  012777  033110  000254          MOV     #20$,@IBV3      ;LOAD LISTNER INTER. VECOTR
4110   032772  012777  000340  000250          MOV     #340,@IBV3A
4111   033000  012502                          MOV     (R5)+,R2        ;GET COMMAND POINTER
4112   033002  012503                          MOV     (R5)+,R3        ;GET POINTER WHERE TO STORE ANSWER
4113   033004  012777  000105  000214          MOV     #105,@IBCSR     ;GENERATE ''IBC'' (IFC) THEN ''TCS''
4114   033012  012746  000000                  MOV     #0,-(SP)
4115   033016  012746  033024                  MOV     #1$,-(SP)
4116   033022  000002                          RTI                     ;FUNNY WAY TO LOWER PS
4117   033024  000777                  1$:     BR      .
4118   033026  000776                          BR      1$
4119   033030  000775                          BR      1$
4120   033032  012704  040000          10$:    MOV     #40000,R4       ;LOAD DELAY COUNTER
4121   033036  005304                  7$:     DEC     R4              ;DELAY
4122   033040  001376                          BNE     7$
4123   033042  005712                          TST     (R2)            ;TEST IF MORE DATA/COMMANDS TO BE SENT
4124   033044  001407                          BEQ     12$             ;BR IF NONE
4125   033046  100403                          BMI     11$             ;BR IF DATA TO BE SENT
4126   033050  012277  000152                  MOV     (R2)+,@IBCSR    ;LOAD COMMAND
4127   033054  000002                          RTI                     ;RETURN
4128   033056  012277  000146          11$:    MOV     (R2)+,@IBDAT    ;LOAD DATA
4129   033062  000002                          RTI
4130   033064  005703                  12$:    TST     R3              ;CHECK IF ANSWER IS EXPECTED
4131   033066  001404                          BEQ     13$             ;BR IF NON-EXPECTED
4132   033070  012777  000320  000130          MOV     #320,@IBCSR     ;SWITCH IBV TO A LISTNER
4133   033076  000002                          RTI
4134   033100  022626                  13$:    CMP     (SP)+,(SP)+     ;CLEAN STACK
4135   033102  005077  000120                  CLR     @IBCSR          ;REMOVE IE ENABLE
4136   033106  000205                          RTS     R5              ;EXIT
4137                                    ;COME HERE UPON LISTNER INTERRUPT
4138   033110  117713  000114          20$:    MOVB    @IBDAT,(R3)     ;SAVE DATA RCVD
4139   033114  122723  000012                  CMPB    #12,(R3)+       ;TEST IF END OF MESSAGE
4140   033120  001403                          BEQ     21$             ;BR WHEN RCVD DATA FINISHED
4141   033122  005077  000102                  CLR     @IBDAT          ;ISSUE A 'DAC' TO ENABLE NEXT DATA WORD
4142   033126  000002                          RTI
4143   033130  005077  000074          21$:    CLR     @IBDAT          ;LET BUS ALONE
4144   033134  005077  000066                  CLR     @IBCSR
4145   033140  162703  000006                  SUB     #6,R3           ;REMOVE 6 BYTES FROM DATA RCVD.
4146   033144  105013                          CLRB    (R3)            ;LOAD ASCII 'END OF TEXT' INDICATOR
4147   033146  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF STACK
4148   033150  000205                          RTS     R5              ;EXIT
4149                                    ;RETURN HERE IF UNEXPECTED IBV - IEEE BUS ERROR
4150   033152  005077  000050          30$:    CLR     @IBCSR          ;REMOVE ANY BITS
4151   033156  104401  040663                  TYPE    ,ERIEE          ;TELL OPERATOR ABOUT IEEE ERROR
4152   033162  042737  040000  001602          BIC     #BIT14,WFTEST   ;REMOVE TESTER BIT
4153   033170  000137  001674                  JMP     BEG2            ;RESTART PROGRAM
4154                                    ;RETURN HERE IF AN ''SRQ'' INTERRUPT (MATH 1 FAIL)
4155   033174  012737  047505  030202  40$:    MOV     #ERMSG,DECRPC   ;LOAD ERROR TEXT POINTER
4156   033202  005237  001112                  INC     $ERTTL          ;UPDATE ERROR COUNT
```

```
4157  033206  004537  032720                    JSR      R5,DVMIEE      ;TELL DVM TO REMOVE ''SRQ''
4158  033212  032310                             RDVOUE                  ;IEEE COMMAND POINTER
4159  033214  000000                             0                       ;NO ANSWER EXPECTED
4160  033216  000240                             NOP
4161  033220  000240                             NOP
4162  033222  000137  033100                     JMP      13$            ;EXIT
4163  033226  171420                    IBCSR:   171420
4164  033230  171422                    IBDAT:   171422
4165  033232  000420                    IBV0:    420
4166  033234  000422                    IBV0A:   422
4167  033236  000424                    IBV1:    424
4168  033240  000426                    IBV1A:   426
4169  033242  000430                    IBV2:    430
4170  033244  000432                    IBV2A:   432
4171  033246  000434                    IBV3:    434
4172  033250  000436                    IBV3A:   436
4173
```

```
4175                                          ;SUBROUTINE TO CHANGE BASE AND VECTOR ADDRESSES
4176  033252  104401  043355      BASEXC: TYPE    .MADR                    ;ASK FOR MODULE ADDRESS
4177  033256  013746  001244              MOV     $BASE,-(SP)      ;;SAVE $BASE FOR TYPEOUT
 (1)  033262  104402                       TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4178  033264  104401  043447              TYPE    .ENCOM
4179  033270  104413                       RDOCT
4180  033272  005726                       TST     (SP)+                    ;DEFAULT ADDRESS ?
4181  033274  001403                       BEQ     5$                       ;NO BRANCH
4182  033276  016637  177776  001244       MOV     -2(SP),$BASE             ;SAVE ADDRESS IN $BASE
4183  033304  104401  043411      5$:      TYPE    .MVCT                    ;ASK FOR MODULE VECTOR
4184  033310  013701  001240              MOV     $VECT1,R1                ;GET VECTOR
4185  033314  010146                       MOV     R1,-(SP)         ;;SAVE R1 FOR TYPEOUT
 (1)  033316  104403                       TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  033320     003                       .BYTE   3                ;;TYPE 3 DIGIT(S)
 (1)  033321     001                       .BYTE   1                ;;TYPE LEADING ZEROS
4186  033322  104401  043447              TYPE    .ENCOM
4187  033326  104413                       RDOCT
4188  033330  005726                       TST     (SP)+                    ;TAKE DEFAULT ?
4189  033332  001403                       BEQ     7$
4190  033334  016637  177776  001240       MOV     -2(SP),$VECT1
4191  033342  052737  100000  001240  7$:  BIS     #BIT15,$VECT1            ;SET PRIORITY LEVEL
4192  033350  000137  003052              JMP     MTEST1                   ;RESTART
```

```
4194
4198                                    .SBTTL         DETERMINE IF MORE MNCAD'S TO BE TESTED
4199   033354  005737  001550           BUMPAD: TST    NBEXT                    ;ADDITIONAL AD'S?
4200   033360  001433                           BEQ    FIXADR                   ;NO-INITIALIZE ADDRESSES
4201   033362  006337  001620                   ASL    MASKNM           ;MOVE BIT TO NEXT MODULE
4202   033366  005001                           CLR    R1
4203   033370  013700  001620                   MOV    MASKNM,R0                ;GET MASK NUMBER
4204   033374  006200           1$:             ASR    R0                       ;MOVE RIGHT
4205   033376  001403                           BEQ    2$                       ;BR IF DONE
4206   033400  062701  000004                   ADD    #4,R1                    ;UPDATE INDEX VALUE
4207   033404  000773                           BR     1$
4208   033406  016137  001436  001456  2$:      MOV    MNCAD0(R1),STREG         ;GET NEW ADDRESS
4209   033414  062701  000002                   ADD    #2,R1                    ;NEW NEXT INDEX
4210   033420  016137  001436  001464           MOV    MNCAD0(R1),VECTOR        ;GET NEW VECTOR
4211   033426  013737  001456  001460           MOV    STREG,ADST1              ;PRIME OTHER ADDRESSES
4212   033434  013737  001456  001462           MOV    STREG,ADBUFF
4213   033442  005337  001550                   DEC    NBEXT                    ;ONE LESS MNCAD
4214   033446  000427                           BR     BYPASS
4215   033450  062716  000002           FIXADR: ADD    #2,(SP)
4216   033454  012737  040354  000004   FIXONE: MOV    #IOTRD,@#ERRVEC ;SET UP ERRVEC
4217   033462  012737  000001  001620           MOV    #1,MASKNM                ;INIT. MODULE ERROR TEST BIT
4218   033470  013737  001244  001456           MOV    $BASE,STREG              ;RELOAD INITIAL ADDRESSES
4219   033476  013737  001244  001460           MOV    $BASE,ADST1
4220   033504  013737  001244  001462           MOV    $BASE,ADBUFF
4221   033512  013737  001240  001464           MOV    $VECT1,VECTOR            ;GET DEFAULT VECTOR
4222   033520  013737  001552  001550           MOV    NMBEXT,NBEXT             ;RESET UNIT COUNTER
4223   033526  005237  001460           BYPASS: INC    ADST1
4224   033532  062737  000002  001462           ADD    #2,ADBUFF
4225   033540  042737  170000  001464           BIC    #170000,VECTOR
4226   033546  013737  001464  001466           MOV    VECTOR,VECTR1
4227   033554  062737  000002  001466           ADD    #2,VECTR1
4228   033562  013737  001464  001470           MOV    VECTOR,VECTR2
4229   033570  062737  000004  001470           ADD    #4,VECTR2
4230   033576  013737  001464  001472           MOV    VECTOR,VECTR3
4231   033604  062737  000006  001472           ADD    #6,VECTR3
4232   033612  004737  034132                   JSR    PC,SETINT        ;;LOAD .+2 AND JSR PC,R0 TRAP CATCHER;;
4233   033616  004737  055712                   JSR    PC,WHICHU        ;DETERMINE UNIT #
4234   033622  000207                           RTS    PC                       ;TEST NEXT A/D
```

```
4236  033624  012777  000200  145630  TYPSET: MOV     #200,@ADBUFF        ;ENSURE NORMAL VER. DAC
4237  033632  012737  046620  033776          MOV     #OKMSG,11$          ;PRIME FOR ''OK'' MESSAGE
4238  033640  020437  037702                  CMP     R4,VSET             ;TEST IF WITHIN LIMITS
4239  033644  003003                          BGT     3$                  ;BR IF OUTSIDE
4240  033646  005737  001624                  TST     QUIET               ;TEST IF QUIET MODE
4241  033652  001052                          BNE     12$                 ;BR IF YES
4242  033654  104416          3$:             TYPDC
4243  033656  104401  046507                  TYPE    .LSB
4244  033662  013746  001546                  MOV     CH2,-(SP)           ;;SAVE CH2 FOR TYPEOUT
 (1)                                                                      ;;TYPE CH
 (1)  033666  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
 (1)  033670     002                          .BYTE   2                   ;;TYPE 2 DIGIT(S)
 (1)  033671     000                          .BYTE   0                   ;;SUPPRESS LEADING ZEROS
4245  033672  104401  046544                  TYPE    .ATMSG              ;TYPE ASCIZ STRING
4246  033676  004737  034066                  JSR     PC,TYPEDG
4247  033702  104401  046522                  TYPE    .SETCH
4248  033706  013746  001544                  MOV     CH1,-(SP)           ;;SAVE CH1 FOR TYPEOUT
 (1)                                                                      ;;TYPE CH
 (1)  033712  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
 (1)  033714     002                          .BYTE   2                   ;;TYPE 2 DIGIT(S)
 (1)  033715     000                          .BYTE   0                   ;;SUPPRESS LEADING ZEROS
4249  033716  104401  046544                  TYPE    .ATMSG
4250  033722  013737  001544  033734          MOV     CH1,1$
4251  033730  004537  036332                  JSR     R5,CONVRT           ;SAMPLE THE CHANNEL
4252  033734  000000          1$:             0
4253  033736  013746  001540                  MOV     TEMP,-(SP)          ;;SAVE TEMP FOR TYPEOUT
 (1)                                                                      ;;TYPE VALUE
 (1)  033742  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
 (1)  033744     004                          .BYTE   4                   ;;TYPE 4 DIGIT(S)
 (1)  033745     001                          .BYTE   1                   ;;TYPE LEADING ZEROS
4254  033746  020437  037702                  CMP     R4,VSET             ;TEST IF WITHIN LIMITS
4255  033752  003001                          BGT     2$                  ;BR IF OUTSIDE LIMITS
4256  033754  000407                          BR      10$                 ;BR FOR GOOD REPORT
4257  033756  012737  047505  033776  2$:     MOV     #ERMSG,11$          ;MAKE ERROR MESSAGE HAPPEN
4258  033764  004737  055704                  JSR     PC,WHICHV           ;INDICATE BAD UNIT
4259  033770  005237  001112                  INC     $ERTTL              ;UPDATE ERROR COUNT
4260  033774  104401          10$:            TYPE                        ;TELL OPERATOR THE NEWS
4261  033776  046620          11$:    OKMSG   ;ERMSG                      ; ''OK'' OR 'ERROR''
4262  034000  000207          12$:    RTS     PC                          ;EXIT
4263
4264                                  ;;SUBROUTINE FOR SETTLING TESTS;;
4265  034002  012737  034064  015362  SET1A:  MOV     #1$,ERRADR          ;SET UP ERROR RECOVERY ADDRESS
4266  034010  013737  001546  001554          MOV     CH2,DUMMY           ;LOAD DUMMY
4267  034016  004537  034170                  JSR     R5,SARSUB           ;DO SAR ROUTINE AT 50%
4268  034022  000062                          50.
4269  034024  004737  015336                  JSR     PC,TSTDAC           ;CHECK VERNIER DAC SETTING
4270  034030  063702  001570                  ADD     DAC,R2              ;ADD RESULT TO R2
4271  034034  013737  001544  001554          MOV     CH1,DUMMY           ;CHANGE DUMMY VALUE
4272  034042  004537  034170                  JSR     R5,SARSUB           ;DO SAR ROUTINE AT 50%
4273  034046  000062                          50.
4274  034050  004737  015336                  JSR     PC,TSTDAC           ;CHECK VERNIER DAC SETTING
4275  034054  163702  001570                  SUB     DAC,R2              ;SUBTRACT RESULT FROM R2
4276  034060  062716  000002                  ADD     #2,(SP)             ;BUMP RETURN ADDRESS TO SKIP OVER BRANCH
4277  034064  000207          1$:     RTS     PC                          ;RETURN
```

```
4279                                      ;;SUBROUTINE TO TYPE EDGE VALUES;;
4280   034066  013703  001574    TYPEDG: MOV     EDGE,R3
4281   034072  010346                     MOV     R3,-(SP)             ;;SAVE R3 FOR TYPEOUT
 (1)                                                                   ;;TYPE OCTAL VALUE OF EDGE
 (1)   034074  104403                     TYPOS                        ;;GO TYPE--OCTAL ASCII
 (1)   034076     004                     .BYTE   4                    ;;TYPE 4 DIGIT(S)
 (1)   034077     001                     .BYTE   1                    ;;TYPE LEADING ZEROS
4282   034100  023727  034130  000001     CMP     EDGFLG,#1
4283   034106  001407                     BEQ     RET
4284   034110  062703  000007             ADD     #7,R3
4285   034114  104401  042154             TYPE    ,MINUS               ;TYPE ASCIZ STRING
4286   034120  010346                     MOV     R3,-(SP)             ;;SAVE R3 FOR TYPEOUT
 (1)                                                                   ;;TYPE EDGE VALUE
 (1)   034122  104403                     TYPOS                        ;;GO TYPE--OCTAL ASCII
 (1)   034124     004                     .BYTE   4                    ;;TYPE 4 DIGIT(S)
 (1)   034125     001                     .BYTE   1                    ;;TYPE LEADING ZEROS
4287   034126  000207            RET:      RTS     PC
4288   034130  000000            EDGFLG: 0
4289                             ;SUBROUTINE TO LOAD VECTOR AREA WITH TRAP CATCHER
4290   034132  012700  000222    SETINT: MOV     #222,R0              ;LOAD UP POINTER
4291   034136  012701  000220             MOV     #220,R1              ;LOAD ADDRESS
4292   034142  010021            2$:       MOV     R0,(R1)+             ;LOAD POINTER TO NEXT WORD
4293   034144  012721  004700             MOV     #4700,(R1)+          ;LOAD 'BAD' INSTRUCTION
4294   034150  010100                     MOV     R1,R0                ;LOAD NEW ADDRESS POINTER
4295   034152  005720                     TST     (R0)+                ;BUMP VALUE
4296   034154  022700  001002             CMP     #1002,R0             ;FINISHED?
4297   034160  001370                     BNE     2$                   ;BR IF NOT
4298   034162  000240                     NOP
4299   034164  000240                     NOP
4300   034166  000207                     RTS     PC                   ;EXIT
```

```
4302
4303
4304                                    ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
4305                                    ;CALL=JSR    R5,SARSUB
4306                                    ;   XXX;XXX=PERCENT
4307                                    ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
4308  034170  012537  001610           SARSUB: MOV    (R5)+,PERCNT            ;GET PERCENT
4309  034174  006337  001610                   ASL    PERCNT
4310  034200  006337  001610                   ASL    PERCNT                  ;RESCALE PERCENT FOR 1600.
4311  034204  006337  001610                   ASL    PERCNT                  ;POINTS PER BURST
4312  034210  006337  001610                   ASL    PERCNT
4313  034214  012737  000200  001576           MOV    #200,BITPNT             ;INITIALIZE BIT POINTER AT MSB
4314  034222  005037  001570                   CLR    DAC                     ;INITIALIZE DAC VALUE
4315  034226  005000           TRY:            CLR    R0
4316  034230  063737  001576  001570           ADD    BITPNT,DAC              ;TRY BIT
4317  034236  013777  001570  145216           MOV    DAC,@ADBUFF
4318  034244  012701  003100                   MOV    #1600.,R1               ;SET UP FOR 1600. CONVERSIONS
4319  034250  113777  001554  145202  NXTCVT:  MOVB   DUMMY,@ADST1            ;PRESET MUX TO DUMMY CHANNEL
4320  034256  012777  001650  145200           MOV    #RETURN,@VECTOR         ;RETURN ADDRESS
4321  034264  052777  000101  145164           BIS    #101,@STREG             ;CONVERSION ON DUMMY CHANNEL
4322  034272  000001                            WAIT                          ;WAIT FOR INTERRUPT
4323  034274  017704  145162                    MOV   @ADBUFF,R4              ;DUMMY READ
4324  034300  013704  001556                    MOV   CHANL,R4
4325  034304  000304                            SWAB  R4
4326  034306  052704  000101                    BIS   #101,R4                 ;INTERRUPT ENABLE START
4327  034312  010477  145140                    MOV   R4,@STREG               ;JUMP TO CHANNEL + START CONVERT
4328  034316  000001                            WAIT                          ;WAIT FOR INTERRUPT
4329  034320  027737  145136  001574            CMP   @ADBUFF,EDGE
4330  034326  002001                            BGE   2$
4331  034330  005200                            INC   R0                      ;COUNT RESULTS .LT. EDGE
4332  034332  005301           2$:              DEC   R1
4333  034334  001345                            BNE   NXTCVT
4334  034336  020037  001610                    CMP   R0,PERCNT
4335  034342  003003                            BGT   SHIFT
4336  034344  163737  001576  001570            SUB   BITPNT,DAC              ;TAKE THE BIT OUT
4337  034352  006237  001576   SHIFT:           ASR   BITPNT
4338  034356  001323                            BNE   TRY
4339  034360  000205                            RTS   R5
4340
4341                                    ;ROUTINE TO DELAY IF PROCESSER CAN NOT DO SOB INSTRUCTION
4342
4343  034362  005300           DELAY4: DEC      R0                      ;DECREMENT R0, IS IT ZERO?
4344  034364  001376                   BNE      DELAY4                  ;NO
4345  034366  000002                   RTI                              ;RETURN
```

```
4347                                       ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
4348                                       ;''CHA'' CONTAINS THE CHANNEL NUMBER
4349   034370  104401  050110     DIFLIN:  TYPE    ,MSG20             ;IDENTIFY TEST
4350   034374  004737  055712              JSR     PC,WHICHU          ;DETERMINE UNIT #
4351   034400  013746  001622              MOV     UNITBD,-(SP)
4352   034404  104403                      TYPOS                      ;TELL OPER. THE #
4353   034406     001     000              .BYTE   1,0
4354   034410  104401  046601              TYPE    ,CHAN              ;TELL OPER THE CHANNEL NUMBER
4355   034414  013746  023176              MOV     CHA,-(SP)          ;LOAD NUMBER
4356   034420  104403                      TYPOS                      ;TELL OPER. THE #
4357   034422     002     000              .BYTE   2,0
4358   034424  104401  001165              TYPE    ,$CRLF
4359   034430  012702  062341              MOV     #62341,R2          ;SET UP RANDOM NUMBER GENERATOR
4360   034434  012704  142315              MOV     #142315,R4
4361   034440  012705  127623              MOV     #127623,R5
4362   034444  012700  061306              MOV     #BUFFER,R0
4363   034450  012701  010000              MOV     #4096.,R1          ;4096 WORDS FOR HISTOGRAM
4364   034454  005020            CLEAR1:   CLR     (R0)+              ;CLEAR BUFFER AREA
4365   034456  005301                      DEC     R1
4366   034460  001375                      BNE     CLEAR1
4367   034462  012700  060466              MOV     #DIST,R0           ;DISTRIBUTION BUFFER POINTER
4368   034466  012701  000310              MOV     #200.,R1           ;200. WORDS FOR DISTRIBUTION
4369   034472  005003                      CLR     R3
4370   034474  005037  001612              CLR     OUT
4371   034500  005037  001530              CLR     WIDE
4372   034504  005037  001532              CLR     NARROW
4373   034510  005037  001534              CLR     FIRST
4374   034514  005037  001536              CLR     SKIPST
4375   034520  005020            CLEAR2:   CLR     (R0)+              ;CLEAR DISTRIBUTION BUFFER AREA
4376   034522  005301                      DEC     R1
4377   034524  001375                      BNE     CLEAR2
4378   034526  013700  023176              MOV     CHA,R0             ;CHANNEL 3
4379   034532  000300                      SWAB    R0                 ;LOAD MUX BITS
4380   034534  052700  000100              BIS     #100,R0
4381   034540  010077  144712              MOV     R0,@STREG
4382   034544  012737  001440  001572      MOV     #800.,DELAY        ;NOMINAL STATE WIDTH - 1 LSB
4383   034552  012777  001660  144704      MOV     #RET1,@VECTOR
4384   034560  012701  007776     AGAIN:   MOV     #4094.,R1
4385   034564  060402            NEXT1:    ADD     R4,R2              ;GENERATE A RANDOM NUMBER
4386   034566  060502                      ADD     R5,R2
4387   034570  005502                      ADC     R2
4388   034572  010200                      MOV     R2,R0              ;PUT RANDOM NUMBER IN R0
4389   034574  042700  177770              BIC     #177770,R0         ;MASK IT TO 3 BITS ONLY
4390   034600  001401                      BEQ     CONVR1
4391   034602  077001            DELAY1:   SOB     R0,DELAY1          ;STALL TIME
4392   034604  005277  144646     CONVR1:  INC     @STREG             ;START CONVERSION
4393   034610  000001                      WAIT
4394   034612  000240                      NOP
4395   034614  017700  144642              MOV     @ADBUFF,R0         ;GET CONVERTED VALUE
4396   034620  001416                      BEQ     LODLY1             ;IGNORE IF =0
4397   034622  020027  007777              CMP     R0,#7777           ;IGNORE IF =7777
4398   034626  001416                      BEQ     HIDLY1
4399   034630  006300                      ASL     R0
4400   034632  005260  061306              INC     BUFFER(R0)         ;MAKE HISTOGRAM
4401   034636  100016                      BPL     OKAY1
4402   034640  012760  077777  061306      MOV     #077777,BUFFER(R0)    ;PREVENT OVERFLOW
```

```
4403   034646   000412                         BR       OKAY1
4404   034650   005037  001540      NOTOK1:    CLR      TEMP
4405   034654   000407                         BR       OKAY1
4406   034656   020027  007777      LODLY1:    CMP      R0,#7777        ;EQUALIZE LOOP TIME
4407   034662   001400                         BEQ      HIDLY1          ;WITH DUMMY INSTR.
4408   034664   005201              HIDLY1:    INC      R1
4409   034666   005263  001540                 INC      TEMP(R3)
4410   034672   100766                         BMI      NOTOK1
4411   034674   005301              OKAY1:     DEC      R1
4412   034676   001514                         BEQ      AROUND
4413   034700   060204                         ADD      R2,R4           ;GENERATE A RANDOM NUMBER
4414   034702   060504                         ADD      R5,R4
4415   034704   005504                         ADC      R4
4416   034706   010400                         MOV      R4,R0           ;PUT RANDOM NUMBER IN R0
4417   034710   042700  177770                 BIC      #177770,R0      ;MASK IT TO 3 BITS ONLY
4418   034714   001401                         BEQ      CONVR2
4419   034716   077001              DELAY2:    SOB      R0,DELAY2       ;STALL TIME
4420   034720   005277  144532      CONVR2:    INC      @STREG          ;START CONVERSION
4421   034724   000001                         WAIT
4422   034726   000240                         NOP
4423   034730   017700  144526                 MOV      @ADBUFF,R0      ;GET CONVERTED VALUE
4424   034734   001416                         BEQ      LODLY2          ;IGNORE IF =0
4425   034736   020027  007777                 CMP      R0,#7777        ;IGNORE IF =7777
4426   034742   001416                         BEQ      HIDLY2
4427   034744   006300                         ASL      R0
4428   034746   005260  061306                 INC      BUFFER(R0)      ;MAKE HISTOGRAM
4429   034752   100016                         BPL      OKAY2
4430   034754   012760  077777  061306         MOV      #077777,BUFFER(R0)       ;PREVENT OVERFLOW
4431   034762   000412                         BR       OKAY2
4432   034764   005037  001540      NOTOK2:    CLR      TEMP
4433   034770   000407                         BR       OKAY2
4434   034772   020027  007777      LODLY2:    CMP      R0,#7777        ;EQUALIZE LOOP TIME
4435   034776   001400                         BEQ      HIDLY2          ;WITH DUMMY INSTR.
4436   035000   005201              HIDLY2:    INC      R1
4437   035002   005263  001540                 INC      TEMP(R3)
4438   035006   100766                         BMI      NOTOK2
4439   035010   005301              OKAY2:     DEC      R1
4440   035012   001446                         BEQ      AROUND
4441   035014   060205                         ADD      R2,R5           ;GENERATE A RANDOM NUMBER
4442   035016   060405                         ADD      R4,R5
4443   035020   005505                         ADC      R5
4444   035022   010500                         MOV      R5,R0           ;PUT RANDOM NUMBER IN R0
4445   035024   042700  177770                 BIC      #177770,R0      ;MASK IT TO 3 BITS ONLY
4446   035030   001401                         BEQ      CONVR3
4447   035032   077001              DELAY3:    SOB      R0,DELAY3       ;STALL TIME
4448   035034   005277  144416      CONVR3:    INC      @STREG          ;START CONVERSION
4449   035040   000001                         WAIT
4450   035042   000240                         NOP
4451   035044   017700  144412                 MOV      @ADBUFF,R0      ;GET CONVERTED VALUE
4452   035050   001416                         BEQ      LODLY3          ;IGNORE IF =0
4453   035052   020027  007777                 CMP      R0,#7777        ;IGNORE IF =7777
4454   035056   001416                         BEQ      HIDLY3
4455   035060   006300                         ASL      R0
4456   035062   005260  061306                 INC      BUFFER(R0)      ;MAKE HISTOGRAM
4457   035066   100016                         BPL      OKAY3
4458   035070   012760  077777  061306         MOV      #077777,BUFFER(R0)       ;PREVENT OVERFLOW
```

```
4459    035076  000412                  BR      OKAY3
4460    035100  005037  001540  NOTOK3: CLR     TEMP
4461    035104  000407                  BR      OKAY3
4462    035106  020027  007777  LODLY3: CMP     R0,#7777        ;EQUALIZE LOOP TIME
4463    035112  001400                  BEQ     HIDLY3          ;WITH DUMMY INSTR.
4464    035114  005201          HIDLY3: INC     R1
4465    035116  005263  001540          INC     TEMP(R3)
4466    035122  100766                  BMI     NOTOK3
4467    035124  005301          OKAY3:  DEC     R1
4468    035126  001216                  BNE     NEXT1
4469    035130  005337  001572  AROUND: DEC     DELAY
4470    035134  001211                  BNE     AGAIN
4471                            ;TAKE THE CONTENTS OF THE ACQUIRED DATA BUFFER AND TEST IF WITHIN CERTAIN LIMITS
4472                            ;AND CREATE A STATE DISTRIBUTION BUFFER AND SORT THE VALUES INTO 'BINS'
4473    035136  012700  007776          MOV     #4094.,R0
4474    035142  012701  061310          MOV     #BUFFER+2,R1
4475    035146  012102          READ:   MOV     (R1)+,R2        ;GET STATE WIDTH
4476    035150  006202                  ASR     R2              ;1 LSB = 800.
4477    035152  006202                  ASR     R2
4478    035154  006202                  ASR     R2
4479    035156  005502                  ADC     R2              ;1 LSB = 100.
4480    035160  020227  000310          CMP     R2,#200.        ;OUT OF RANGE?
4481    035164  002403                  BLT     INRNGE
4482    035166  005237  001612          INC     OUT             ;YES - INCREMENT COUNTER
4483    035172  000423                  BR      TYPBAD
4484    035174  006302          INRNGE: ASL     R2
4485    035176  005262  060466          INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
4486    035202  006202                  ASR     R2
4487    035204  020227  000062          CMP     R2,#50.         ;IS IT 1/2 LSB?
4488    035210  002007                  BGE     NOTNAR
4489    035212  005237  001532          INC     NARROW
4490    035216  005702                  TST     R2              ;IS IT A SKIPPED STATE?
4491    035220  001002                  BNE     31$
4492    035222  005237  001536          INC     SKIPST
4493    035226  000405          31$:    BR      TYPBAD
4494    035230  020227  000226  NOTNAR: CMP     R2,#150.        ;IS IT 1.5 LSB?
4495    035234  003425                  BLE     LAST
4496    035236  005237  001530          INC     WIDE
4497    035242  005737  001534  TYPBAD: TST     FIRST
4498    035246  001004                  BNE     60$
4499    035250  005237  001534          INC     FIRST
4500    035254  104401  046457          TYPE    .STATE
4501    035260  010103          60$:    MOV     R1,R3
4502    035262  162703  061310          SUB     #BUFFER+2,R3
4503    035266  006203                  ASR     R3
4504    035270  010346                  MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
(1)                                                             ;;TYPE STATE
(1)     035272  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
(1)     035274  004                     .BYTE   4               ;;TYPE 4 DIGIT(S)
(1)     035275  001                     .BYTE   1               ;;TYPE LEADING ZEROS
4505    035276  104401  046453          TYPE    .DASH
4506    035302  104416                  TYPDC
4507    035304  104401  046444          TYPE    .LSBMSG
4508    035310  005300          LAST:   DEC     R0
4509    035312  001315                  BNE     READ
```

K 10
CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 68
CVMNAC.P11   10-JUL-81 14:30   DETERMINE IF MORE MNCAD'S TO BE TESTED

SEQ 0127

```
4511                                      ;REPORT TO THE OPERATOR THE DIFFERENT STATE VALUES
4512                                      ;       IN THE FORM OF A GENERAL STATUS AND INDICATE OK/ERROR
4513   035314  112737  000177  053610            MOVB    #177,DECPNT
4514   035322  013702  001536                    MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
4515   035326  104416                            TYPDC                           ;TYPE IT
4516   035330  104401  047522                    TYPE    .SKPMSG         ;TYPE MESSAGE
4517   035334  005737  001536                    TST     SKIPST
4518   035340  001407                            BEQ     1$
4519   035342  104401  047505                    TYPE    .ERMSG          ;TYPE 'ERROR'
4520   035346  004737  055704                    JSR     PC,WHICHV       ;INDICATE BAD UNIT
4521   035352  005237  001112                    INC     $ERTTL          ;UPDATE ERROR COUNT
4522   035356  000402                            BR      NAR
4523   035360  104401  046620            1$:     TYPE    .OKMSG          ;TYPE #OK#
4524   035364  013702  001532            NAR:    MOV     NARROW,R2               ;GET NO. OF NARROW STATES
4525   035370  104416                            TYPDC                           ;TYPE IT
4526   035372  104401  047601                    TYPE    .NARMSG         ;TYPE MESSAGE
4527   035376  013702  001530                    MOV     WIDE,R2
4528   035402  063702  001612                    ADD     OUT,R2
4529   035406  104416                            TYPDC                           ;TYPE NO. OF WIDE STATES
4530   035410  104401  047640                    TYPE    .WIDMSG         ;TYPE MESSAGE
4531   035414  013702  001612                    MOV     OUT,R2
4532   035420  104416                            TYPDC                           ;TYPE NO. OF STATES OUTSIDE 2 LSB
4533   035422  104401  047677                    TYPE    .OUTMSG         ;TYPE MESSAGE
4534   035426  005737  001612                    TST     OUT
4535   035432  001407                            BEQ     11$
4536   035434  104401  047505                    TYPE    .ERMSG          ;TYPE 'ERROR'
4537   035440  004737  055704                    JSR     PC,WHICHV       ;DETERMINE BAD UNIT
4538   035444  005237  001112                    INC     $ERTTL          ;UPDATE ERROR COUNT
4539   035450  000402                            BR      HALF
4540   035452  104401  046620            11$:    TYPE    .OKMSG          ;TYPE 'OK'
4541   035456  013702  001532            HALF:   MOV     NARROW,R2
4542   035462  063702  001530                    ADD     WIDE,R2
4543   035466  063702  001612                    ADD     OUT,R2
4544   035472  010200                            MOV     R2,R0
4545   035474  104416                            TYPDC                           ;TYPE NO. OF STATES OUTSIDE LIMITS
4546   035476  112737  000056  053610            MOVB    #56,DECPNT
4547   035504  104401  047732                    TYPE    .HAFMSG
4548   035510  020027  000051                    CMP     R0,#41.         ;COMPARE IT TO NOMINAL
4549   035514  003407                            BLE     21$
4550   035516  104401  047505                    TYPE    .ERMSG          ;TYPE 'ERROR'
4551   035522  004737  055704                    JSR     PC,WHICHV       ;INDICATE BAD UNIT
4552   035526  005237  001112                    INC     $ERTTL          ;UPDATE ERROR COUNT
4553   035532  000402                            BR      SWDIST
4554   035534  104401  046620            21$:    TYPE    .OKMSG          ;TYPE 'OK'
```

```
4556                                    ;DETERMINE IF VT55 TYPE TERMINAL IS CONNECTED
4557                                    ;         IF NOT BYPASS THIS SECTION
4558                                    ;         IF VT55/VT105 GRAHIC TERMINAL REPORT THE DISTRIBUTION CURVE
4559   035540   005737   001564  SWDIST: TST    VTFLAG           ;BIT MAP TERMINAL AVAILABLE?
4560   035544   001426           BEQ    RELACC           ;BR IF NOT
4561   035546   004737   036234          JSR    PC,DELCLR        ;WAIT AWHILE, THEN CLEAR BIT MAP TERMINAL
4562   035552   104401   050162          TYPE   ,MSG16
4563   035556   104401   050663          TYPE   ,BUFF1           ;TYPE BUFF1-PRINT GRID
4564   035562   012700   060466          MOV    #DIST,R0         ;POINTER TO STATE WIDTH DISTRIBUTION
4565   035566   012701   000310          MOV    #200.,R1         ;GO 200. TIMES UP TO 2 LSB
4566   035572   012002           NXTY1:  MOV    (R0)+,R2
4567   035574   004737   036726          JSR    PC,LOADY
4568   035600   005002           CLR    R2
4569   035602   004737   036726          JSR    PC,LOADY
4570   035606   005301           DEC    R1
4571   035610   001370           BNE    NXTY1
4572   035612   104401   050621          TYPE   ,C2              ;TYPE ASCIZ STRING
4573   035616   004737   036234          JSR    PC,DELCLR
```

```
4575                                 ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR
4576
4577  035622  005001           RELACC: CLR    R1                  ;RUNNING ERROR = 0
4578  035624  005003                   CLR    R3                  ;MAXIMUM ERROR = 0
4579  035626  104401  050507           TYPE   ,MSG21
4580  035632  012700  061310           MOV    #BUFFER+2,R0
4581  035636  011002           NXTSTA: MOV    (R0),R2             ;STATE WIDTH = R2
4582  035640  162702  001440           SUB    #800.,R2            ;STATE WIDTH ERROR IN R2
4583  035644  060201                    ADD    R2,R1               ;UPDATE RUNNING ERROR
4584  035646  010120                    MOV    R1,(R0)+            ;SAVE IN BUFFER
4585  035650  010104                    MOV    R1,R4               ;SAVE IN R4 ALSO
4586  035652  100001                    BPL    PLUS                ;IS IT POSITIVE?
4587  035654  005404                    NEG    R4                  ;NO - MAKE IT POSITIVE
4588  035656  020403           PLUS:   CMP    R4,R3               ;CHECK AGAINST PREVIOUS MAX. ERROR
4589  035660  003405                    BLE    NOTNEW              ;NOT A NEW MAXIMUM
4590  035662  010403                    MOV    R4,R3               ;UPDATE MAXIMUM IN R3
4591  035664  010005                    MOV    R0,R5
4592  035666  162705  061310           SUB    #BUFFER+2,R5
4593  035672  006205                    ASR    R5                  ;R5=EDGE VALUE AT MAX. RELACC
4594  035674  020027  101304   NOTNEW: CMP    R0,#BUFFER+8190.   ;DONE?
4595  035700  001356                    BNE    NXTSTA              ;NO - REPEAT
4596  035702  006203                    ASR    R3                  ;RESCALE FROM 1 LSB = 800. SCALING
4597  035704  006203                    ASR    R3                  ;TO 1 LSB = 100. SCALING
4598  035706  006203                    ASR    R3
4599  035710  005503                    ADC    R3
4600  035712  010302                    MOV    R3,R2
4601  035714  104416                    TYPDC
4602  035716  104401  050534           TYPE   ,LINEA
4603  035722  010546                    MOV    R5,-(SP)            ;;SAVE R5 FOR TYPEOUT
 (1)                                                               ;;TYPE VALUE
 (1)  035724  104403                    TYPOS                      ;;GO TYPE--OCTAL ASCII
 (1)  035726     004                    .BYTE  4                   ;;TYPE 4 DIGIT(S)
 (1)  035727     001                    .BYTE  1                   ;;TYPE LEADING ZEROS
4604  035730  104401  046616           TYPE   ,SLASH                        ;PRINT '/'
4605  035734  005205                    INC    R5
4606  035736  010546                    MOV    R5,-(SP)            ;;SAVE R5 FOR TYPEOUT
 (1)                                                               ;;TYPE VALUE
 (1)  035740  104403                    TYPOS                      ;;GO TYPE--OCTAL ASCII
 (1)  035742     004                    .BYTE  4                   ;;TYPE 4 DIGIT(S)
 (1)  035743     001                    .BYTE  1                   ;;TYPE LEADING ZEROS
4607  035744  020337  037704           CMP    R3,VLIN
4608  035750  003407                    BLE    41$
4609  035752  104401  047505           TYPE   ,ERMSG
4610  035756  004737  055704           JSR    PC,WHICHV           ;INDICATE BAD UNIT
4611  035762  005237  001112           INC    $ERTTL              ;UPDATE ERROR COUNT
4612  035766  000402                    BR     42$
4613  035770  104401  046620   41$:    TYPE   ,OKMSG
4614  035774  005737  001564   42$:    TST    VTFLAG              ;BIT MAP TERMINAL ?
4615  036000  001503                    BEQ    LO2                 ;BR IF NOT
4616  036002  012700  061306           MOV    #BUFFER,R0
4617  036006  012701  010000           MOV    #4096.,R1
```

```
4619  036012  011002           GETDAT: MOV     (R0),R2           ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
4620  036014  006202                   ASR     R2                ;RESCALE IT TO 1 LSB = 100.
4621  036016  006202                   ASR     R2
4622  036020  006202                   ASR     R2
4623  036022  005502                   ADC     R2
4624  036024  062702  000166           ADD     #118.,R2          ;AND MOVE IT TO MID-SCREEN
4625  036030  010220                   MOV     R2,(R0)+          ;PUT IT BACK INTO BUFFER
4626  036032  005301                   DEC     R1
4627  036034  001366                   BNE     GETDAT
4628  036036  012700  061306           MOV     #BUFFER,R0
4629  036042  012704  061306           MOV     #BUFFER,R4
4630  036046  012705  061310           MOV     #BUFFER+2,R5
4631  036052  012701  001000           MOV     #512.,R1
4632  036056  012702  000007   NXT8:   MOV     #7.,R2
4633  036062  012003                   MOV     (R0)+,R3
4634  036064  010337  001600           MOV     R3,MIN            ;MINIMUM
4635  036070  010337  001606           MOV     R3,MAX            ;MAXIMUM
4636  036074  012003   NXTCMP: MOV     (R0)+,R3
4637  036076  020337  001600           CMP     R3,MIN
4638  036102  002002                   BGE     MAXTST
4639  036104  010337  001600           MOV     R3,MIN            ;NEW MINIMUM
4640  036110  020337  001606   MAXTST: CMP     R3,MAX
4641  036114  003402                   BLE     TST8
4642  036116  010337  001606           MOV     R3,MAX            ;NEW MAXIMUM
4643  036122  005302           TST8:   DEC     R2
4644  036124  001363                   BNE     NXTCMP
4645  036126  013724  001600           MOV     MIN,(R4)+
4646  036132  013725  001606           MOV     MAX,(R5)+
4647  036136  022425                   CMP     (R4)+,(R5)+       ;BUMP EACH ONCE MORE
4648  036140  005301                   DEC     R1
4649  036142  001345                   BNE     NXT8
4650  036144  104401  050050           TYPE    ,MSG18
4651  036150  104401  050711           TYPE    ,BUFF2            ;TYPE BUFF2
4652  036154  012700  061306           MOV     #BUFFER,R0
4653  036160  004737  036212           JSR     PC,LOAD
4654  036164  104401  050627           TYPE    ,C3               ;TYPE ASCIZ STRING
4655  036170  012700  061310           MOV     #BUFFER+2,R0
4656  036174  004737  036212           JSR     PC,LOAD
4657  036200  104401  050621           TYPE    ,C2               ;TYPE ASCIZ STRING
4658  036204  004737  036234           JSR     PC,DELCLR
4659  036210  000207           LO2:    RTS     PC
4660  036212  012701  001000   LOAD:   MOV     #512.,R1
4661  036216  012002           LOAD0:  MOV     (R0)+,R2
4662  036220  005720                   TST     (R0)+
4663  036222  004737  036726           JSR     PC,LOADY
4664  036226  005301                   DEC     R1
4665  036230  001372                   BNE     LOAD0
4666  036232  000207                   RTS     PC
```

```
4668  036234  032777  002000  142676  DELCLR: BIT    #BIT10,@SWR     ;TEST FOR HALT FOR DISPLAY
4669  036242  001402                          BEQ    1$              ;;DON'T HALT FOR DISPLAY
4670  036244  000000                          HALT
4671  036246  000407                          BR     3$              ;;
4672  036250  005000          1$:     CLR    R0
4673  036252  012701  000020                  MOV    #20,R1          ;DELAY BEFORE CLEANING SCREEN
4674  036256  005300          2$:     DEC    R0
4675  036260  001376                          BNE    2$
4676  036262  005301                          DEC    R1
4677  036264  001374                          BNE    2$
4678  036266  104401  050750  3$:     TYPE   ,VTINIT
4679  036272  000207                          RTS    PC
4680                          ;;TYPE RMS AND PEAK VALUES;;
                              TYPRP:  TST    R2              ;IS NOISE POSITIVE?
4681  036274  005702
4682  036276  100001                          BPL    POSNOI          ;YES
4683  036300  005002                          CLR    R2              ;R2<0,SET R2=0
4684  036302  104416          POSNOI: TYPDC
4685  036304  104401  050652          TYPE   ,MLSBAT         ;TYPE " LSB AT "
4686  036310  004737  034066          JSR    PC,TYPEDG
4687  036314  104401  046601  PSONOI: TYPE   ,CHAN           ;TYPE " ON CHANNEL "
4688  036320  013746  001556          MOV    CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
                                                              ;;TYPE CHANL
 (1)  036324  104403                          TYPOS                  ;;GO TYPE--OCTAL ASCII
 (1)  036326     002                          .BYTE  2               ;;TYPE 2 DIGIT(S)
 (1)  036327     000                          .BYTE  0               ;;SUPPRESS LEADING ZEROS
4689  036330  000207                          RTS    PC
```

```
4691                                         ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
4692                                         ;       * ROUTINE DESTROYS R0 *
4693   036332  012500                 CONVRT: MOV      (R5)+,R0               ;GET CHANNEL VALUE
4694   036334  010037  001556                MOV      R0,CHANL
4695   036340  012777  000200  143114 CONVTC: MOV      #200,@ADBUFF           ;LOAD VERNIER DAC
4696   036346  113700  001556         CONVCD: MOVB     CHANL,R0               ;GET CHANNEL
4697   036352  000300                         SWAB     R0                     ;SET UP A/D STATUS REGISTER
4698   036354  052700  000100                 BIS      #100,R0                ;ENABLE INTERRUPTS
4699   036360  010077  143072                 MOV      R0,@STREG
4700   036364  012700  010000                 MOV      #10000,R0              ;DAC SETTLING DELAY
4701   036370  005300                 1$:     DEC      R0
4702   036372  001376                         BNE      1$
4703   036374  005037  001540                 CLR      TEMP
4704   036400  012777  001650  143056         MOV      #RETURN,@VECTOR        ;LOAD VECTOR
4705   036406  012777  000200  143052         MOV      #200,@VECTR1           ;SET UP NEW PSW
4706   036414  012700  000010                 MOV      #10,R0                 ;SET UP COUNTER
4707   036420  005277  143032         2$:     INC      @STREG                 ;START CONVERSION
4708   036424  000001                         WAIT                            ;WAIT FOR CONVERSION
4709   036426  067737  143030  001540         ADD      @ADBUFF,TEMP           ;READ BUFFER
4710   036434  005300                         DEC      R0
4711   036436  001370                         BNE      2$                     ;DO 8 TIMES
4712   036440  006237  001540                 ASR      TEMP                   ;AVERAGE VALUE
4713   036444  006237  001540                 ASR      TEMP
4714   036450  006237  001540                 ASR      TEMP
4715   036454  005537  001540                 ADC      TEMP
4716   036460  000205                         RTS      R5                     ;RETURN
4717
4718                                   ;COMPARE $GDDAT AND $BDDAT;;
4719                                   ;DESTROYS R0,R1
4720   036462  012537  001124         COMPAR: MOV      (R5)+,$GDDAT           ;GET GOOD DATA
4721   036466  013537  001566                 MOV      @(R5)+,SPREAD          ;GET SPREAD
4722   036472  013737  001540  001126         MOV      TEMP,$BDDAT            ;GET BAD(ACTUAL) DATA
4723   036500  013700  001124         COMPRA: MOV      $GDDAT,R0
4724   036504  163700  001126                 SUB      $BDDAT,R0              ;GET DIFFERENCE
4725   036510  100001                         BPL      7$
4726   036512  005400                         NEG      R0
4727   036514  020037  001566         7$:     CMP      R0,SPREAD              ;COMPARE IT TO SPREAD
4728   036520  003001                         BGT      10$                    ;GO TO ERROR PRINTOUT
4729   036522  005725                         TST      (R5)+                  ;BUMP RETURN POINTER AROUND ERROR CALL
4730   036524  000205                 10$:    RTS      R5
```

```
4732                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS ON GOOD AD;;
4733  036526  012500                GCONVT: MOV     (R5)+,R0                   ;GET CHANNEL VALUE
4734  036530  010037  001556                MOV     R0,CHANL
4735  036534  000300                        SWAB    R0
4736  036536  005037  001540                CLR     TEMP
4737  036542  010077  142740                MOV     R0,@GSTREG                 ;LOAD CHANNEL INTO MIX BITS
4738  036546  012700  010000                MOV     #10000,R0
4739  036552  005300                2$:     DEC     R0
4740  036554  001376                        BNE     2$
4741  036556  012777  001650  142726        MOV     #RETURN,@GVECT             ;LOAD VECTOR
4742  036564  012777  000200  142722        MOV     #200,@GVECT1               ;SET UP NEW PRIORITY
4743  036572  012700  000010                MOV     #10,R0                     ;SET UP COUNTER
4744  036576  152777  000101  142702 1$:    BISB    #101,@GSTREG               ;SET INTRPT. EN., START CONV.
4745  036604  000001                        WAIT                              ;WAIT FOR CONVERSION
4746  036606  067737  142676  001540        ADD     @GADBUF,TEMP               ;READ BUFFER
4747  036614  005300                        DEC     R0
4748  036616  001367                        BNE     1$                         ;DO 8 TIMES
4749  036620  006237  001540                ASR     TEMP                       ;AVERAGE VALUE
4750  036624  006237  001540                ASR     TEMP
4751  036630  006237  001540                ASR     TEMP
4752  036634  005537  001540                ADC     TEMP
4753  036640  000205                        RTS     R5                         ;RETURN
4754
4755                                        ;;SUBROUTINE TO CONVERT 2.60 VOLTS TO 15.00 VOLTS;;
4756                                        ;;FUNNY NUMBER CALCULATED BY:
4757                                        ;;    (15*2.56/(VOLTAGE))/0.0025
4758
4759  036642  032703  004000        CONV15: BIT     #BIT11,R3                  ;IS RESULT MINUS?
4760  036646  001003                        BNE     1$                         ;;NO
4761  036650  005403                        NEG     R3                         ;YES, MAKE IT PLUS
4762  036652  104401  042154                TYPE    ,MINUS                     ;TYPE '-'
4763  036656  042703  174000        1$:     BIC     #174000,R3                 ;CLEAR UPPER 5 BITS
4764  036662  005002                        CLR     R2                         ;CLEAR RESULT REGISTER
4765  036664  012701  013424                MOV     #5908.,R1                  ;PUT FUNNY NUMBER INTO R1
4766  036670  012700  002000                MOV     #BIT10,R0                  ;SETUP TEST BIT
4767  036674  030003                2$:     BIT     R0,R3                      ;MULTIPLY TEMP BY FUNNY NUMBER
4768  036676  001401                        BEQ     3$                         ;;
4769  036700  060102                        ADD     R1,R2
4770  036702  006201                3$:     ASR     R1
4771  036704  006200                        ASR     R0
4772  036706  001372                        BNE     2$                         ;;NOT FINISHED YET
4773  036710  006202                        ASR     R2                         ;SCALE TO .01 VOLTS / BIT
4774  036712  006202                        ASR     R2
4775  036714  005502                        ADC     R2
4776  036716  104416                        TYPDC                              ;TYPE RESULTS
4777  036720  104401  044522                TYPE    ,VOLTS                     ;TYPE 'VOLTS'
4778  036724  000207                        RTS     PC
4779
```

```
4781
4782                                    ;SUBROUTINE LOADY;
4783  036726  005702           LOADY:  TST    R2            ;ROUTINE TO LOAD VLAUE INTO R2
4784  036730  100001                   BPL    PLUSR2        ;AS A VT55 Y-VALUE
4785  036732  005002                   CLR    R2
4786  036734  020227  000353   PLUSR2: CMP    R2,#235.
4787  036740  002402                   BLT    LESS
4788  036742  012702  000353           MOV    #235.,R2
4789  036746  010203           LESS:   MOV    R2,R3
4790  036750  042702  177740           BIC    #177740,R2
4791  036754  052702  000040           BIS    #40,R2
4792  036760  105777  142164   B10:    TSTB   @$TPS         ;PRINT CHARACTER
4793  036764  100375                   BPL    B10
4794  036766  110277  142160           MOVB   R2,@$TPB
4795  036772  006203                   ASR    R3
4796  036774  006203                   ASR    R3
4797  036776  006203                   ASR    R3
4798  037000  006203                   ASR    R3
4799  037002  006203                   ASR    R3
4800  037004  042703  177770           BIC    #177770,R3
4801  037010  052703  000040           BIS    #40,R3
4802  037014  105777  142130   B11:    TSTB   @$TPS         ;PRINT CHARACTER
4803  037020  100375                   BPL    B11
4804  037022  110377  142124           MOVB   R3,@$TPB
4805  037026  000207                   RTS    PC
4806
4807                                    ;SUBROUTINE TO DO A BUS RESET
4808  037030  004737  022144   ARESET: JSR    PC,STALL      ;DELAY
4809  037034  000005                   RESET                ;BUS RESET
4810  037036  004737  022144           JSR    PC,STALL      ;DELAY
4811  037042  000207                   RTS    PC            ;EXIT
4812
4813                                    ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
4814                                    ;;IN R2 AS X.XX;;
4815  037044  005702           DECTYP: TST    R2            ;TEST VALUE TO BE TYPED
4816  037046  100003                   BPL    POS
4817  037050  104401  042154           TYPE   ,MINUS             ;TYPE MINUS SIGN
4818  037054  005402                   NEG    R2
4819  037056  020227  023417   POS:    CMP    R2,#9999.          ;>9999. REPLACE IT WITH 9999.
4820  037062  003402                   BLE    OKAYD
4821  037064  012702  023417           MOV    #9999.,R2
4822  037070  105037  053612   OKAYD:  CLRB   ONES          ;CLEAR ONES
4823  037074  105037  053611           CLRB   TENS          ;CLEAR TENS
4824  037100  105037  053607           CLRB   HUNS          ;CLEAR HUNS
4825  037104  105037  053606           CLRB   THOUS         ;CLEAR THOUS
4826  037110  005702           TESTR2: TST    R2                 ;CONVERT VALUE TO A DECIMAL VALUE
4827  037112  001434                   BEQ    TYPOUT
4828  037114  005302                   DEC    R2
4829  037116  105237  053612           INCB   ONES
4830  037122  123727  053612  000012   CMPB   ONES,#10.
4831  037130  001367                   BNE    TESTR2
4832  037132  105037  053612           CLRB   ONES
4833  037136  105237  053611           INCB   TENS
4834  037142  123727  053611  000012   CMPB   TENS,#10.
4835  037150  001357                   BNE    TESTR2
4836  037152  105037  053611           CLRB   TENS
```

```
4837  037156  105237  053607              INCB    HUNS
4838  037162  123727  053607  000012      CMPB    HUNS,#10.
4839  037170  001347                      BNE     TESTR2           ;;
4840  037172  105037  053607              CLRB    HUNS
4841  037176  105237  053606              INCB    THOUS
4842  037202  000742                      BR      TESTR2
4843  037204  152737  000060  053606 TYPOUT: BISB  #60,THOUS                    ;PREPARE FOR TYPOUT
4844  037212  152737  000060  053607      BISB    #60,HUNS
4845  037220  152737  000060  053611      BISB    #60,TENS
4846  037226  152737  000060  053612      BISB    #60,ONES
4847  037234  123727  053606  000060      CMPB    THOUS,#60
4848  037242  001403                      BEQ     1$               ;;
4849  037244  104401  053606              TYPE    ,THOUS
4850  037250  000002                      RTI
4851  037252  104401  053607      1$:     TYPE    ,HUNS            ;TYPE VALUE
4852  037256  000002                      RTI
4853                          ;SUBROUTINE TO SENSE THE 'WFTEST' FLAG AND USE WIDE/NARROW ERROR TOLERANCES
4854  037260  012701  037644      WFADJ:  MOV     #LIMITS,R1       ;SUBROUTINE TO SET LIMITS
4855  037264  005021      3$:     CLR     (R1)+            ;CLEAR THE BUFFER
4856  037266  022701  037746              CMP     #LIMITE,R1       ;TEST IF AT END OF ALL THE BUFFER
4857  037272  001374                      BNE     3$               ;BR IF NOT
4858  037274  012701  037644              MOV     #LIMITS,R1       ;RELOAD BUFFER POINTER
4859  037300  005737  001602              TST     WFTEST           ;RUNNING ON TESTER ?
4860  037304  100403                      BMI     1$               ;;YES
4861  037306  012702  037750              MOV     #VARLT1,R2       ;WFTEST NOT MINUS, USE NORMAL LIMITS
4862  037312  000402                      BR      2$               ;;
4863  037314  012702  040012      1$:     MOV     #VARLT2,R2       ;WFTEST MINUS, USE OPTION AREA LIMITS
4864  037320  012221      2$:     MOV     (R2)+,(R1)+      ;SET UP LIMITS
4865  037322  022701  037706              CMP     #LIMITM,R1       ;TEST IF END OF 1ST PART
4866  037326  001374                      BNE     2$               ;;BR IF NOT
4867  037330  032737  040000  001602 WFADJO: BIT  #BIT14,WFTEST    ;TEST IF INCOMMING TESTS OF MNCTP
4868  037336  001403                      BEQ     1$               ;BR IF NOT
4869  037340  012702  040114              MOV     #VARLT3,R2       ;LOAD INCOMMING MNCTP LIMIT POINTER
4870  037344  000402                      BR      2$
4871  037346  012702  040054      1$:     MOV     #VARLT4,R2       ;LOAD NORMAL MNCTP LIMIT POINTER
4872  037352  012221      2$:     MOV     (R2)+,(R1)+      ;LOAD A VALUE INTO TABLE
4873  037354  022701  037746              CMP     #LIMITE,R1       ;TEST IF END
4874  037360  001374                      BNE     2$               ;BR IF NOT
4875  037362  000207                      RTS     PC
4876
4877  037364  000000              V0:     0                        ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
4878  037366  000002              V2:     2
4879  037370  000012              V10:    10.
4880  037372  000012              V12:    12
4881  037374  000062              V50D:   50.
4882  037376  000144              V100D:  100.
4883  037400  000326              V326:   326
4884  037402  000270              K270:   270
4885  037404  000516              K516:   516
4886  037406  000275              K275:   275
4887  037410  000072              K72:    72
4888  037412  000152              K152:   152
4889  037414  000400              K400:   400
```

```
4891
4892                                    ;LIST OF OFFSET TOLERANCES FOR EACH GAIN/TC TYPE FOR THE MNCTP OFFSET TEST
4893                                    ;FORMAT OF THE EXPECTED VALUE IS XXXX.X AND THE TOLERANCES Y.Y
4894
4895   037416   000060                  TPOF00:  60
4896   037420   000060                  TPOF01:  60
4897   037422   000134                  TPOF02:  134
4898   037424   000134                  TPOF03:  134
4899   037426   000260                  TPOF04:  260
4900   037430   000260                  TPOF05:  260
4901   037432   000272                  TPOF06:  272
4902   037434   000405                  TPOF07:  405
4903   037436   000331                  TPOF10:  331
4904   037440   000331                  TPOF11:  331
4905   037442   001127                  TPOF12:  1127
4906   037444   001120                  TPOF13:  1120
4907   037446   002400                  TPOF14:  2400
4908   037450   002400                  TPOF15:  2400
4909   037452   003156                  TPOF16:  3156
4910   037454   003632                  TPOF17:  3632
4911
4912                                    ;TABLE OF EXPECTED VALUES FOR THE MNCTP TESTS
4913                                    ;HIGH 4 BITS ARE THE GAIN/TP TYPE AND LOW 12 BITS ARE THE EXPECTED VALUE
4914                                    ;   SECOND WORD IS THE ADDRESS OF THE TOLERANCE FOR THAT GAIN/TC TYPE
4915   037456   024302                  TPVALS: 024302        ;CHAN A
4916   037460   037410                          K72           ; CHANNEL A TOLERANCE
4917   037462   035654                          035654        ;CHAN B
4918   037464   037402                          K270          ; CHANNEL B TOLERANCE
4919   037466   044500                          044500        ;CHAN C
4920   037470   037412                          K152          ; CHANNEL C TOLERANCE
4921   037472   117156                          117156        ;CHAN D
4922   037474   037404                          K516          ; CHANNEL D TOLERANCE
4923   037476   110622                          110622        ;CHAN E
4924   037500   037404                          K516          ; CHANNEL E TOLERANCE
4925   037502   053441                          053441        ;CHAN F
4926   037504   037412                          K152          ; CHANNEL F TOLERANCE
4927   037506   012224                          012224        ;CHAN G
4928   037510   037406                          K275          ; CHANNEL G TOLERANCE
4929   037512   163621                          163621        ;CHAN H
4930   037514   037414                          K400          ; CHANNEL H TOLERANCE
4931   037516   000000                          0
4932   037520   000000                          0
```

H 11

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 77
CVMNAC.P11   10-JUL-81 14:30   DETERMINE IF MORE MNCAD'S TO BE TESTED   SEQ 0137

```
4934
4935                                            ;TABLE OF EXPECTED CONVERTED VALUES FOR THE MNCTP OFFSET TEST
4936                                            ;      IN THE FORMAT  4000.0
4937    037522    040000                        TPOFFK: 40000    ;GAIN-TYPE REGISTER = 00
4938    037524    037770                                37770
4939    037526    040130                                40130
4940    037530    040130                                40130
4941    037532    040330                                40330
4942    037534    040330                                40330
4943    037536    040420                                40420
4944    037540    040477                                40477
4945    037542    040000                                40000      ;GAIN-TYPE REGISTER = 10
4946    037544    037770                                37770
4947    037546    040560                                40560
4948    037550    040540                                40540
4949    037552    041510                                41510
4950    037554    041520                                41520
4951    037556    042100                                42100
4952    037560    042370                                42370      ;GAIN-TYPE REGISTER = 17
4953
4954                                            ;TABLE OF TOLERANCES FOR THE ABOVE LISTED CONVERTED VALUES OF THE MNCTP
4955                                            ;      IN THE FORMAT  00.1
4956    037562    037416                        TPOFFS: TPOF00   ;GAIN-TYPE REGISTER = 00
4957    037564    037420                                TPOF01
4958    037566    037422                                TPOF02
4959    037570    037424                                TPOF03
4960    037572    037426                                TPOF04
4961    037574    037430                                TPOF05
4962    037576    037432                                TPOF06
4963    037600    037434                                TPOF07
4964    037602    037436                                TPOF10   ;GAIN-TYPE REGISTER = 10
4965    037604    037440                                TPOF11
4966    037606    037442                                TPOF12
4967    037610    037444                                TPOF13
4968    037612    037446                                TPOF14
4969    037614    037450                                TPOF15
4970    037616    037452                                TPOF16
4971    037620    037454                                TPOF17   ;GAIN-TYPE REGISTER = 17
4972
```

```
4974                                   ;*VOLTAGE TABLE OF EXPECTED VALUES (SINGLE ENDED) <TEST MODULE>
4975   037622   005560          VTABLE: 5560              ;+2.2 VOLTS        <CH10, 20, 30 ETC>
4976   037624   002220                  2220              ;-2.2 VOLTS
4977   037626   004670                  4670              ;+1.1 VOLTS
4978   037630   003110                  3110              ;-1.1 VOLTS
4979   037632   007340                  7340              ;+4.4 VOLTS        <CH14, 24, 34 ETC>
4980   037634   000440                  0440              ;-4.4 VOLTS
4981   037636   006450                  6450              ;+3.3 VOLTS
4982   037640   001330                  1330              ;-3.3 VOLTS        <CH17, 27, 37 ETC>
4983   037642   100000                  BIT15             ;END INDICATOR
4984
4985                                   ;TABLE OF LIMITS FOR THE MNCAD, MNCAM AND MNCAG (LOCATIONS MODIFIED BY PROGRAM)
4986   037644                   LIMITS:
4987   037644   000050          VNR:    40.               ;RMS NOISE TEST LIMITS FOR MNCAD-MNCAM CHANNELS
4988   037646   000310          VNP:    200.              ;PEAK NOISE TEST LIMITS FOR MNCAD-MNCAM CHANNELS
4989   037650   000074          VNRAG0: 60.               ;RMS NOISE TEST LIMIT FOR .5 MNCAG CHANNELS
4990   037652   000257          VNPAG0: 175.              ;PEAK NOISE TEST LIMIT FOR .5 MNCAG CHANNELS
4991   037654   000113          VNRAG1: 75.               ;RMS NOISE TEST LIMIT FOR 5. MNCAG CHANNELS
4992   037656   000341          VNPAG1: 225.              ;PEAK NOISE TEST LIMIT FOR 5. MNCAG CHANNELS
4993   037660   000000          VRAG2A: 0                 ;MSW OF RMS NOISE TEST LIMIT FOR 50. MNCAG CHANNELS
4994   037662   000000          VRAG2B: 0                 ;LSW OF RMS NOISE TEST LIMIT FOR 50. MNCAG CHANNELS
4995   037664   000000          VPAG2A: 0                 ;MSW OF PEAK NOISE TEST LIMIT FOR 50. MNCAG CHANNELS
4996   037666   000000          VPAG2B: 0                 ;LSW OF PEAK NOISE TEST LIMIT FOR 50. MNCAG CHANNELS
4997   037670   000000          VRAG3A: 0                 ;MSW OF RMS NOISE TEST LIMIT FOR 500. MNCAG CHANNELS
4998   037672   000000          VRAG3B: 0                 ;LSW OF RMS NOISE TEST LIMIT FOR 500. MNCAG CHANNELS
4999   037674   000000          VPAG3A: 0                 ;MSW OF PEAK NOISE TEST LIMIT FOR 500. MNCAG CHANNELS
5000   037676   000000          VPAG3B: 0                 ;LSW OF PEAK NOISE TEST LIMIT FOR 500. MNCAG CHANNELS
5001   037700   000003          VCM:    3                 ;COMMON MODE TEST LIMIT FOR MNCAG CHANNELS
5002   037702   000144          VSET:   100.              ;SETTLING TEST LIMIT FOR MNCAD-MNCAM CHANNELS
5003   037704   000175          VLIN:   125.              ;RELATIVE ACCURACY TEST LIMIT
5004   037706                   LIMITM:
5005                                   ;LIMITS FOR THE MNCTP MODULE (LOCATIONS MODIFIED BY PROGRAM)
5006   037706   000000          VRTP0A: 0        ;.67 LSB          ;MSW OF RMS VALUE FOR TP TYPE = 00
5007   037710   124000          VRTP0B: 124000                     ;LSW OF RMS VALUE FOR TP TYPE = 00
5008   037712   000002          VPTP0A: 2        ;2                ;MSW OF PEAK VALUE FOR TP TYPE = 00
5009   037714   000000          VPTP0B: 0                          ;LSW OF PEAK VALUE FOR TP TYPE = 00
5010   037716   000000          VRTP1A: 0        ;.83 LSB          ;MSW OF RMS VALUE FOR TP TYPE = 01
5011   037720   151000          VRTP1B: 151000                     ;LSW OF RMS VALUE FOR TP TYPE = 01
5012   037722   000002          VPTP1A: 2        ;2.5              ;MSW OF PEAK VALUE FOR TP TYPE = 01
5013   037724   100000          VPTP1B: 100000                     ;LSW OF PEAK VALUE FOR TP TYPE = 01
5014   037726   000001          VRTP2A: 1        ;1.0              ;MSW OF RMS VALUE FOR TP TYPE = 10
5015   037730   000000          VRTP2B: 0                          ;LSW OF RMS VALUE FOR TP TYPE = 10
5016   037732   000003          VPTP2A: 3        ;3.0              ;MSW OF PEAK VALUE FOR TP TYPE = 10
5017   037734   000000          VPTP2B: 0                          ;LSW OF PEAK VALUE FOR TP TYPE = 10
5018   037736   000001          VRTP3A: 1        ;1.3              ;MSW OF RMS VALUE FOR TP TYPE = 11
5019   037740   051000          VRTP3B: 051000                     ;LSW OF RMS VALUE FOR TP TYPE = 11
5020   037742   000004          VPTP3A: 4        ;4.0              ;MSW OF PEAK VALUE FOR TP TYPE = 11
5021   037744   000000          VPTP3B: 0                          ;LSW OF PEAK VALUE FOR TP TYPE = 11
5022   037746   000000          LIMITE: 0
```

```
5024                        ;LIMITS FOR MNCAD, MNCAM AND MNCAG WITHOUT THE TESTER
5025  037750  000050        VARLT1: 40.             ;.4 LSB, NORMAL LIMITS FOR SYSTEM
5026  037752  000310                200.            ;2 LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
5027  037754  000074                60.     ;0.60   ;RMS VALUE FOR .5
5028  037756  000257                175.    ;1.75   ;PEAK VALUE FOR .5
5029  037760  000113                75.     ;0.75   ;RMS VALUE FOR 5.
5030  037762  000341                225.    ;2.25   ;PEAK VALUE FOR 5.
5031  037764  000001                1       ;1.33   ;MSW OF RMS VALUE FOR 50.
5032  037766  052173                052173          ;LSW OF RMS VALUE FOR 50.
5033  037770  000004                4       ;4.00   ;MSW OF PEAK VALUE FOR 50.
5034  037772  000000                0               ;LSW OF PEAK VALUE FOR 50.
5035  037774  000004                4       ;4.00   ;MSW OF RMS VALUE FOR 500.
5036  037776  000000                0               ;LSW OF RMS VALUE FOR 500.
5037  040000  000014                12.     ;12.00  ;MSW OF PEAK VALUE FOR 500.
5038  040002  000000                0               ;LSW OF PEAK VALUE FOR 500.
5039  040004  000004                4       ;0.04   ;COMMON MODE VALUE
5040  040006  000144                100.            ;1 LSB
5041  040010  000175                125.            ;1.25 LSB
5042
5043                        ;LIMITS FOR MNCAD, MNACM AND MNCAG WITH THE TESTER
5044  040012  000041        VARLT2: 33.             ;.33 LSB RMS NOISE LIMIT
5045  040014  000226                150.            ;1.5 LSB PEAK NOISE LIMIT
5046  040016  000062                50.     ;0.50   ;RMS VALUE FOR .5
5047  040020  000226                150.    ;1.50   ;PEAK VALUE FOR .5
5048  040022  000074                60.     ;0.60   ;RMS VALUE FOR 5.
5049  040024  000257                175.    ;1.75   ;PEAK VALUE FOR 5.
5050  040026  000001                1       ;1.15   ;MSW OF RMS VALUE FOR 50.
5051  040030  023146                023146          ;LSW OF RMS VALUE FOR 50.
5052  040032  000003                3       ;3.50   ;MSW OF PEAK VALUE FOR 50.
5053  040034  100000                100000          ;LSW OF PEAK VALUE FOR 50.
5054  040036  000003                3       ;3.66   ;MSW OF RMS VALUE FOR 500.
5055  040040  124366                124366          ;LSW OF RMS VALUE FOR 500.
5056  040042  000013                11.     ;11.0   ;MSW OF PEAK VALUE FOR 500.
5057  040044  000000                0               ;LSW OF PEAK VALUE FOR 500.
5058  040046  000003                3       ;0.03   ;COMMON MODE VALUE
5059  040050  000132                90.             ;.9 LSB INTER-CHANNEL SETTLING LIMIT
5060  040052  000144                100.            ;1 LSB RELATIVE ACCURACY ERROR LIMIT
```

```
5062
5063                                  ;LIMITS FOR THE MNCTP NORMAL TEST
5064   040054   000001               VARLT4: 1        ;1.67    ;MSW OF RMS VALUE FOR TP TYPE = 00
5065   040056   124000                       124000            ;LSW OF RMS VALUE FOR TP TYPE = 00
5066   040060   000003                       3        ;3.5     ;MSW OF PEAK VALUE FOR TP TYPE = 00
5067   040062   100000                       100000            ;LSW OF PEAK VALUE FOR TP TYPE = 00
5068   040064   000002                       2        ;2.5     ;MSW OF RMS VALUE FOR TP TYPE = 01
5069   040066   100000                       100000            ;LSW OF RMS VALUE FOR TP TYPE = 01
5070   040070   000004                       4        ;4.5     ;MSW OF PEAK VALUE FOR TP TYPE = 01
5071   040072   100000                       100000            ;LSW OF PEAK VALUE FOR TP TYPE = 01
5072   040074   000003                       3        ;3.      ;MSW OF RMS VALUE FOR TP TYPE = 10
5073   040076   000000                       0                 ;LSW OF RMS VALUE FOR TP TYPE = 10
5074   040100   000005                       5        ;5.5     ;MSW OF PEAK VALUE FOR TP TYPE = 10
5075   040102   100000                       100000            ;LSW OF PEAK VALUE FOR TP TYPE = 10
5076   040104   000003                       3        ;3.5     ;MSW OF RMS VALUE FOR TP TYPE = 11
5077   040106   100000                       100000            ;LSW OF RMS VALUE FOR TP TYPE = 11
5078   040110   000006                       6        ;6.5     ;MSW OF PEAK VALUE FOR TP TYPE = 11
5079   040112   100000                       100000            ;LSW OF PEAK VALUE FOR TP TYPE = 11
5080
5081                                  ;LIMITS FOR INCOMMING TESTING OF MNCTP
5082   040114   000000               VARLT3: 0        ;.67     ;MSW OF RMS VALUE FOR TP TYPE = 00
5083   040116   124000                       124000            ;LSW OF RMS VALUE FOR TP TYPE = 00
5084   040120   000002                       2        ;2.LSB   ;MSW OF PEAK VALUE FOR TP TYPE = 00
5085   040122   000000                       0                 ;LSW OF PEAK VALUE FOR TP TYPE = 00
5086   040124   000000                       0        ;.83     ;MSW OF RMS VALUE FOR TP TYPE = 01
5087   040126   151000                       151000            ;LSW OF RMS VALUS FOR TP TYPE = 01
5088   040130   000002                       2        ;2.5     ;MSW OF PEAK VALUE FOR TP TYPE = 01
5089   040132   100000                       100000            ;LSW OF PEAK VALUE FOR TP TYPE = 01
5090   040134   000001                       1        ;1.LSB   ;MSW OF RMS VALUE FOR TP TYPE = 10
5091   040136   000000                       0                 ;LSW OF RMS VALUE FOR TP TYPE = 10
5092   040140   000003                       3        ;3.LSB   ;MSW OF PEAK VALUE FOR TP TYPE = 10
5093   040142   000000                       0                 ;LSW OF PEAK VALUE FOR TP TYPE = 10
5094   040144   000001                       1        ;1.33    ;MSW OF RMS VALUE FOR TP TYPE = 11
5095   040146   051000                       051000            ;LSW OF RMS VALUE FOR TP TYPE = 11
5096   040150   000004                       4        ;4.LSB   ;MSW OF PEAK VALUE FOR TP TYPE = 11
5097   040152   000000                       0                 ;LSW OF PEAK VALUE FOR TP TYPE = 11
5098
```

```
5100                          .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)   040154                $EOP:
  (2)   040154   000240               NOP
  (1)   040156   005037   001102      CLR     $TSTNM         ;;ZERO THE TEST NUMBER
  (1)   040162   005037   001160      CLR     $TIMES         ;;ZERO THE NUMBER OF ITERATIONS
  (1)   040166   005237   001176      INC     $PASS          ;;INCREMENT THE PASS NUMBER
  (1)   040172   042737   100000  001176  BIC #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
  (1)   040200   005327               DEC     (PC)+          ;;LOOP?
  (1)   040202   000001        $EOPCT: .WORD   1
  (1)   040204   003022               BGT     $DOAGN         ;;YES
  (1)   040206   012737               MOV     (PC)+,@(PC)+   ;;RESTORE COUNTER
  (1)   040210   000001        $ENDCT: .WORD   1
  (1)   040212   040202               $EOPCT
  (1)   040214   104401   040261      TYPE    ,$ENDMG        ;;TYPE 'END PASS #''
  (2)   040220   013746   001176      MOV     $PASS,-(SP)    ;;SAVE $PASS FOR TYPEOUT
  (2)   040224   104405               TYPDS                  ;;GO TYPE--DECIMAL ASCII WITH SIGN
  (1)   040226   104401   040256      TYPE    ,$ENULL        ;;TYPE A NULL CHARACTER
  (1)   040232   013700   000042      $GET42: MOV     @#42,R0        ;;GET MONITOR ADDRESS
  (1)   040236   001405               BEQ     $DOAGN         ;;BRANCH IF NO MONITOR
  (1)   040240   000005               RESET                  ;;CLEAR THE WORLD
  (1)   040242   004710        $ENDAD: JSR     PC,(R0)        ;;GO TO MONITOR
  (1)   040244   000240               NOP                    ;;SAVE ROOM
  (1)   040246   000240               NOP                    ;;FOR
  (1)   040250   000240               NOP                    ;;ACT11
  (1)   040252                $DOAGN:
  (1)   040252   000137               JMP     @(PC)+         ;;RETURN
  (1)   040254   040276        $RTNAD: .WORD   EXTMSG
  (1)   040256   377     377    000  $ENULL: .BYTE   -1,-1,0        ;;NULL CHARACTER STRING
  (1)   040261   015   042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
  (1)   040266   050040   051501  020123
  (1)   040274   000043
5101   040276   052777   000100  140640  EXTMSG: BIS     #BIT6,@$TKS    ;ENABLE KRB INTR.
5102   040304   005737   001112      TST     $ERTTL         ;ANY ERRORS
5103   040310   001415               BEQ     1$             ;BR IF NOT
5104   040312   104401   050555      TYPE    ,ERRTOT        ;TYPE TOTAL ERROR COUNT PRIMER
5105   040316   013746   001112      MOV     $ERTTL,-(SP)   ;GET VALUE
5106   040322   104405               TYPDS                  ;REPORT IT
5107   040324   005737   001552      TST     NMBEXT         ;TEST IF MULTIPLE
5108   040330   001405               BEQ     1$             ;BR IF NOT
5109   040332   104401   050604      TYPE    ,MESGD         ;TYPE BAD UNIT PRIMER
5110   040336   013746   001616      MOV     BADUNT,-(SP)
5111   040342   104406               TYPBN                  ;REPORT 1 + 0'S
5112   040344   104401   001165  1$: TYPE,    $CRLF         ;ENSURE FRESH LINE
5113   040350   000137               JMP     @(PC)+         ;RETURN
5114   040352   001666        AGTST:  BEGIN
```

```
5116                                    ;*
5117                                    ;*THIS ROUTINE WILL PROTECT THE PROGRAM
5118                                    ;*FROM INTERUPTS (BAD ONES).
5119                                    ;*
5120                                    ;*THE TRAP CATCHER IS SET UP FOR
5121                                    ;*      .WORD    .+2
5122                                    ;*       JSR PC,R0
5123                                    ;*
5124                                    ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
5125                                    ;*GOTO THE VECTOR AND PCITK UP THE ''.+2'' AS AN ADDRESS
5126                                    ;*AND ''4700'' AS NEW STATUS.
5127                                    ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE ''JSR PC,R0'' (AN ILLEGAL INSTR.).
5128                                    ;*AND TRAP TO LOCATION ''4''. IN LOCATION 4 WE HAVE A
5129                                    ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4.
5130                                    ;*WE WILL REPORT IT IN THE SAME MANNER THAT WER WOULD
5131                                    ;*REPORT ANY OTHER ERROR.
5132                                    ;*IF A BUSS ERROR TRAP DID OCCUT AND CAUSE A TRAP TO 4.
5133                                    ;*WE WILL HALT.
5134
5135
5136  040354  011637  040634     IOTRD: MOV    (SP),TRTO       ;GET WHERE WE CAME TO.
5137  040360  162737  000004 040634    SUB    #4,TRTO         ;FORM READ ADDR.
5138  040366  023727  040634 001000    CMP    TRTO,#1000      ;DID TRAP FROM LESS THAN ADDR. 1000?
5139  040374  003402               BLE    2$              ;NO-CONTINUE.
5140  040376  000000          1$:   HALT                   ;A BUSS ERROR TIME OUT TRAP BROUGHT US HERE.
5141                                               ;ADDRESS CONTAINED IN TRTO.
5142  040400  000776               BR     1$              ;DON'T ALLOW CONTINUE.
5143  040402  016637  000004 040636 2$: MOV    4(SP),TRFRO     ;GET TRAPPED FROM ADDR.
5144  040410  122737  000021 001102    CMPB   #21,$TSTNM      ;LESS THAN INTERRUPT TESTS?
5145  040416  003402               BLE    3$              ;NO MUST BE WRONG VECTOR.
5146                        ;//////////////////////////////////////////////////////////
5147  040420  104003               ERROR  3               ;ERROR! ILLEGAL INTERRUPT OR
5148                                               ;INTERRUPT TO WRONG VECTOR.
5149                                               ;IF TEST NO. IS LESS THAN 10,ITS
5150                                               ;LIKELY(BUT NO EXCLUSIVELY)TO BE A
5151                                               ;DEVICE OTHER THAN THE DEVICE UNDER TEST.
5152                                               ;IF THE INTERRUPT OCCURED
5153                                               ;DURING AN INTERRUPT TEST, I'D
5154                                               ;SUSPECT A PROBLEM WITH THE DEVICE UNDER TEST.
5155                                               ;IF THE ADDRESS THE INTERRUPT
5156                                               ;VECTORED TO IS WITHIN THE RANGE OF
5157                                               ;VECTORS ASSIGNED TO THE DEVICE.
5158                                               ;THEN I'D SUSPECT THE DEVICE
5159                                               ;INTERRUPTD ILLEGALLY.
5160                                               ;IF THE ADDRESS THE INTERRUPT
5161                                               ;VECTORED TO IS OUTSIDE OF THE
5162                                               ;RANGE ASSIGNED TO THE DEVICE
5163                                               ;I'D SUSPECT THAT THE
5164                                               ;DEVICE PUT THE WRONG INTERRUPT
5165                                               ;VECTOR ON THE BUS DURING THE INTERRUPT
5166                                               ;PROCESS.
5167                                               ;     NOTE:
5168                                               ;FOR THIS ERROR - DON'T USE
5169                                               ; ''LOOP ON ERROR'' OPTION.
5170                                               ;ALSO EXPECT THAT THE INTERRUPT TEST TO
5171                                               ;WILL REPOT THAT THE DEVICE DIDN'T
```

```
5172                                                       ;INTERRUPT.
5173                                                       ;FOLLOW THE RECOMMENDED PROCEEDURE
5174                                                       ;  IN THE DOCUMENT (ON THIS DIAGNOSTIC)
5175                                                       ;FOR LOOPING ON TEST.
5176
5177                              ;//////////////////////////////////////////////////////////////////
                                      RTI
5178   040422  000002
5179   040424  022626         3$:    CMP      (SP)+,(SP)+       ;POP OFF JSR TRAP
5180   040426  022626                CMP      (SP)+,(SP)+       ;POP OFF WRONG INTR.
5181   040430  005737  001176        TST      $PASS            ;IS THIS THE FIRST PASS?
5182   040434  001025                BNE      4$               ;;NO, DON'T REPORT
5183   040436  104401  043454        TYPE     ,VTMSG           ;TYPE 'EXPECTED INTR. AT ''
5184   040442  004737  055712        JSR      PC,WHICHU        ;DETERMINE THE UNIT #
5185   040446  013746  001202        MOV      $UNIT,-(SP)
5186   040452  104405                TYPDS
5187   040454  104401  043500        TYPE     ,VTMSG3          ;REPORT INTR. TO
5188   040460  013746  001464        MOV      VECTOR,-(SP)     ;;SAVE VECTOR FOR TYPEOUT
(1)    040464  104403                TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)    040466     003                .BYTE    3                ;;TYPE 3 DIGIT(S)
(1)    040467     001                .BYTE    1                ;;TYPE LEADING ZEROS
5189   040470  104401  043531        TYPE     ,VTMSG1          ;TYPE '' RECEIVED INTR. AT ''
5190   040474  013746  040634        MOV      TRTO,-(SP)       ;;SAVE TRTO FOR TYPEOUT
(1)    040500  104403                TYPOS                     ;;GO TYPE--OCTAL ASCII
(1)    040502     003                .BYTE    3                ;;TYPE 3 DIGIT(S)
(1)    040503     001                .BYTE    1                ;;TYPE LEADING ZEROS
5191   040504  104401  043561        TYPE     ,VTMSG2          ;TYPE 'RESTARTING TEST''
5192   040510  013777  001466  140746  4$:   MOV      VECTR1,@VECTOR
5193   040516  013777  001472  140744        MOV      VECTR3,@VECTR2
5194   040524  012777  004700  140734        MOV      #4700,@VECTR1
5195   040532  012777  004700  140732        MOV      #4700,@VECTR3
5196   040540  013737  040634  001464        MOV      TRTO,VECTOR
5197   040546  042737  000003  001464        BIC      #3,VECTOR
5198   040554  013737  001464  001466        MOV      VECTOR,VECTR1
5199   040562  062737  000002  001466        ADD      #2,VECTR1
5200   040570  013737  001464  001470        MOV      VECTOR,VECTR2
5201   040576  062737  000004  001470        ADD      #4,VECTR2
5202   040604  013737  001470  001472        MOV      VECTR2,VECTR3
5203   040612  062737  000002  001472        ADD      #2,VECTR3
5204   040620  005077  140632                CLR      @STREG
5205   040624  005777  140632                TST      @ADBUFF          ;READ A/D BUFFER TO CLEAR DONE FLAG
5206   040630  000177  140252                JMP      @$LPADR          ;START TEST OVER AGAIN.
5207   040634  000000         TRTO:  .WORD    0                ;CONTAINS ADDR. WE TRAPPED OR INTERRUPTED TO.
5208   040636  000000         TRFRO: .WORD    0                ;CONTAINS ADDR. WE TRAPPED OR INTR. FROM.
```

```
5210                                     .SBTTL        ASCII MESSAGES
5211                                          .NLIST  BEX
5212   040640  047516  047040  044517  NONOIS: .ASCIZ  /NO NOISE DETECTED /
5213   040663     200  043200  052101  ERIEE:  .ASCII  <200><200>/FATAL IEEE BUS ERROR DETECTED/
5214   040722  047600  042520  040522          .ASCII  <200>/OPERATION ABORTED/<200>
5215   040746  047200  020117  047115  NOTSIE: .ASCIZ  <200>\NO MNCTP TESTER OR IEEE INTERFACE DETECTED\<200>
5216   041023     200  047516  044440  NOLSTN: .ASCII  <200>\NO IEEE DVM DETECTED - IS DVM POWER ON ?\<200>
5217   041075     040  020040  044440          .ASCIZ  \     IS DVM CABLE CONNECTED ? - IS DVM AT ADDRESS #2 ?\<200>
5218   041164  051600  040524  052122  SCHAN:  .ASCIZ  <200>\STARTING ON CHANNEL (8) = \
5219   041220  042600  042116  047111  ECHAN:  .ASCIZ  <200>\ENDING ON CHANNEL (8) = \
5220   041252  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST ON UNIT # /
5221   041302  005015  042523  052124  SETMSG: .ASCIZ  <15><12>/SETTLING TEST ON MNCAD UNIT # /
5222   041343     200  042523  052124  SETTPM: .ASCIZ  <200>/SETTLING TEST ON MNCTP/<200>
5223   041374  046600  041516  042101  OFSET:  .ASCIZ  <200>/MNCAD OFFSET TEST ON UNIT # /
5224   041432  046600  041516  050124  SETPMS: .ASCIZ  <200>/MNCTP OFFSET TEST /<200>
5225   041457     111  020123  044124  DWRFAD: .ASCIZ  \IS THE MNCAD (A/D) TEST MODULE CONNECTED ? \
5226   041533     111  020123  020101  DWRFAM: .ASCIZ  \IS A MNCAM (MUX) TEST MODULE CONNECTED ? \
5227   041605     111  020123  020101  DWRFAG: .ASCIZ  \IS A MNCAG (PREAMP) TEST MODULE CONNECTED ? \
5228   041662  051511  040440  046440  DWRFTP: .ASCIZ  \IS A MNCTP (TC AMP) TEST MODULE CONNECTED ? \
5229   041737     111  020123  044124  ASKAMB: .ASCIZ  \IS THE MNCTP (TC AMP) AT AMBIENT (25C) TEMPERATURE ? \
5230   042025     040  046440  041516  NOTPOF: .ASCIZ  \  MNCTP OFFSET TEST WILL NOT BE EXECUTED  \<200>
5231   042101     111  020123  044124  DWRMAP: .ASCIZ  \IS THE CONSOLE TERMINAL A VT55 OR VT105 ? \
5232   042154     055     000          MINUS:  .BYTE   55,0
5233   042156     040     077     040  QUEST:  .BYTE   40,77,40,0
5234   042162  026440  000040          MDASH:  .ASCIZ  / - /
5235   042166  052600  045516  047516  IDONTK: .ASCIZ  <200>\UNKNOWN TYPE OF CHANNEL DETECTED - CHECK MNCAG FRONT PANEL SWITCHE
5236   042274  041600  042510  045503  WOWAGS: .ASCIZ  <200>\CHECK SYSTEM CONFIGURATION - TOO MANY MNCAG DETECTED\<200>
5237   042363     200  043111  046440  RMPTXT: .ASCII  <200>\IF MNCAG CHANNEL - SET MNCAG-TA SWITCH #1, 2, 3 AND 4 TO POSITION
5238   042467     200  020040  047101          .ASCII  <200>\   AND FRONT PANEL SWITCHES TO 'V' AND '100/10' POSITIONS\
5239   042560  044600  020106  047516          .ASCII  <200>\IF NOT, ENSURE SELECTED CHANNELS HAVE THE TEST RAMP CONNECTED\<200
5240   042660  046200  042105  051440  LEDON:  .ASCIZ  <200>\LED SHOULD BE 'ON'\
5241   042704  046200  042105  051440  LEDOFF: .ASCIZ  <200>\LED SHOULD BE 'OFF'\
5242   042731     200  046120  040505  PUSHAG: .ASCII  <200>\PLEASE DEPRESS MNCAG-TA SWITCH #\
5243   042772     065     000          AGTASW: .BYTE   65,0
5244   042774  051511  040440  046440  SCLOCK: .ASCIZ  \IS A MNCKW (CLOCK) IN THE SYSTEM ? \
5245   043040  051440  047111  046107  MSE:    .ASCIZ  / SINGLE ENDED/<15><12>
5246   043060  042600  043111  042506  MDIF:   .ASCIZ  / DIFFERENTIAL/<15><12>
5247   043100  050040  042522  046501  MPRMP:  .ASCIZ  / PREAMP/<15><12>
5248   043112  052040  020103  046501  MTCMP:  .ASCIZ  / TC AMP/<15><12>
5249   043124  042504  044523  042522  GCHAN:  .ASCIZ  \DESIRED 'GAIN OR TC TYPE' REGISTER VALUE TO BE LOADED (0-17) ? \
5250   043224  052600  044523  043516  RW1:    .ASCIZ  <200>\USING CHANNEL # ? \
5251   043250  042600  050130  041505  RW3:    .ASCIZ  <200>\EXPECTED 'GAIN OR TC TYPE' REGISTER VALUE TO BE READ BACK (0-17) ?
5252   043355     200  047115  040503  MADR:   .ASCIZ  <200>\MNCAD (A/D) BASE ADDRESS <\
5253   043411     200  047115  040503  MVCT:   .ASCIZ  <200>\MNCAD (A/D) VECTOR ADDRESS <\
5254   043447     076  037440  000040  ENCOM:  .ASCIZ  #> ? #
5255   043454  046600  041516  042101  VTMSG:  .ASCIZ  <200>\MNCAD (A/D) UNIT #\
5256   043500  005015  054105  042520  VTMSG3: .ASCIZ  <15><12>/EXPECTED INTERRUPT AT /
5257   043531     040  042522  042503  VTMSG1: .ASCIZ  / RECEIVED INTERRUPT AT /
5258   043561     200  046120  040505  VTMSG2: .ASCII  <200>/PLEASE CHECK VECTOR SWITCHES/
5259   043616  005015  051011  051505          .ASCIZ  <15><12>/        RESTARTING LOGIC TEST/<15><12>
5260   043651     015  052012  051505  TCHAN:  .ASCIZ  <15><12>/TEST CHANNELS /
5261   043672  054524  042520  054440  YESNO:  .ASCIZ  /TYPE Y FOR YES, N FOR NO/<15><12>
5262   043725     015  052012  051505  TSTAD:  .ASCIZ  <15><12>/TESTING MNCAD/<15><12>
5263   043747     015  052012  051505  TSTADM: .ASCIZ  <15><12>/TESTING MNCAM/<15><12>
5264   043771     200  042524  052123  TSTAG:  .ASCIZ  <200>/TESTING MNCAG/<200>
5265   044011     123  052105  046440  SADTST: .ASCIZ  #SET MNCAD (A/D) FRONT PANEL SWITCHES TO 'TEST'#<15><12>
```

```
5266  044072  042523  020124  046101  SAGTST: .ASCIZ  #SET ALL MNCAG (PREAMP) RANGE SWITCHES TO THE 'P'' POSITION#<200>
5267  044165     015  051412  052105  SDSE:   .ASCIZ  <15><12>\SET MNCAD-TA SWITCH TO SINGLE ENDED\<15><12>
5268  044235     015  051412  052105  SDDIF:  .ASCIZ  <15><12>\SET MNCAD-TA SWITCH TO DIFFERENTIAL\<15><12>
5269  044305     200  042523  020124  SDMSE:  .ASCIZ  <200>\SET MNCAM-TA SWITCH TO SINGLE ENDED\<200>
5270  044353     200  042523  020124  SDMDIF: .ASCIZ  <200>\SET MNCAM-TA SWITCH TO DIFFERENTIAL\<200>
5271  044421     015  050012  042522  EXTST:  .ASCIZ  <15><12>\PRESS EXTERNAL START ON MNCAD-TA (A/D) ON UNIT #\
5272  044504  005015  030453  036465  TP15:   .ASCIZ  <15><12>/+15=/
5273  044513     015  026412  032461  TM15:   .ASCIZ  <15><12>/-15=/
5274  044522  053040  046117  051524  VOLTS:  .ASCIZ  / VOLTS/
5275  044531     123  052105  046440  SCM:    .ASCIZ  /SET MNCAG (PREAMP) MODE SWITCH TO 'MA'', /
5276  044602  042523  020124  047115  SRM:    .ASCIZ  /SET MNCAG (PREAMP) MODE SWITCH TO 'K'', /
5277  044652  042523  020124  047115  SVM:    .ASCIZ  /SET MNCAG (PREAMP) MODE SWITCH TO 'V'', /
5278  044722  047600  020116  044103  CHAPOS: .ASCIZ  <200>/ON CHANNEL 'A'' - /
5279  044745     200  047117  041440  CHBPOS: .ASCIZ  <200>/ON CHANNEL 'B'' - /
5280  044770  047600  020116  044103  CHCPOS: .ASCIZ  <200>/ON CHANNEL 'C'' - /
5281  045013     200  047117  041440  CHDPOS: .ASCIZ  <200>/ON CHANNEL 'D'' - /
5282  045036  051600  052105  040440  TXTP2:  .ASCIZ  <200>/SET ALL (PREAMP) TEST MODULE CHANNEL SWITCHES TO POSITION 2/<200>
5283  045134  051600  052105  040440  TXTP3:  .ASCIZ  <200>/SET ALL (PREAMP) TEST MODULE CHANNEL SWITCHES TO POSITION 3/<200>
5284  045232  040507  047111  052040  GHLF:   .ASCIZ  \GAIN TO 100/10\<15><12>
5285  045253     123  052105  043440  GAIN5:  .ASCIZ  \SET GAIN TO 10/1\<15><12>
5286  045276  042523  020124  040507  GAIN50: .ASCIZ  \SET GAIN TO 1/.1\<15><12>
5287  045321     123  052105  043440  GAIN5M: .ASCIZ  \SET GAIN TO .1/.01\<15><12>
5288  045346  052600  044523  043516  GANP5:  .ASCIZ  <200>/USING A MNCAG GAIN OF .5/<200>
5289  045401     200  051525  047111  GAN5P:  .ASCIZ  <200>/USING A MNCAG GAIN OF 5./<200>
5290  045434  052600  044523  043516  GAN5D:  .ASCIZ  <200>/USING A MNCAG GAIN OF 50./<200>
5291  045470  052600  044523  043516  GAN5T:  .ASCIZ  <200>/USING A MNCAG GAIN OF 500./<200>
5292  045525     200  051525  047111  TPGN00: .ASCIZ  <200>/USING A MNCTP GAIN-TYPE = 00/<200>
5293  045564  052600  044523  043516  TPGN01: .ASCIZ  <200>/USING A MNCTP GAIN-TYPE = 01/<200>
5294  045623     200  051525  047111  TPGN10: .ASCIZ  <200>/USING A MNCTP GAIN-TYPE = 10/<200>
5295  045662  052600  044523  043516  TPGN11: .ASCIZ  <200>/USING A MNCTP GAIN-TYPE = 11/<200>
5296  045721     200  042523  020124  TPSWOF: .ASCIZ  <200>\SET MNCTP-TA SWITCHES TO THE 'OFF'' OR 'OPEN'' POSITION\<200>
5297  046011     200  042523  020124  TPSWON: .ASCIZ  <200>\SET MNCTP-TA SWITCHES TO THE 'ON'' OR 'SHORTED'' POSITION\<200>
5298  046103     015     012          EXCNOI: .BYTE   15,12
5299  046105     105  041530  051505          .ASCIZ  \EXCESSIVE NOISE ON CHANNEL CAUSED AN\
5300  046152     015     012          ERDIV:  .BYTE   15,12
5301  046154  051101  052111  046510          .ASCIZ  /ARITHMETIC ERROR IN DIVISION - PC= /
5302  046220     015     012          ERMUL:  .BYTE   15,12
5303  046222  051101  052111  046510          .ASCIZ  /ARITHMETIC ERROR IN MULTIPLICATION - PC= /
5304  046274     015     012          EROVF:  .BYTE   15,12
5305  046276  051101  052111  046510          .ASCIZ  /ARITHMETIC OVERFLOW ERROR - PC= /
5306  046337     015     012          ERSQR:  .BYTE   15,12
5307  046341     101  044522  044124          .ASCIZ  /ARITHMETIC ERROR IN SQUARE A 32 BIT NUMBER - PC= /
5308  046423     200  042524  052123  ENDTST: .ASCIZ  <200>/TEST COMPLETED/<200>
5309  046444  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
5310  046453     055  020055     000  DASH:   .ASCIZ  /-- /
5311  046457     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
5312  046477     103  000110          CH:     .ASCIZ  /CH/
5313  046502  020040  020040     000  SPACE:  .ASCIZ  /  /
5314  046507     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
5315  046522  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
5316  046544  040440  020124     000  ATMSG:  .ASCIZ  / AT /
5317  046551     122  051515  020040  RMSNOI: .ASCIZ  /RMS  NOISE /
5318  046565     120  040505  020113  PKNOI:  .ASCIZ  /PEAK NOISE /
5319  046601     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
5320  046616  000057                  SLASH:  .ASCIZ  #/#
5321  046620  020040  020040  045517  OKMSG:  .ASCIZ  /    OK/<15><12>
```

```
5322  046631    015   052012  050131   CCHAN:   .ASCIZ  <15><12>/TYPE IN OCTAL CHANNEL NUMBER AND DEPRESS 'RETURN': /
5323  046717    015   052012  050131   SEL:     .ASCIZ  <15><12>/TYPE 'O' FOR OFFSET, 'G' FOR GAIN & DEPRESS 'RETURN': /
5324  047010  005015  042101  052512   XADJ:    .ASCII  <15><12>/ADJUST R83/
5325  047024  043040  051117  030040   MOLSB:   .ASCII  / FOR 0.00 LSB ERROR/
5326  047047    015   042012  050105            .ASCII  <15><12>/DEPRESS 'RETURN' WHEN ADJUSTED/<15><12>
5327  047112  005015  047111  052520   IGND:    .ASCII  <15><12>/INPUT A GROUND ON THE CHANNEL/ ;MUST BE JUST BEFORE 'CRWR'
5328  047151    015   042012  050105   CRWR:    .ASCIZ  <15><12>/DEPRESS 'RETURN' WHEN READY/<15><12>
5329  047211    015   044412  050116   IVOLT:   .ASCIZ  <15><12>/INPUT +5.115 VOLTS ON THE CHANNEL/
5330  047255    015   040412  045104   YADJ:    .ASCIZ  <15><12>/ADJUST R84/
5331  047272  000053                   POSITV:  .ASCIZ  /+/
5332  047274  051600  046105  041505   SELGT:   .ASCIZ  <200>\SELECT A MNCTP GAIN/TYPE CODE ? \
5333  047336  052200  050131  020105   SELGTA:  .ASCII  <200>\TYPE 0 FOR CODE = 0000\
5334  047365    200   054524  042520            .ASCII  <200>\TYPE 1 FOR CODE = 0001\
5335  047414  052200  050131  020105            .ASCII  <200>\TYPE 2 FOR CODE = 1000\
5336  047443    200   054524  042520            .ASCII  <200>\TYPE 3 FOR CODE = 1001\
5337  047472  047200  046525  042502            .ASCIZ  <200>\NUMBER = \
5338  047505    040   025052  051105   ERMSG:   .ASCIZ  / **ERROR**/<15><12>
5339  047522  051440  044513  050120   SKPMSG:  .ASCIZ  / SKIPPED STATE(S)/
5340  047544  041600  046517  047515   COMOD1:  .ASCIZ  <200>/COMMON MODE REJECTION TEST /
5341  047601    040   040516  051122   NARMSG:  .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
5342  047640  053440  042111  020105   WIDMSG:  .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
5343  047677    040   052123  052101   OUTMSG:  .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
5344  047732  051440  040524  042524   HAFMSG:  .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
5345  050001    200   051120  043517   FOUND1:  .ASCIZ  <200>/PROGRAM DETECTED /
5346  050024  046440  041516  042101   FOUND2:  .ASCIZ  \ MNCAD (A/D)'S    \<15><12>
5347  050050  005012  025412  027461   MSG18:   .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
5348  050100  030455  031057  051514            .ASCIZ  \-1/2LSB\
5349  050110  005015  044504  043106   MSG20:   .ASCIZ  <15><12>/DIFFERENTIAL LINEARITY ON MNCAD UNIT # /
5350  050162  020040  020040  020040   MSG16:   .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
5351  050242  020040  020043  043117            .ASCII  /   # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
5352  050302  020040  020040  020040            .ASCII  /                                            STATE WIDTH (LSB)/<15>
5353  050406  030040  020040  020040            .ASCIZ  # 0              1/2              1              1 1/2              2#
5354  050507    122   046105  052101   MSG21:   .ASCIZ  /RELATIVE ACCURACY:/<15><12>
5355  050534  046040  041123  046440   LINEA:   .ASCIZ  / LSB MAXIMUM AT /
5356  050555    040   052073  052117   ERRTOT:  .ASCIZ  / ;TOTAL ERROR COUNT = /
5357  050604  035440  040502  020104   MESGD:   .ASCIZ  / ;BAD UNITS /
5358  050621    033   015462  005110   C2:      .ASCIZ  <33><62><33><110><12>   ;EXIT GRAPH MODE, HOME AND LINE FEED
5359  050627    112    000             C3:      .ASCIZ  <112>                ;ASCII 'I'
5360  050631    015   047412  043106   MOFSET:  .ASCIZ  <15><12>/OFFSET =/
5361  050644  046040  041123  000040   MLSB:    .ASCIZ  / LSB /
5362  050652  046040  041123  040440   MLSBAT:  .ASCIZ  / LSB AT /
5363                                            ;CODE TO SETUP BIT MAP VIEWING SCREEN
5364  050663    033    061            BUFF1:    .BYTE   33,61             ;GRAPH ON
5365  050665    101    061                      .BYTE   101,61            ;ENABLE HISTOGRAM 0
5366  050667    111    062                      .BYTE   111,62            ;CLEAR DATA + ENABLE VERT LINES
5367  050671    114    041    060               .BYTE   114,41,60         ;LOAD VERT LINE CORD
5368  050674    045    063                      .BYTE   45,63
5369  050676    051    066                      .BYTE   51,66
5370  050700    055    071                      .BYTE   55,71
5371  050702    061    074                      .BYTE   61,74
5372  050704    110    041    040               .BYTE   110,41,40         ;LOAD STARTING CORD.
5373  050707    112    000                      .BYTE   112,0             ;LOAD GRAPH 1 COMMAND <DATA TO FOLLOW>
5374  050711    033    061            BUFF2:    .BYTE   33,61             ;GRAPH ON
5375  050713    101    047                      .BYTE   101,47            ;ENABLE GRAPH 0 AND 1
5376  050715    111    061                      .BYTE   111,61            ;ENABLE DISPLAY
5377  050717    104    050    065               .BYTE   104,50,65,44,62   ;LOAD HORIZ CORDINATES
```

```
5378  050724    110    040     040              .BYTE   110,40,40           ;LOAD STARTING GRAPH CORD.
5379  050727    102    000                      .BYTE   102,0               ;LOAD GRAPH 0 <DATA TO FOLLOW>
5380  050731    033    061              INITVT:  .BYTE   33,61               ;GRAPH ON
5381  050733    101    040     040              .BYTE   101,40,40           ;DISABLE SCREEN
5382  050736    111    060     040              .BYTE   111,60,40           ;SET RECTANGEL ASPECT RATIO
5383  050741    033    062                      .BYTE   33,62               ;EXIT GRAPH MODE
5384                                     :       .BYTE   33,133,77,62,105    ;ENSURE ''ASCII'' <CAUSES HOLD SCREEN ON VT55>
5385  050743    033    110                      .BYTE   33,110              ;''HOME''
5386  050745    033    112     000              .BYTE   33,112,0            ;''ERASE SCREEN''
5387  050750    033    110              VTINIT:  .BYTE   33,110              ;''HOME''
5388  050752    033    112                      .BYTE   33,112              ;''ERASE SCREEN''
5389  050754    033    061                      .BYTE   33,61               ;ENTER GRAPHIC MODE
5390  050756    101    040                      .BYTE   101,40              ;CLEAR GRAPH DATA
5391  050760    033    062     000              .BYTE   33,62,0             ;EXIT GRAPHIC MODE
5392  050763    200    020114  020075   PRIME1:  .ASCII  <200>/L = LOGIC TESTS/
5393  051003    200    020127  020075           .ASCII  <200>/W = WRAPAROUND ANALOG TESTS/
5394  051037    200    020101  020075           .ASCII  <200>/A = AUTO TESTS/
5395  051056    050600 036440  050440           .ASCII  <200>/Q = QUIET AUTO TESTS/
5396  051103    200    020116  020075           .ASCII  <200>/N = NOISE TESTS ON SELECTED CHANNELS/
5397  051150    042200 036440  042040           .ASCII  <200>/D = DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY/
5398  051231    200    020115  020075           .ASCII  <200>/M = COMMON MODE REJECTION TEST FOR MNCAG CHANNELS/
5399  051313    200    020123  020075           .ASCII  <200>/S = SETTLING TEST ON SELECTED CHANNELS/
5400  051362    050200 036440  050040           .ASCII  <200>/P = PRINT CONVERTED ANALOG VALUE LOOP/
5401  051430    041600 036440  041440           .ASCII  <200>/C = CALIBRATION LOOP FOR MNCAD/
5402  051467    200    020106  020075           .ASCII  <200>/F = FRONT PANEL MODE AND GAIN SWITCH LOOP FOR MNCAG/
5403  051553    200    020124  020075           .ASCII  <200>/T = TEST MODULE VALUE VERIFY LOOP FOR MNCAG/
5404  051627    200    020111  020075           .ASCII  <200>/I = INCOMMING INSPECTION TEST FOR MNCTP/
5405  051677    200    020130  020075           .ASCII  <200>/X = READ-WRITE GAIN BITS LOOP/
5406  051735    200    020102  020075           .ASCII  <200>/B = BASE AND VECTOR ADDRESS CHANGES/
5407  052001    200    020107  020075           .ASCII  <200>/G = GET NEW SWITCH REGISTER VALUE/
5408  052043    200    020110  020075           .ASCIZ  <200>/H = HELP THE OPERATOR AND RETYPE THIS LIST     /
5409  052123    015    012              DOT:     .BYTE   15,12
5410  052125    124    050131  020105           .ASCIZ  /TYPE THE ''TEST CHARACTER'' THEN DEPRESS 'RETURN KEY''  /
5411  052213    115    041516  042101   EM1:     .ASCIZ  \MNCAD (A/D)      STATUS REG. ERROR\
5412  052251    115    041516  042101   EM2:     .ASCIZ  \MNCAD (A/D)      FAILED TO INTERRUPT\
5413  052311    115    041516  042101   EM3:     .ASCIZ  \MNCAD (A/D)      UNEXPECTED INTERRUPT\
5414  052352    047115 040503  020104   EM4:     .ASCIZ  #MNCAD (A/D)      ERROR ON A/D CHANNEL#
5415  052413    115    041516  042101   EM5:     .ASCIZ  \MNCAD (A/D)      EXISTING MNCAD NOW FAIL'S TO RESPOND\
5416  052474    047115 040503  020104   EM6:     .ASCIZ  \MNCAD (A/D)      DOES NOT EXIST <BUS ERROR> CHECK ADDRESS SWITCHES\
5417  052572    047111 047503  051122   EM7:     .ASCIZ  \INCORRECT I.D. VALUE\
5418  052617    111    041516  051117   EM10:    .ASCIZ  \INCORRECT ''MNCAG HOLD'' SIGNAL LEVEL\
5419  052663    111    041516  051117   EM11:    .ASCIZ  \INCORRECT MNCAG FRONT PANEL SWITCH POSITION\
5420  052737    115    041516  043501   EM12:    .ASCIZ  \MNCAG (PREAMP) GAIN REGISTER IN ERROR\
5421  053005    115    041516  050124   EM13:    .ASCIZ  \MNCTP (TC AMP) GAIN REGISTER IN ERROR\
5422  053053    042    040507  047111   EM14:    .ASCIZ  \''GAIN OR TC TYPE'' REGISTER IN ERROR\
5423  053117    115    041516  050124   EM15:    .ASCIZ  \MNCTP (TC AMP) CHANNEL VALUE IN ERROR\
5424  053165    125    044516  004524   DH1:     .ASCIZ  /UNIT   ERRPC   STREG  EXPECTED ACTUAL/
5425  053231    125    044516  004524   DH2:     .ASCIZ  /UNIT   ERRPC   STREG   CHANNEL NOMINAL  TOL.   ACTUAL/
5426  053315    125    044516  004524   DH3:     .ASCIZ  /UNIT   ERRPC   STREG  ACTUAL/
5427  053351    125    044516  004524   DH5:     .ASCIZ  /UNIT   ERRPC   WERE    ARE/
5428  053375    125    044516  004524   DH6:     .ASCIZ  /UNIT   ERRPC   STREG/
5429  053416    051105 050122  004503   DH7:     .ASCIZ  /ERRPC  ACTUAL  EXPECT     OR       OR/
5430  053454    047125 052111  042411   DH12:    .ASCIZ  /UNIT   ERRPC   STREG   CHAN    EXPECT  ACTUAL/
5431  053520    047125 052111  042411   DH15:    .ASCIZ  /UNIT   ERRPC   STREG   CHANNEL TP TYPE NOMINAL   TOL.   ACTUAL/
5432  053606    000                      THOUS:   .BYTE   0
5433  053607    000                      HUNS:    .BYTE   0
```

```
5434  053610    056              DECPNT: .BYTE   56
5435  053611    000              TENS:   .BYTE   0
5436  053612    000      000     ONES:   .BYTE   0,0
5437                                     .EVEN
5438                                     .LIST   BEX
5439  053614  001622  001116  001456  DT1:   UNITBD,$ERRPC, STREG, $GDDAT, $BDDAT,0
      053622  001124  001126  000000
5440  053630  001622  001116  001456  DT2:   UNITBD,$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
      053636  001556  001124  001566
      053644  001126  000000
5441  053650  001622  001116  001456  DT3:   UNITBD,$ERRPC,STREG,$BDDAT,0
      053656  001126  000000
5442  053662  001622  001116  001202  DT5:   UNITBD,$ERRPC,$UNIT,TEMP,0
      053670  001540  000000
5443  053674  001622  001116  001456  DT6:   UNITBD,$ERRPC,STREG,0
      053702  000000
5444  053704  001116  001126  022136  DT7:   $ERRPC,$BDDAT,K60,K20,K340,0
      053712  022140  022142  000000
5445  053720  001622  001116  001456  DT12:  UNITBD,$ERRPC,STREG,CHANL,$GDDAT,$BDDAT,0
      053726  001556  001124  001126
      053734  000000
5446  053736  001622  001116  001456  DT15:  UNITBD,$ERRPC,STREG,CHANL,TPVAL,$GDDAT,SPREAD,$BDDAT,0
      053744  001556  013546  001124
      053752  001566  001126  000000
5447  053760    000     000     000  DF1:   .BYTE   0,0,0,0,0,0,0,0,0,0
      053763    000     000     000
      053766    000     000     000
      053771    000
```

```
 5449                               .SBTTL   TTY INPUT ROUTINE
  (1)
  (2)                               ;;******************************************************************
  (1)                               .ENABL  LSB
  (1) 053772  000000                $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
  (1) 053774  000000                $TKQIN: .WORD   0               ;;INPUT POINTER
  (1) 053776  000000                $TKQOUT: .WORD  0               ;;OUTPUT POINTER
  (1) 054000  000040                $TKQSRT: .BLKB  32.             ;;TTY KEYBOARD QUEUE
  (1)         054040                $TKQEND=.
  (1)
  (1)                               ;*TK INITIALIZE ROUTINE
  (1)                               ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
  (1)                               ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
  (1)                               ;
  (1)                               ;*CALL:                                           .
  (1)                               ;*      JSR     PC,$TKINT
  (1)                               ;*      RETURN
  (1)                               ;
  (1) 054040  005037  053772        $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
  (1) 054044  012737  054000  053774        MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
  (1) 054052  013737  053774  053776        MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
  (1) 054060  012737  054110  000060        MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
  (1) 054066  012737  000200  000062        MOV     #200,@#TKVEC+2  ;;''BR'' LEVEL 4
  (1) 054074  005777  125046                TST     @$TKB           ;;CLEAR DONE FLAG
  (1) 054100  012777  000100  125036        MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
  (1) 054106  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) 054110  117746  125032        $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
  (1) 054114  042716  177600                BIC     #^C177,(SP)     ;;STRIP THE JUNK
  (1) 054120  021627  000003                CMP     (SP),#3         ;;IS IT A CONTROL C?
  (1) 054124  001007                        BNE     1$              ;;BRANCH IF NO
  (1) 054126  104401  055260                TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
  (1) 054132  004737  054040                JSR     PC,$TKINT       ;;INIT THE KEYBOARD
  (1) 054136  005726                        TST     (SP)+           ;;CLEAN UP STACK
  (1) 054140  000137  001674                JMP     BEG2            ;;CONTROL C RESTART
  (1) 054144  021627  000007        1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
  (1) 054150  001004                        BNE     2$              ;;BRANCH IF NO
  (1) 054152  022737  000176  001140        CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
  (1) 054160  001500                        BEQ     6$              ;;GO TO SWR CHANGE
  (1)
  (1) 054162                        2$:
  (1) 054162  022737  000040  053772        CMP     #32.,$TKCNT     ;;IS THE QUEUE FULL?
  (1) 054170  001004                        BNE     3$              ;;BRANCH IF NO
  (1) 054172  104401  055254                TYPE    ,$BELL          ;;RING THE TTY BELL
  (1) 054176  005726                        TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
  (1) 054200  000451                        BR      5$              ;;EXIT
  (1) 054202  021627  000023        3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
  (1) 054206  001021                        BNE     32$             ;;BRANCH IF NO
  (1) 054210  005077  124730                CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
```

```
(1)   054214  005726                           TST     (SP)+               ;;CLEAN CHAR OFF STACK
(1)   054216  105777  124722          31$:     TSTB    @$TKS               ;;WAIT FOR A CHAR
(1)   054222  100375                            BPL     31$                 ;;LOOP UNTIL ITS THERE
(1)   054224  117746  124716                   MOVB    @$TKB,-(SP)         ;;GET THE CHARACTER
(1)   054230  042716  177600                   BIC     #^C177,(SP)         ;;MAKE IT 7-BIT ASCII
(1)   054234  022627  000021                   CMP     (SP)+,#21           ;;IS IT A CONTROL-Q?
(1)   054240  001366                            BNE     31$                 ;;BRANCH IF NO
(1)   054242  012777  000100  124674           MOV     #100,@$TKS          ;;REENABLE TTY KEYBOARD INTERRUPTS
(1)   054250  000002                            RTI                         ;;RETURN
(1)   054252  005237  053772          32$:     INC     $TKCNT              ;;COUNT THIS CHARACTER
(1)   054256  021627  000140                   CMP     (SP),#140           ;;IS IT UPPER CASE?
(1)   054262  002405                            BLT     4$                  ;;BRANCH IF YES
(1)   054264  021627  000175                   CMP     (SP),#175           ;;IS IT A SPECIAL CHAR?
(1)   054270  003002                            BGT     4$                  ;;BRANCH IF YES
(1)   054272  042716  000040                   BIC     #40,(SP)            ;;MAKE IT UPPER CASE
(1)   054276  112677  177472          4$:      MOVB    (SP)+,@$TKQIN       ;;AND PUT IT IN QUEUE
(1)   054302  005237  053774                   INC     $TKQIN              ;;UPDATE THE POINTER
(1)   054306  023727  053774  054040           CMP     $TKQIN,#$TKQEND     ;;GO OFF THE END?
(1)   054314  001003                            BNE     5$                  ;;BRANCH IF NO
(1)   054316  012737  054000  053774           MOV     #$TKQSRT,$TKQIN     ;;RESET THE POINTER
(1)   054324  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)   054326  022737  000176  001140  $CKSWR:  CMP     #SWREG,SWR          ;;IS THE SOFT-SWR SELECTED
(1)   054334  001124                            BNE     15$                 ;;EXIT IF NOT
(1)   054336  105777  124602                   TSTB    @$TKS               ;;IS A CHAR WAITING?
(1)   054342  100121                            BPL     15$                 ;;IF NOT, EXIT
(1)   054344  117746  124576                   MOVB    @$TKB,-(SP)         ;;YES
(1)   054350  042716  177600                   BIC     #^C177,(SP)         ;;MAKE IT  7-BIT ASCII
(1)   054354  021627  000007                   CMP     (SP),#7             ;;IS IT A CONTROL-G?
(1)   054360  001300                            BNE     2$                  ;;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)   054362  123727  001134  000001  6$:      CMPB    $AUTOB,#1           ;;ARE WE RUNNING IN AUTO-MODE?
(1)   054370  001674                            BEQ     2$                  ;;BRANCH IF YES
(1)   054372  005726                            TST     (SP)+               ;;CLEAR CONTROL-G OFF STACK
(1)   054374  004737  054040                   JSR     PC,$TKINT           ;;FLUSH THE TTY INPUT QUEUE
(1)   054400  005077  124540                   CLR     @$TKS               ;;DISABLE TTY KEYBOARD INTERRUPTS
(1)   054404  112737  000001  001135           MOVB    #1,$INTAG           ;;SET INTERRUPT MODE INDICATOR
(1)
(1)   054412  104401  055272                   TYPE    ,$CNTLG             ;;ECHO THE CONTROL-G (^G)
(1)   054416  104401  055277          $GTSWR:  TYPE    ,$MSWR              ;;TYPE CURRENT CONTENTS
(2)   054422  013746  000176                   MOV     SWREG,-(SP)         ;;SAVE SWREG FOR TYPEOUT
(2)   054426  104402                            TYPOC                       ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)   054430  104401  055310                   TYPE    ,$MNEW              ;;PROMPT FOR NEW SWR
(1)   054434  005046                   19$:    CLR     -(SP)               ;;CLEAR COUNTER
(1)   054436  005046                            CLR     -(SP)               ;;THE NEW SWR
(1)   054440  105777  124500          7$:      TSTB    @$TKS               ;;CHAR THERE?
```

```
(1)   054444  100375                          BPL      7$                ;;IF NOT TRY AGAIN
(1)
(1)   054446  117746  124474                  MOVB     @$TKB,-(SP)       ;;PICK UP CHAR
(1)   054452  042716  177600                  BIC      #^C177,(SP)       ;;MAKE IT 7-BIT ASCII
(1)
(1)   054456  021627  000003                  CMP      (SP),#3           ;;IS IT A CONTROL-C?
(1)   054462  001015                          BNE      9$                ;;BRANCH IF NOT
(1)   054464  104401  055260                  TYPE     ,$CNTLC           ;;YES, ECHO CONTROL-C (^C)
(1)   054470  062706  000006                  ADD      #6,SP             ;;CLEAN UP STACK
(1)   054474  123727  001135  000001          CMPB     $INTAG,#1         ;;REENABLE TTY KEYBOARD INTERRUPTS?
(1)   054502  001003                          BNE      8$                ;;BRANCH IF NO
(1)   054504  012777  000100  124432          MOV      #100,@$TKS        ;;ALLOW TTY KEYBOARD INTERRUPTS
(1)   054512  000137  001674         8$:      JMP      BEG2              ;;CONTROL-C RESTART
(1)
(1)
(1)   054516  021627  000025         9$:      CMP      (SP),#25          ;;IS IT A CONTROL-U?
(1)   054522  001005                          BNE      10$               ;;BRANCH IF NOT
(1)   054524  104401  055265                  TYPE     ,$CNTLU           ;;YES, ECHO CONTROL-U (^U)
(1)   054530  062706  000006         20$:     ADD      #6,SP             ;;IGNORE PREVIOUS INPUT
(1)   054534  000737                          BR       19$               ;;LET'S TRY IT AGAIN
(1)
(1)
(1)   054536  021627  000015         10$:     CMP      (SP),#15          ;;IS IT A <CR>?
(1)   054542  001022                          BNE      16$               ;;BRANCH IF NO
(1)   054544  005766  000004                  TST      4(SP)             ;;YES, IS IT THE FIRST CHAR?
(1)   054550  001403                          BEQ      11$               ;;BRANCH IF YES
(1)   054552  016677  000002  124360          MOV      2(SP),@SWR        ;;SAVE NEW SWR
(1)   054560  062706  000006         11$:     ADD      #6,SP             ;;CLEAR UP STACK
(1)   054564  104401  001165         14$:     TYPE     ,$CRLF            ;;ECHO <CR> AND <LF>
(1)   054570  123727  001135  000001          CMPB     $INTAG,#1         ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)   054576  001003                          BNE      15$               ;;BRANCH IF NOT
(1)   054600  012777  000100  124336          MOV      #100,@$TKS        ;;RE-ENABLE TTY KBD INTERRUPTS
(1)   054606  000002         15$:             RTI                        ;;RETURN
(1)   054610  004737  056742         16$:     JSR      PC,$TYPEC         ;;ECHO CHAR
(1)   054614  021627  000060                  CMP      (SP),#60          ;;CHAR < 0?
(1)   054620  002420                          BLT      18$               ;;BRANCH IF YES
(1)   054622  021627  000067                  CMP      (SP),#67          ;;CHAR > 7?
(1)   054626  003015                          BGT      18$               ;;BRANCH IF YES
(1)   054630  042726  000060                  BIC      #60,(SP)+         ;;STRIP-OFF ASCII
(1)   054634  005766  000002                  TST      2(SP)             ;;IS THIS THE FIRST CHAR
(1)   054640  001403                          BEQ      17$               ;;BRANCH IF YES
(1)   054642  006316                          ASL      (SP)              ;;NO, SHIFT PRESENT
(1)   054644  006316                          ASL      (SP)              ;;   CHAR OVER TO MAKE
(1)   054646  006316                          ASL      (SP)              ;;   ROOM FOR NEW ONE.
(1)   054650  005266  000002         17$:     INC      2(SP)             ;;KEEP COUNT OF CHAR
(1)   054654  056616  177776                  BIS      -2(SP),(SP)       ;;SET IN NEW CHAR
(1)   054660  000667                          BR       7$                ;;GET THE NEXT ONE
(1)   054662  104401  001164         18$:     TYPE     ,$QUES            ;;TYPE ?<CR><LF>
(1)   054666  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)    054670  011646                  $RDCHR: MOV       (SP),-(SP)             ;;PUSH DOWN THE PC AND
(1)    054672  016666  000004  000002          MOV       4(SP),2(SP)            ;;THE PS
(1)    054700  005066  000004                  CLR       4(SP)                  ;;GET READY FOR A CHARACTER
(2)    054704  005046                          CLR       -(SP)                  ;;PUT NEW PS ON STACK
(2)    054706  012746  054714                  MOV       #64$,-(SP)             ;;PUT NEW PC ON  STACK
(2)    054712  000002                          RTI                              ;;POP NEW PC AND PS
(2)    054714                          64$:
(1)    054714  005737  053772          1$:     TST       $TKCNT                 ;;WAIT ON A CHARACTER
(1)    054720  001775                          BEQ       1$
(1)    054722  005337  053772                  DEC       $TKCNT                 ;;DECREMENT THE COUNTER
(1)    054726  117766  177044  000004          MOVB      @$TKQOUT,4(SP)         ;;GET ONE CHARACTER
(1)    054734  005237  053776                  INC       $TKQOUT                ;;UPDATE THE POINTER
(1)    054740  023727  053776  054040          CMP       $TKQOUT,#$TKQEND       ;;DID IT GO OFF OF THE END?
(1)    054746  001003                          BNE       2$                     ;;BRANCH IF NO
(1)    054750  012737  054000  053776          MOV       #$TKQSRT,$TKQOUT       ;;RESET THE POINTER
(1)    054756  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)    054760  010346                  $RDLIN: MOV       R3,-(SP)               ;;SAVE R3
(1)    054762  005046                          CLR       -(SP)                  ;;CLEAR THE RUBOUT KEY
(1)    054764  012703  055214          1$:     MOV       #$TTYIN,R3             ;;GET ADDRESS
(1)    054770  022703  055254          2$:     CMP       #$TTYIN+32.,R3         ;;BUFFER FULL?
(1)    054774  101456                          BLOS      4$                     ;;BR IF YES
(1)    054776  104411                          RDCHR                            ;;GO READ ONE CHARACTER FROM THE TTY
(1)    055000  112613                          MOVB      (SP)+,(R3)             ;;GET CHARACTER
(1)    055002  122713  000177          10$:    CMPB      #177,(R3)              ;;IS IT A RUBOUT
(1)    055006  001022                          BNE       5$                     ;;BR IF NO
(1)    055010  005716                          TST       (SP)                   ;;IS THIS THE FIRST RUBOUT?
(1)    055012  001007                          BNE       6$                     ;;BR IF NO
(1)    055014  112737  000134  055212          MOVB      #'\,9$                 ;;TYPE A BACK SLASH
(1)    055022  104401  055212                  TYPE      ,9$
(1)    055026  012716  177777                  MOV       #-1,(SP)               ;;SET THE RUBOUT KEY
(1)    055032  005303                  6$:     DEC       R3                     ;;BACKUP BY ONE
(1)    055034  020327  055214                  CMP       R3,#$TTYIN             ;;STACK EMPTY?
(1)    055040  103434                          BLO       4$                     ;;BR IF YES
(1)    055042  111337  055212                  MOVB      (R3),9$                ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)    055046  104401  055212                  TYPE      ,9$                    ;;GO TYPE
(1)    055052  000746                          BR        2$                     ;;GO READ ANOTHER CHAR.
(1)    055054  005716                  5$:     TST       (SP)                   ;;RUBOUT KEY SET?
(1)    055056  001406                          BEQ       7$                     ;;BR IF NO
(1)    055060  112737  000134  055212          MOVB      #'\,9$                 ;;TYPE A BACK SLASH
(1)    055066  104401  055212                  TYPE      ,9$
(1)    055072  005016                          CLR       (SP)                   ;;CLEAR THE RUBOUT KEY
(1)    055074  122713  000025          7$:     CMPB      #25,(R3)               ;;IS CHARACTER A CTRL U?
(1)    055100  001003                          BNE       8$                     ;;BR IF NO
(1)    055102  104401  055265                  TYPE      ,$CNTLU                ;;TYPE A CONTROL 'U'
(1)    055106  000726                          BR        1$                     ;;GO START OVER
```

```
(1)   055110   122713   000022      8$:    CMPB    #22,(R3)         ::IS CHARACTER A ''^R''?
(1)   055114   001011                      BNE     3$               ;:BRANCH IF NO
(1)   055116   105013                      CLRB    (R3)             ;:CLEAR THE CHARACTER
(1)   055120   104401   001165             TYPE    ,$CRLF           ;:TYPE A ''CR'' & 'LF''
(1)   055124   104401   055214             TYPE    ,$TTYIN          ;:TYPE THE INPUT STRING
(1)   055130   000717                      BR      2$               ;:GO PICKUP ANOTHER CHACTER
(1)   055132   104401   001164      4$:    TYPE    ,$QUES           ;:TYPE A '?'
(1)   055136   000712                      BR      1$               ;:CLEAR THE BUFFER AND LOOP
(1)   055140   111337   055212      3$:    MOVB    (R3),9$          ;:ECHO THE CHARACTER
(1)   055144   104401   055212             TYPE    ,9$
(1)   055150   122723   000015             CMPB    #15,(R3)+        ;:CHECK FOR RETURN
(1)   055154   001305                      BNE     2$               ;:LOOP IF NOT RETURN
(1)   055156   105063   177777             CLRB    -1(R3)           ;:CLEAR RETURN (THE 15)
(1)   055162   104401   001166             TYPE    ,$LF             ;:TYPE A LINE FEED
(1)   055166   005726                      TST     (SP)+            ;:CLEAN RUBOUT KEY FROM THE STACK
(1)   055170   012603                      MOV     (SP)+,R3         ;:RESTORE R3
(1)   055172   011646                      MOV     (SP),-(SP)       ;:ADJUST THE STACK AND PUT ADDRESS OF THE
(1)   055174   016666   000004  000002     MOV     4(SP),2(SP)      ;:      FIRST ASCII CHARACTER ON IT
(1)   055202   012766   055214  000004     MOV     #$TTYIN,4(SP)
(1)   055210   000002                      RTI                      ;:RETURN
(1)   055212      000              9$:     .BYTE   0                ;:STORAGE FOR ASCII CHAR. TO TYPE
(1)   055213      000                      .BYTE   0                ;:TERMINATOR
(1)   055214   000040              $TTYIN: .BLKB   32.              ;:RESERVE 32. BYTES FOR TTY INPUT
(1)   055254   177607   000377     $BELL:  .ASCIZ  <207><377><377> ;:CODE FOR BELL
(1)   055260   041536   005015    000      $CNTLC: .ASCIZ  /^C/<15><12>     ;:CONTROL 'C'
(1)   055265      136   006525  000012     $CNTLU: .ASCIZ  /^U/<15><12>     ;:CONTROL 'U'
(1)   055272   043536   005015    000      $CNTLG: .ASCIZ  /^G/<15><12>     ;:CONTROL 'G'
(1)   055277      015   051412  051127     $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)   055304   036440   000040
(1)   055310   020040   042516  020127     $MNEW:  .ASCIZ  /  NEW = /
(1)   055316   020075      000
(1)            055322                       .EVEN
```

```
 5451                             .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) 055322 011646      $RDOCT:  MOV      (SP),-(SP)             ;;PROVIDE SPACE FOR THE
  (1) 055324 016666 000004 000002 MOV      4(SP),2(SP)            ;;INPUT NUMBER
  (3) 055332 010046              MOV      R0,-(SP)               ;;PUSH R0 ON STACK
  (3) 055334 010146              MOV      R1,-(SP)               ;;PUSH R1 ON STACK
  (3) 055336 010246              MOV      R2,-(SP)               ;;PUSH R2 ON STACK
  (1) 055340 104412      1$:      RDLIN                           ;;READ AN ASCIZ LINE
  (1) 055342 012600              MOV      (SP)+,R0               ;;GET ADDRESS OF 1ST CHARACTER
  (1) 055344 005001              CLR      R1                     ;;CLEAR DATA WORD
  (1) 055346 005002              CLR      R2
  (1) 055350 112046      2$:      MOVB     (R0)+,-(SP)            ;;PICKUP THIS CHARACTER
  (1) 055352 001412              BEQ      3$                     ;;IF ZERO GET OUT
  (1) 055354 006301              ASL      R1                     ;;*2
  (1) 055356 006102              ROL      R2
  (1) 055360 006301              ASL      R1                     ;;*4
  (1) 055362 006102              ROL      R2
  (1) 055364 006301              ASL      R1                     ;;*8
  (1) 055366 006102              ROL      R2
  (1) 055370 042716 177770       BIC      #^C7,(SP)              ;;STRIP THE ASCII JUNK
  (1) 055374 062601              ADD      (SP)+,R1               ;;ADD IN THIS DIGIT
  (1) 055376 000764              BR       2$                     ;;LOOP
  (1) 055400 005726      3$:      TST      (SP)+                  ;;CLEAN TERMINATOR FROM STACK
  (1) 055402 010166 000012       MOV      R1,12(SP)              ;;SAVE THE RESULT
  (1) 055406 010237 055422       MOV      R2,$HIOCT
  (3) 055412 012602              MOV      (SP)+,R2               ;;POP STACK INTO R2
  (3) 055414 012601              MOV      (SP)+,R1               ;;POP STACK INTO R1
  (3) 055416 012600              MOV      (SP)+,R0               ;;POP STACK INTO R0
  (1) 055420 000002              RTI                             ;;RETURN
  (1) 055422 000000      $HIOCT:  .WORD    0                      ;;HIGH ORDER BITS GO HERE
```

```
 5453                                   .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)  055424                           $SCOPE:
  (1)  055424  104410                           CKSWR                     ;;TEST  FOR CHANGE IN SOFT-SWR
  (1)  055426  032777  040000  123504   1$:     BIT     #BIT14,@SWR       ;;LOOP ON PRESENT TEST?
  (1)  055434  001114                           BNE     $OVER             ;;YES IF SW14=1
  (1)                                   ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  055436  000416                   $XTSTR: BR      6$                ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
  (1)                                                                     ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
  (1)  055440  013746  000004                   MOV     @#ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  055442  012737  055464  000004           MOV     #5$,@#ERRVEC      ;SET FOR TIMEOUT
  (1)  055452  005737  177060                   TST     @#177060          ;;TIME OUT ON XOR?
  (1)  055456  012637  000004                   MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
  (1)  055462  000463                           BR      $SVLAD            ;;GO TO THE NEXT TEST
  (1)  055464  022626                   5$:     CMP     (SP)+,(SP)+       ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  055466  012637  000004                   MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
  (1)  055472  000423                           BR      7$                ;;LOOP ON THE PRESENT TEST
  (1)  055474                           6$:;;#####END OF CODE FOR THE XOR TESTER#####
  (1)  055474  032777  000400  123436           BIT     #BIT08,@SWR       ;;LOOP ON SPEC. TEST?
  (1)  055502  001404                           BEQ     2$                ;;BR IF NO
  (1)  055504  127737  123430  001102           CMPB    @SWR,$TSTNM       ;;ON THE RIGHT TEST?      SWR<7:0>
  (1)  055512  001465                           BEQ     $OVER             ;;BR IF YES
  (1)  055514  105737  001103           2$:     TSTB    $ERFLG            ;;HAS AN ERROR OCCURRED?
  (1)  055520  001421                           BEQ     3$                ;;BR IF NO
  (1)  055522  123737  001115  001103           CMPB    $ERMAX,$ERFLG     ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  055530  101015                           BHI     3$                ;;BR IF NO
  (1)  055532  032777  001000  123400           BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
  (1)  055540  001404                           BEQ     4$                ;;BR IF NO
  (1)  055542  013737  001110  001106   7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
  (1)  055550  000446                           BR      $OVER
  (1)  055552  105037  001103           4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
  (1)  055556  005037  001160                   CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  (1)  055562  000415                           BR      1$                ;;ESCAPE TO THE NEXT TEST
  (1)  055564  032777  004000  123346   3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
  (1)  055572  001011                           BNE     1$                ;;BR IF YES
  (1)  055574  005737  001176                   TST     $PASS             ;;IF FIRST PASS OF PROGRAM
  (1)  055600  001406                           BEQ     1$                ;;      INHIBIT ITERATIONS
  (1)  055602  005237  001104                   INC     $ICNT             ;;INCREMENT ITERATION COUNT
  (1)  055606  023737  001160  001104           CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
  (1)  055614  002024                           BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
  (1)  055616  012737  000001  001104   1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
  (1)  055624  013737  055702  001160           MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
  (1)  055632  105237  001102           $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
  (1)  055636  113737  001102  001174           MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
```

```
   (1)  055644  011637  001106              MOV    (SP),$LPADR       ::SAVE SCOPE LOOP ADDRESS
   (1)  055650  011637  001110              MOV    (SP),$LPERR       ::SAVE ERROR LOOP ADDRESS
   (1)  055654  005037  001162              CLR    $ESCAPE           ::CLEAR THE ESCAPE FROM ERROR ADDRESS
   (1)  055660  112737  000001  001115      MOVB   #1,$ERMAX         ::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
   (1)  055666  013777  001102  123246  $OVER:  MOV  $TSTNM,@DISPLAY ::DISPLAY TEST NUMBER
   (1)  055674  013716  001106              MOV    $LPADR,(SP)       ::FUDGE RETURN ADDRESS
   (1)  055700  000002                      RTI                      ::FIXES PS
   (1)  055702  003720              $MXCNT: 2000.                    ::MAX. NUMBER OF ITERATIONS
  5454  055704  053737  001620  001616  WHICHV: BIS  MASKNM,BADUNT   :SET CURRENT UNIT INTO BAD FIELD
  5455  055712  013737  001620  055742  WHICHU: MOV  MASKNM,11$      :GET CURRENT UNIT
  5456  055720  012737  000000  001622      MOV    #0,UNITBD         :PRIME THE VALUE
  5457  055726  006237  055742      10$:    ASR    11$               :CONVERT
  5458  055732  001404                      BEQ    12$               :BR WHEN DONE
  5459  055734  005237  001622              INC    UNITBD            :BUMP POINTER
  5460  055740  000772                      BR     10$
  5461  055742  000000              11$:    0
  5462  055744  000207              12$:    RTS    PC                :EXIT
  5466                              .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)  055746              $ERROR:
   (1)  055746  104410              CKSWR                    ::TEST FOR CHANGE IN SOFT-SWR
   (5)  055750  004737  055704              JSR    PC,WHICHV         ;INDICATE BAD UNIT
   (1)  055754  105237  001103      7$:     INCB   $ERFLG            ::SET THE ERROR FLAG
   (1)  055760  001775                      BEQ    7$                ::DON'T LET THE FLAG GO TO ZERO
   (1)  055762  013777  001102  123152      MOV    $TSTNM,@DISPLAY   ::DISPLAY TEST NUMBER AND ERROR FLAG
   (1)  055770  005237  001112              INC    $ERTTL            ::INC THE ERROR COUNT
   (1)  055774  011637  001116              MOV    (SP),$ERRPC       ::GET ADDRESS OF ERROR INSTRUCTION
   (1)  056000  162737  000002  001116      SUB    #2,$ERRPC
   (1)  056006  117737  123104  001114      MOVB   @$ERRPC,$ITEMB    ::STRIP AND SAVE THE ERROR ITEM CODE
   (1)  056014  032777  020000  123116      BIT    #BIT13,@SWR       ::SKIP TYPEOUT IF SET
   (1)  056022  001004                      BNE    20$               ::SKIP TYPEOUTS
   (1)  056024  004737  056136              JSR    PC,$ERRTYP        ::GO TO USER ERROR ROUTINE
   (1)  056030  104401  001165              TYPE   .$CRLF
   (1)  056034              20$:
   (1)  056034  122737  000001  001210      CMPB   #APTENV,$ENV      ::RUNNING IN APT MODE
   (1)  056042  001007                      BNE    2$                ::NO,SKIP APT ERROR REPORT
   (1)  056044  113737  001114  056056      MOVB   $ITEMB,21$        ::SET ITEM NUMBER AS ERROR NUMBER
   (1)  056052  004737  057326              JSR    PC,$ATY4          ::REPORT FATAL ERROR TO APT
   (1)  056056  000              21$:    .BYTE  0
   (1)  056057  000                      .BYTE  0
   (1)  056060  000777              22$:    BR     22$               ::APT ERROR LOOP
   (1)  056062  005777  123052      2$:     TST    @SWR              ::HALT ON ERROR
   (1)  056066  100002                      BPL    3$                ::SKIP IF CONTINUE
   (1)  056070  000000                      HALT                     ::HALT ON ERROR!
   (1)  056072  104410                      CKSWR                    ::TEST FOR CHANGE IN SOFT-SWR
```

```
    (1)   056074   032777  001000  123036  3$:     BIT     #BIT09,@SWR       ;;LOOP ON ERROR SWITCH SET?
    (1)   056102   001402                          BEQ     4$                ;;BR IF NO
    (1)   056104   013716  001110                  MOV     $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
    (1)   056110   005737  001162          4$:     TST     $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
    (1)   056114   001402                          BEQ     5$                ;;BR IF NONE
    (1)   056116   013716  001162                  MOV     $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
    (1)   056122                          5$:
    (1)   056122   022737  040242  000042          CMP     #$ENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
    (1)   056130   001001                          BNE     6$                ;;BRANCH IF NO
    (1)   056132   000000                          HALT                      ;;YES
    (1)   056134                          6$:
    (1)   056134   000002                          RTI                       ;;RETURN
   5467                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
    (1)
    (2)                                    ;;****************************************************************
    (1)                                    ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($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)   056136                          $ERRTYP:
    (1)   056136   104401  001165                  TYPE    ,$CRLF            ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   056142   010046                          MOV     R0,-(SP)          ;;SAVE R0
    (1)   056144   005000                          CLR     R0                ;;PICKUP THE ITEM INDEX
    (1)   056146   153700  001114                  BISB    @#$ITEMB,R0
    (1)   056152   001004                          BNE     1$                ;;IF ITEM NUMBER IS ZERO, JUST
    (1)                                                                      ;;TYPE THE PC OF THE ERROR
    (2)   056154   013746  001116                  MOV     $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
    (2)                                                                      ;;ERROR ADDRESS
    (2)   056160   104402                          TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)   056162   000445                          BR      10$               ;;GET OUT
    (1)   056164   005300                  1$:     DEC     R0                ;;ADJUST THE INDEX SO THAT IT WILL
    (1)   056166   006300                          ASL     R0                ;;      WORK FOR THE ERROR TABLE
    (1)   056170   006300                          ASL     R0
    (1)   056172   006300                          ASL     R0
    (1)   056174   062700  001252                  ADD     #$ERRTB,R0        ;;FORM TABLE POINTER
    (1)   056200   012037  056210                  MOV     (R0)+,2$          ;;PICKUP 'ERROR MESSAGE'' POINTER
    (1)   056204   001404                          BEQ     3$                ;;SKIP TYPEOUT IF NO POINTER
    (1)   056206   104401                          TYPE                      ;;TYPE THE 'ERROR MESSAGE''
    (1)   056210   000000                  2$:     .WORD   0                 ;;''ERROR MESSAGE'' POINTER GOES HERE
    (1)   056212   104401  001165                  TYPE    ,$CRLF            ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   056216   012037  056226          3$:     MOV     (R0)+,4$          ;;PICKUP 'DATA HEADER'' POINTER
    (1)   056222   001404                          BEQ     5$                ;;SKIP TYPEOUT IF 0
    (1)   056224   104401                          TYPE                      ;;TYPE THE 'DATA HEADER''
    (1)   056226   000000                  4$:     .WORD   0                 ;;''DATA HEADER'' POINTER GOES HERE
    (1)   056230   104401  001165                  TYPE    ,$CRLF            ;;''CARRIAGE RETURN'' & ''LINE FEED''
    (1)   056234   010146                  5$:     MOV     R1,-(SP)          ;;SAVE R1
    (1)   056236   012001                          MOV     (R0)+,R1          ;;PICKUP 'DATA TABLE'' POINTER
    (1)   056240   001415                          BEQ     9$                ;;BR IF NO DATA TO BE TYPED
    (1)   056242   012000                          MOV     (R0)+,R0          ;;PICKUP 'DATA FORMAT'' POINTER
    (1)   056244   105720                  6$:     TSTB    (R0)+             ;;''OCTAL'' OR 'DECIMAL''
    (1)   056246   001003                          BNE     7$                ;;BR IF DECIMAL
    (2)   056250   013146                          MOV     @(R1)+,-(SP)      ;;SAVE @(R1)+ FOR TYPEOUT
    (2)   056252   104402                          TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)   056254   000402                          BR      8$
    (1)   056256                          7$:
    (2)   056256   013146                          MOV     @(R1)+,-(SP)      ;;SAVE @(R1)+ FOR TYPEOUT
```

```
 (2)   056260  104405                        TYPDS                       ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)   056262  005711              8$:       TST     (R1)                ;;IS THERE ANOTHER NUMBER?
 (1)   056264  001403                        BEQ     9$                  ;;BR IF NO
 (1)   056266  104401  056306               TYPE    ,11$                ;;TYPE TWO(2) SPACES
 (1)   056272  000764                        BR      6$                  ;;LOOP
 (1)
 (1)   056274  012601              9$:       MOV     (SP)+,R1            ;;RESTORE R1
 (1)   056276  012600              10$:      MOV     (SP)+,R0            ;;RESTORE R0
 (1)   056300  104401  001165               TYPE    ,$CRLF              ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)   056304  000207                        RTS     PC                  ;;RETURN
 (1)   056306  020040  000         11$:      .ASCIZ  /  /                ;;TWO(2) SPACES
 (1)           056312                        .EVEN
5468                                 .SBTTL  POWER DOWN AND UP ROUTINES
 (1)
 (2)           )                     ;;********************************************************************
 (1)                                 ;POWER DOWN ROUTINE
 (1)   056312  012737  056456  000024  $PWRDN: MOV   #$ILLUP,@#PWRVEC   ;;SET FOR FAST UP
 (1)   056320  012737  000340  000026         MOV   #340,@#PWRVEC+2    ;;PRIO:7
 (3)   056326  010046                         MOV   R0,-(SP)          ;;PUSH R0 ON STACK
 (3)   056330  010146                         MOV   R1,-(SP)          ;;PUSH R1 ON STACK
 (3)   056332  010246                         MOV   R2,-(SP)          ;;PUSH R2 ON STACK
 (3)   056334  010346                         MOV   R3,-(SP)          ;;PUSH R3 ON STACK
 (3)   056336  010446                         MOV   R4,-(SP)          ;;PUSH R4 ON STACK
 (3)   056340  010546                         MOV   R5,-(SP)          ;;PUSH R5 ON STACK
 (3)   056342  017746  122572                 MOV   @SWR,-(SP)        ;;PUSH @SWR ON STACK
 (1)   056346  010637  056462                 MOV   SP,$SAVR6         ;;SAVE SP
 (1)   056352  012737  056364  000024         MOV   #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
 (1)   056360  000000                         HALT
 (1)   056362  000776                         BR    .-2               ;;HANG UP
 (1)
 (2)                                 ;;********************************************************************
 (1)                                 ;POWER UP ROUTINE
 (1)   056364  012737  056456  000024  $PWRUP: MOV   #$ILLUP,@#PWRVEC   ;;SET FOR FAST DOWN
 (1)   056372  013706  056462                 MOV   $SAVR6,SP         ;;GET SP
 (1)   056376  005037  056462                 CLR   $SAVR6            ;;WAIT LOOP FOR THE TTY
 (1)   056402  005237  056462         1$:     INC   $SAVR6            ;;WAIT FOR THE INC
 (1)   056406  001375                         BNE   1$                ;;OF  WORD
 (3)   056410  012677  122524                 MOV   (SP)+,@SWR        ;;POP STACK INTO @SWR
 (3)   056414  012605                         MOV   (SP)+,R5          ;;POP STACK INTO R5
 (3)   056416  012604                         MOV   (SP)+,R4          ;;POP STACK INTO R4
 (3)   056420  012603                         MOV   (SP)+,R3          ;;POP STACK INTO R3
 (3)   056422  012602                         MOV   (SP)+,R2          ;;POP STACK INTO R2
 (3)   056424  012601                         MOV   (SP)+,R1          ;;POP STACK INTO R1
 (3)   056426  012600                         MOV   (SP)+,R0          ;;POP STACK INTO R0
 (1)   056430  012737  056312  000024         MOV   #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
 (1)   056436  012737  000340  000026         MOV   #340,@#PWRVEC+2   ;;PRIO:7
 (1)   056444  104401                         TYPE                    ;;REPORT THE POWER FAILURE
 (1)   056446  056464              $PWRMG: .WORD  PWRMSG               ;;POWER FAIL MESSAGE POINTER
 (1)   056450  012716                         MOV   (PC)+,(SP)        ;;RESTART AT BEGIN
 (1)   056452  001666              $PWRAD: .WORD  BEGIN                ;;RESTART ADDRESS
 (1)   056454  000002                         RTI
 (1)   056456  000000              $ILLUP: HALT                        ;;THE POWER UP SEQUENCE WAS STARTED
 (1)   056460  000776                         BR    .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
 (1)   056462  000000              $SAVR6: 0                           ;;PUT THE SP HERE
5469   056464  051200  051505  040524  PWRMSG: .ASCIZ  <200>/RESTARTING AFTER A POWER FAILURE  /
       056472  052122  047111  020107
```

```
              056500  043101  042524  020122
              056506  020101  047520  042527
              056514  020122  040506  046111
              056522  051125  020105  000040
      5470
      5471                               .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)   056530  105737  001157    $TYPE:  TSTB     $TPFLG          ;;IS THERE A TERMINAL?
      (1)   056534  100002                    BPL      1$              ;;BR IF YES
      (1)   056536  000000                    HALT                     ;;HALT HERE IF NO TERMINAL
      (1)   056540  000430                    BR       3$              ;;LEAVE
      (1)   056542  010046          : 1$:     MOV      R0,-(SP)        ;;SAVE R0
      (1)   056544  017600  000002            MOV      @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
      (1)   056550  122737  000001  001210    CMPB     #APTENV,$ENV    ;;RUNNING IN APT MODE
      (1)   056556  001011                    BNE      62$             ;;NO,GO CHECK FOR APT CONSOLE
      (1)   056560  132737  000100  001211    BITB     #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
      (1)   056566  001405                    BEQ      62$             ;;NO,GO CHECK FOR CONSOLE
      (1)   056570  010037  056600            MOV      R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
      (1)   056574  004737  057316            JSR      PC,$ATY3        ;;SPOOL MESSAGE TO APT
      (1)   056600  000000          61$:      .WORD    0               ;;MESSAGE ADDRESS
      (1)   056602  132737  000040  001211 62$: BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
      (1)   056610  001003                    BNE      60$             ;;YES,SKIP TYPE OUT
      (1)   056612  112046          2$:       MOVB     (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
      (1)   056614  001005                    BNE      4$              ;;BR IF IT ISN'T THE TERMINATOR
      (1)   056616  005726                    TST      (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
      (1)   056620  012600          60$:      MOV      (SP)+,R0        ;;RESTORE R0
      (1)   056622  062716  000002  3$:       ADD      #2,(SP)         ;;ADJUST RETURN PC
      (1)   056626  000002                    RTI                      ;;RETURN
      (1)   056630  122716  000011  4$:       CMPB     #HT,(SP)        ;;BRANCH IF <HT>
      (1)   056634  001430                    BEQ      8$
      (1)   056636  122716  000200            CMPB     #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
      (1)   056642  001006                    BNE      5$
      (1)   056644  005726                    TST      (SP)+           ;;POP  <CR><LF> EQUIV
      (1)   056646  104401                    TYPE                     ;;TYPE A CR AND LF
      (1)   056650  001165                    $CRLF
      (1)   056652  105037  057060            CLRB     $CHARCNT        ;;CLEAR CHARACTER COUNT
      (1)   056656  000755                    BR       2$              ;;GET NEXT CHARACTER
      (1)   056660  004737  056742  5$:       JSR      PC,$TYPEC       ;;GO TYPE THIS CHARACTER
      (1)   056664  123726  001156  6$:       CMPB     $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
      (1)   056670  001350                    BNE      2$              ;;IF NO GO GET NEXT CHAR.
      (1)   056672  013746  001154            MOV      $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
  (1)                                                               ;;AND THE NULL CHAR.
  (1)  056676  105366  000001          7$:    DECB    1(SP)          ;;DOES A NULL NEED TO BE TYPED?
  (1)  056702  002770                         BLT     6$             ;;BR IF NO--GO POP THE NULL OFF OF STACK
  (1)  056704  004737  056742                 JSR     PC,$TYPEC      ;;GO TYPE A NULL
  (1)  056710  105337  057060                 DECB    $CHARCNT       ;;DO NOT COUNT AS A COUNT
  (1)  056714  000770                         BR      7$             ;;LOOP
  (1)
  (1)                                  ;HORIZONTAL TAB PROCESSOR
  (1)
  (1)  056716  112716  000040          8$:    MOVB    #' ,(SP)       ;;REPLACE TAB WITH SPACE
  (1)  056722  004737  056742          9$:    JSR     PC,$TYPEC      ;;TYPE A SPACE
  (1)  056726  132737  000007  057060         BITB    #7,$CHARCNT    ;;BRANCH IF NOT AT
  (1)  056734  001372                         BNE     9$             ;;TAB STOP
  (1)  056736  005726                         TST     (SP)+          ;;POP SPACE OFF STACK
  (1)  056740  000724                         BR      2$             ;;GET NEXT CHARACTER
  (1)  056742                          $TYPEC:
  (1)  056742  105777  122176                 TSTB    a$TKS          ;;CHAR IN KYBD BUFFER?          ;MJD001
  (1)  056746  100022                         BPL     10$            ;;BR IF NOT                     ;MJD001
  (1)  056750  017746  122172                 MOV     a$TKB,-(SP)    ;;GET CHAR                      ;MJD001
  (1)  056754  042716  177600                 BIC     #177600,(SP)   ;;STRIP EXTRANEOUS BITS         ;MJD001
  (1)  056760  122716  000023                 CMPB    #$XOFF,(SP)    ;;WAS CHAR XOFF                 ;MJD001
  (1)  056764  001012                         BNE     102$           ;;BR IF NOT                     ;MJD001
  (1)  056766                          101$:                                                         ;MJD001
  (1)  056766  105777  122152                 TSTB    a$TKS          ;;WAIT FOR CHAR                 ;MJD001
  (1)  056772  100375                         BPL     101$                                          ;MJD001
  (1)  056774  117716  122146                 MOVB    a$TKB,(SP)     ;;GET CHAR                      ;MJD001
  (1)  057000  042716  177600                 BIC     #177600,(SP)   ;;STRIP IT                      ;MJD001
  (1)  057004  122716  000021                 CMPB    #$XON,(SP)     ;;WAS IT XON?                   ;MJD001
  (1)  057010  001366                         BNE     101$           ;;BR IF NOT                     ;MJD001
  (1)  057012                          102$:                                                         ;MJD001
  (1)  057012  005726                         TST     (SP)+          ;;FIX STACK                     ;MJD001
  (1)  057014                          10$:                                                          ;MJD001
  (1)  057014  105777  122130                 TSTB    a$TPS          ;;WAIT UNTIL PRINTER IS READY   ;MJD001
  (1)  057020  100375                         BPL     10$                                           ;MJD001
  (1)  057022  116677  000002  122122         MOVB    2(SP),a$TPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
  (1)  057030  122766  000015  000002         CMPB    #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
  (1)  057036  001003                         BNE     1$             ;;BRANCH IF NO
  (1)  057040  105037  057060                 CLRB    $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
  (1)  057044  000406                         BR      $TYPEX         ;;EXIT
  (1)  057046  122766  000012  000002  1$:    CMPB    #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
  (1)  057054  001402                         BEQ     $TYPEX         ;;BRANCH IF YES
  (1)  057056  105227                         INCB    (PC)+          ;;COUNT THE CHARACTER
  (1)  057060  000000                  $CHARCNT:.WORD  0             ;;CHARACTER COUNT STORAGE
  (1)  057062  000207                  $TYPEX: RTS     PC
  (1)
 5472                                  .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)
(1)   057064                          $TYPDS:
(3)   057064   010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)   057066   010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)   057070   010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)   057072   010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)   057074   010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)   057076   012746   020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)   057102   016605   000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)   057106   100004                         BPL     1$              ;;BR IF INPUT IS POS.
(1)   057110   005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)   057112   112766   000055   000001       MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)   057120   005000                  1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)   057122   012703   057300                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)   057126   112723   000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)   057132   005002                  2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
(1)   057134   016001   057270                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)   057140   160105                  3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)   057142   002402                         BLT     4$              ;;BR IF DONE
(1)   057144   005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)   057146   000774                         BR      3$
(1)   057150   060105                  4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)   057152   005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
(1)   057154   001002                         BNE     5$              ;;FALL THROUGH IF 0
(1)   057156   105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)   057160   100407                         BMI     7$              ;;BR IF YES
(1)   057162   106316                  5$:    ASLB    (SP)            ;;MSD?
(1)   057164   103003                         BCC     6$              ;;BR IF NO
(1)   057166   116663   000001   177777       MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)   057174   052702   000060         6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)   057200   052702   000040         7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)   057204   110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)   057206   005720                         TST     (R0)+           ;;JUST INCREMENTING
(1)   057210   020027   000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)   057214   002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
(1)   057216   003002                         BGT     8$              ;;GO TO EXIT
(1)   057220   010502                         MOV     R5,R2           ;;GET THE LSD
(1)   057222   000764                         BR      6$              ;;GO CHANGE TO ASCII
(1)   057224   105726                  8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)   057226   100003                         BPL     9$              ;;BR IF NO
(1)   057230   116663   177777   177776       MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)   057236   105013                  9$:    CLRB    (R3)            ;;SET THE TERMINATOR
(3)   057240   012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)   057242   012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)   057244   012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)   057246   012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)   057250   012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)   057252   104401   057300                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)   057256   016666   000002   000004       MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)   057264   012616                         MOV     (SP)+,(SP)
(1)   057266   000002                         RTI                     ;;RETURN TO USER
(1)   057270   023420                  $DTBL: 10000.
(1)   057272   001750                         1000.
(1)   057274   000144                         100.
(1)   057276   000012                         10.
(1)   057300   000004                  $DBLK: .BLKW   4
```

```
5473                                    .SBTTL  APT COMMUNICATIONS ROUTINE
  (1)
  (1)                                    ;;******************************************************************
  (2)
  (1)  057310  112737  000001  057554  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
  (1)  057316  112737  000001  057552  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
  (1)  057324  000403                           BR      $ATYC
  (1)  057326  112737  000001  057554  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
  (1)  057334                          $ATYC:
  (3)  057334  010046                           MOV     R0,-(SP)         ;;PUSH R0 ON STACK
  (3)  057336  010146                           MOV     R1,-(SP)         ;;PUSH R1 ON STACK
  (1)  057340  105737  057552                   TSTB    $MFLG            ;;SHOULD TYPE A MESSAGE?
  (1)  057344  001450                           BEQ     5$               ;;IF NOT:  BR
  (1)  057346  122737  000001  001210           CMPB    #APTENV,$ENV     ;;OPERATING UNDER APT?
  (1)  057354  001031                           BNE     3$               ;;IF NOT:  BR
  (1)  057356  132737  000100  001211           BITB    #APTSPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?
  (1)  057364  001425                           BEQ     3$               ;;IF NOT:  BR
  (1)  057366  017600  000004                   MOV     @4(SP),R0        ;;GET MESSAGE ADDR.
  (1)  057372  062766  000002  000004           ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
  (1)  057400  005737  001170          1$:      TST     $MSGTYPE         ;;SEE IF DONE W/ LAST XMISSION?
  (1)  057404  001375                           BNE     1$               ;;IF NOT:  WAIT
  (1)  057406  010037  001204                   MOV     R0,$MSGAD        ;;PUT ADDR IN MAILBOX
  (1)  057412  105720                  2$:      TSTB    (R0)+            ;;FIND END OF MESSAGE
  (1)  057414  001376                           BNE     2$
  (1)  057416  163700  001204                   SUB     $MSGAD,R0        ;;SUB START OF MESSAGE
  (1)  057422  006200                           ASR     R0               ;;GET MESSAGE LNGTH IN WORDS
  (1)  057424  010037  001206                   MOV     R0,$MSGLGT       ;;PUT LENGTH IN MAILBOX
  (1)  057430  012737  000004  001170           MOV     #4,$MSGTYPE      ;;TELL APT TO TAKE MSG.
  (1)  057436  000413                           BR      5$
  (1)  057440  017637  000004  057464  3$:      MOV     @4(SP),4$        ;;PUT MSG ADDR IN JSR LINKAGE
  (1)  057446  062766  000002  000004           ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
  (3)  057454  013746  177776                   MOV     177776,-(SP)     ;;PUSH 177776 ON STACK
  (1)  057460  004737  056530                   JSR     PC,$TYPE         ;;CALL TYPE MACRO
  (1)  057464  000000                  4$:      .WORD   0
  (1)  057466                          5$:
  (1)  057466  105737  057554          10$:     TSTB    $FFLG            ;;SHOULD REPORT FATAL ERROR?
  (1)  057472  001416                           BEQ     12$              ;;IF NOT:  BR
  (1)  057474  005737  001210                   TST     $ENV             ;;RUNNING UNDER APT?
  (1)  057500  001413                           BEQ     12$              ;;IF NOT:  BR
  (1)  057502  005737  001170          11$:     TST     $MSGTYPE         ;;FINISHED LAST MESSAGE?
  (1)  057506  001375                           BNE     11$              ;;IF NOT:  WAIT
  (1)  057510  017637  000004  001172           MOV     @4(SP),$FATAL    ;;GET ERROR #
  (1)  057516  062766  000002  000004           ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
  (1)  057524  005237  001170                   INC     $MSGTYPE         ;;TELL APT TO TAKE ERROR
  (1)  057530  105037  057554          12$:     CLRB    $FFLG            ;;CLEAR FATAL FLAG
  (1)  057534  105037  057553                   CLRB    $LFLG            ;;CLEAR LOG FLAG
  (1)  057540  105037  057552                   CLRB    $MFLG            ;;CLEAR MESSAGE FLAG
  (3)  057544  012601                           MOV     (SP)+,R1         ;;POP STACK INTO R1
  (3)  057546  012600                           MOV     (SP)+,R0         ;;POP STACK INTO R0
  (1)  057550  000207                           RTS     PC               ;;RETURN
  (1)  057552     000                  $MFLG:   .BYTE   0                ;;MESSG. FLAG
  (1)  057553     000                  $LFLG:   .BYTE   0                ;;LOG FLAG
  (1)  057554     000                  $FFLG:   .BYTE   0                ;;FATAL FLAG
  (1)          057556                           .EVEN
  (1)          000200                  APTSIZE=200
  (1)          000001                  APTENV=001
  (1)          000100                  APTSPOOL=100
```

```
    (1)           00004C                      APTCSUP=040
   5474                                        .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)  057556  017646  000000     $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
    (1)  057562  116637  000001  060001      MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
    (1)  057570  112637  060003           MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
    (1)  057574  062716  000002           ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
    (1)  057600  000406                   BR      $TYPON
    (1)  057602  112737  000001  060001  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
    (1)  057610  112737  000006  060003      MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
    (1)  057616  112737  000005  060000  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
    (1)  057624  010346                   MOV     R3,-(SP)        ;;SAVE R3
    (1)  057626  010446                   MOV     R4,-(SP)        ;;SAVE R4
    (1)  057630  010546                   MOV     R5,-(SP)        ;;SAVE R5
    (1)  057632  113704  060003           MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
    (1)  057636  005404                   NEG     R4
    (1)  057640  062704  000006           ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
    (1)  057644  110437  060002           MOVB    R4,$OMODE       ;;SAVE IT FOR USE
    (1)  057650  113704  060001           MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
    (1)  057654  016605  000012           MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
    (1)  057660  005003                   CLR     R3              ;;CLEAR THE OUTPUT WORD
    (1)  057662  006105             1$:   ROL     R5              ;;ROTATE MSB INTO ''C''
    (1)  057664  000404                   BR      3$              ;;GO DO MSB
    (1)  057666  006105             2$:   ROL     R5              ;;FORM THIS DIGIT
    (1)  057670  006105                   ROL     R5
    (1)  057672  006105                   ROL     R5
    (1)  057674  010503                   MOV     R5,R3
    (1)  057676  006103             3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
    (1)  057700  105337  060002           DECB    $OMODE          ;;TYPE THIS DIGIT?
    (1)  057704  100016                   BPL     7$              ;;BR IF NO
    (1)  057706  042703  177770           BIC     #177770,R3      ;;GET RID OF JUNK
    (1)  057712  001002                   BNE     4$              ;;TEST FOR 0
    (1)  057714  005704                   TST     R4              ;;SUPPRESS THIS 0?
```

```
 (1)   057716  001403                     BEQ     5$              ::BR IF YES
 (1)   057720  005204            4$:      INC     R4              ::DON'T SUPPRESS ANYMORE 0'S
 (1)   057722  052703  000060             BIS     #'0,R3          ::MAKE THIS DIGIT ASCII
 (1)   057726  052703  000040    5$:      BIS     #' ,R3          ::MAKE ASCII IF NOT ALREADY
 (1)   057732  110337  057776             MOVB    R3,8$           ::SAVE FOR TYPING
 (1)   057736  104401  057776             TYPE    ,8$             ::GO TYPE THIS DIGIT
 (1)   057742  105337  060000    7$:      DECB    $OCNT           ::COUNT BY 1
 (1)   057746  003347                     BGT     2$              ::BR IF MORE TO DO
 (1)   057750  002402                     BLT     6$              ::BR IF DONE
 (1)   057752  005204                     INC     R4              ::INSURE LAST DIGIT ISN'T A BLANK
 (1)   057754  000744                     BR      2$              ::GO DO THE LAST DIGIT
 (1)   057756  012605            6$:      MOV     (SP)+,R5        ::RESTORE R5
 (1)   057760  012604                     MOV     (SP)+,R4        ::RESTORE R4
 (1)   057762  012603                     MOV     (SP)+,R3        ::RESTORE R3
 (1)   057764  016666  000002  000004     MOV     2(SP),4(SP)     ::SET THE STACK FOR RETURNING
 (1)   057772  012616                     MOV     (SP)+,(SP)
 (1)   057774  000002                     RTI                     ::RETURN
 (1)   057776  000         8$:            .BYTE   0               ::STORAGE FOR ASCII DIGIT
 (1)   057777  000                        .BYTE   0               ::TERMINATOR FOR TYPE ROUTINE
 (1)   060000  000         $OCNT:         .BYTE   0               ::OCTAL DIGIT COUNTER
 (1)   060001  000         $OFILL:        .BYTE   0               ::ZERO FILL SWITCH
 (1)   060002  000000      $OMODE:        .WORD   0               ::NUMBER OF DIGITS TO TYPE
5475                                .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
 (1)
 (2)
 (1)                        ::****************************************************************
 (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)   060004  010146      $TYPBN: MOV     R1,-(SP)        ::SAVE R1 ON THE STACK
 (1)   060006  016601  000006             MOV     6(SP),R1        ::GET THE INPUT NUMBER
 (1)   060012  000261                     SEC                     ::SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
 (1)   060014  112737  000060  060056  1$:  MOVB  #'0,$BIN        ::SET CHARACTER TO AN ASCII "0".
 (1)   060022  006101                     ROL     R1              ::GET THIS BIT
 (1)   060024  001406                     BEQ     2$              ::DONE?
 (1)   060026  105537  060056             ADCB    $BIN            ::NO--SET THE CHARACTER EQUAL TO THIS BIT
 (1)   060032  104401  060056             TYPE    ,$BIN           ::GO TYPE THIS BIT
 (1)   060036  000241                     CLC                     ::CLEAR "C" SO CAN KEEP TRACK OF BITS
 (1)   060040  000765                     BR      1$              ::GO DO THE NEXT BIT
 (1)   060042  012601            2$:      MOV     (SP)+,R1        ::POP THE STACK INTO R1
 (1)   060044  016666  000002  000004     MOV     2(SP),4(SP)     ::ADJUST THE STACK
 (1)   060052  012616                     MOV     (SP)+,(SP)
 (1)   060054  000002                     RTI                     ::RETURN TO USER
 (1)   060056  000     000     $BIN:      .BYTE   0,0             ::STORAGE FOR ASCII CHAR. AND TERMINATOR
5476                                .SBTTL  TRAP DECODER
 (1)
 (2)
 (1)                        ::****************************************************************
 (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)   060060  010046      $TRAP:  MOV     R0,-(SP)        ::SAVE R0
 (1)   060062  016600  000002             MOV     2(SP),R0        ::GET TRAP ADDRESS
```

```
  (1)  060066  00574C                      TST      -(R0)            ;;BACKUP BY 2
  (1)  060070  111000                      MOVB     (R0),R0          ;;GET RIGHT BYTE OF TRAP
  (1)  060072  006300                      ASL      R0               ;;POSITION FOR INDEXING
  (1)  060074  016000   060114             MOV      $TRPAD(R0),R0    ;;INDEX TO TABLE
  (1)  060100  000200                      RTS      R0               ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
  (1)
  (1)  060102  011646             $TRAP2:  MOV      (SP),-(SP)       ;;MOVE THE PC DOWN
  (1)  060104  016666   000004  000002     MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
  (1)  060112  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)  060114  060102             $TRPAD:  .WORD    $TRAP2
  (3)  060116  056530                      $TYPE    ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)  060120  057602                      $TYPOC   ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)  060122  057556                      $TYPOS   ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)  060124  057616                      $TYPON   ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)  060126  057064                      $TYPDS   ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (3)  060130  060004                      $TYPBN   ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
  (1)
  (3)  060132  054416                      $GTSWR   ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
  (1)
  (3)  060134  054326                      $CKSWR   ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
  (3)  060136  054670                      $RDCHR   ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
  (3)  060140  054760                      $RDLIN   ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
  (3)  060142  055322                      $RDOCT   ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
 5477  060144  014274                      TEST     ;;CALL=CHECK    TRAP+14(104414)
 5478  060146  014266                      TESTIT   ;;CALL=CHKIT    TRAP+15(104415)
 5479  060150  037044                      DECTYP   ;;CALL=TYPDC    TRAP+16(104416)
 5480  060152  016324                      TPRMP    ;;CALL=TESTID   TRAP+17(104417)
 5481  060154  034362                      DELAY4   ;;CALL=DELY     TRAP+20(104420)
 5485
 5486                                ;BYTE TABLE CONTAINING A TYPE CODE FOR EACH CHANNEL
 5487                                ;0=NON EXISTANT CHANNEL, 1=SINGLE ENDED, 2=DIFFERENTIAL, 3=MNCAG, 4=MNCTP
 5488                                ;0XX=DONT TEST ANALOG VALUES, 2XX=TEST ANALOG VALUES
 5489  060156  000144             CHTABL:  .BLKW    100.               ;CHANNEL TYPE BUFFER
 5490
 5491  060466  000310             DIST:    .BLKW    200.               ;STATE-WIDTH DISTRIBUTION
 5492  061306  010000             BUFFER:  .BLKW    4096.              ;BUFFER AREA
 5493  101306  000000             BUFEND:  0                           ;LAST LOCATION USED BY PROGRAM
 5494          000001             .END
```

```
ABASE  = 171000         23#      59     104     106     108     110     114     115     116
ACDW1  = 000000         59
ACDW2  = 000000         59
ACPUOP= 000000          59
ADBUFF   001462        116#     561     577     588     598     616     627     642     647     665     680     688     705
                       718     804     847     850     853     856     899     919     930     950    1173*   1177*   1244
                      1274    1276    1277    1279    1624    1681    1778    1923    2060    2110    2142    2176    2207
                      2233    2282    2546    2717    4212*   4220*   4224*   4236*   4317*   4323    4329    4395    4423
                      4451    4695*   4709    5205
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    001460        115#     539*    847*    850*    853*    856*    906*    937*   1102*   1103*   1104*   1235*   1236*
                      1237*   1331*   1332*   1333*   1348*   1349*   1350*   1364*   1365*   1366*   1376*   1377*   1378*
                      1392*   1393*   1394*   1404*   1405*   1406*   1416*   1417*   1418*   1428*   1429*   1430*   1532*
                      1533*   1534*   1535*   1536*   1537*   1556    1587*   1588*   1589*   1619*   1642*   1643*   1644*
                      1676*   2050*   2056*   2094*   2102*   2137*   2170*   2171*   2172*   2226*   2227*   2228*   2266*
                      2267*   2268*   2365*   2366*   2367*   2414*   2415*   2416*   3686    3761*   4211*   4219*   4223*
                      4319*
ADTA     001422         96#     227*    246     748     793    2598    2620    2639
AENV   = 000000         59
AENVM  = 000000         59
AFATAL= 000000          59
AFIRST   022156        791    2507#   2531    2539    2577    2608
AGAIN    034560       4384#   4470
AGCHPA   024322       1369*   1381*   1397*   1409*   1421*   1433*   2966#
AGCHPB   024324       1370*   1382*   1398*   1410*   1422*   1434*   2967#
AGCHRA   024170       1367*   1379*   1395*   1407*   1419*   1431*   2922#
AGCHRB   024172       1368*   1380*   1396*   1408*   1420*   1432*   2923#
AGTA     001426         99#     229*    256    2602    2654
AGTASW   042772       2096*   2097*   5243#
AGTST    040352       2306*   2313*   2324*   2334*   2356*   2422*   2450*   3659*   5114#
AJGAIN   015756       1831    1850#
AJOFF    015704       1829    1836#
AMADR1= 000000          59
AMADR2= 000000          59
AMADR3= 000000          59
AMADR4= 000000          59
AMAMS1= 000000          59
AMAMS2= 000000          59
AMAMS3= 000000          59
```

Symbol	Value										
AMAMS4=	000000	59									
AMNCTP	026466	243	373	3519#							
AMSGAD=	000000	59									
AMSGLG=	000000	59									
AMSGTY=	000000	59									
AMTA	001424	97#	228*	251	2600	2625	2642				
AMTYP1=	000000	59									
AMTYP2=	000000	59									
AMTYP3=	000000	59									
AMTYP4=	000000	59									
APASS =	000000	59									
APRIOR=	000000	59									
APTCSU=	000040	5471	5473#								
APTENV=	000001	5466	5471	5473#							
APTSIZ=	000200	190	5473#								
APTSPO=	000100	5471	5473#								
ARESET	037030	189	299	4808#							
AROUND	035130	4412	4440	4469#							
ASKAG	023044	2635	2651#								
ASKAMB*	041737	267	5229#								
ASKC	026314	2627	2644	2656	2671	3479#	3492				
ASKDIF	022764	2616	2634#								
ASKDON	023174	2609	2611	2679	2681#						
ASKOOP	023162	2664	2678#								
ASKSE	022676	2615#	2632	2649	2661	2676					
ASKTA	003002	239	244	249	254	262	266	273	277	282#	369	3563
ASKTP	023106	2652	2663#								
ASKWHO	022654	2599	2601	2603	2605	2608#					
ASWREG=	000000	59									
ATESTN=	000000	59									
ATMSG	046544	4245	4249	5316#							
AUNIT =	000000	59									
AUSWR =	000000	59									
AVECT1=	000400	24#	59	105	107	109	111	117	118	119	120
AVECT2=	000000	59									
BADCAL	024700	2698*	2915	2959	3009*	3059#					
BADUNT	001616	164#	424*	5110	5454*						
BARFO	001434	101#	903	934	2501						
BASECH	001474	121#	2342*	2347	2348	2407*	2412	2413			
BASEND	001476	122#	2345*	2347*	2351	2410*	2412*	2418			
BASEXC	033252	329	4176#								
BEGIN	001666	51	181#	5114	5468						
BEGINA	020572	314	365	2316#							
BEGINC	015632	317	1821#								
BEGIND	021362	352	2401#								
BEGINF	016106	346	1875#								
BEGINL	003746	430#	2310	2320							
BEGINM	021022	355	2360#								
BEGINN	020676	343	2337#								
BEGINP	020212	320	2244#								
BEGINS	021522	358	2425#								
BEGINT	016400	349	1931#								
BEGINW	020636	326	340	2327#							
BEGINX	020006	361	2205#								
BEGIN2	001702	53	185#								
BEGIN3	001712	54	187#								

```
BEGL    020524          220     323    2305#
BEG2    001674           52     183#   1846    1870    1915    2001    2398    4153    5449
BITPNT  001576          156#    4313*  4316    4336    4337*
BIT0  = 000001           21#    611     655     660     669     670     671    1058    2139    2230
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#    5453
BIT09 = 001000           21#    5453    5466
BIT1  = 000002           21#    482     532     541    2589
BIT10 = 002000           21#    4668    4766
BIT11 = 004000           21#    693     4759    5453
BIT12 = 010000           21#    1889    1897
BIT13 = 020000           21#    1893    1897    2272    2290    2754    2816    5466
BIT14 = 040000           21#    187     237     368     487     490     638     640    1885    1889    1893    1897    3547
                        3665    4152    4867    5453
BIT15 = 100000           21#    185     414     518     548     558     638     640     659     668     801    2840    3171
                        3364    4191    4983
BIT2  = 000004           21#    508
BIT3  = 000010           21#    513     684    1620    1677    1881    2138    2229    2587
BIT4  = 000020           21#    504     783
BIT5  = 000040           21#    500     698     711     905     936
BIT6  = 000100           21#    300     496     568     604     611     617     631     656    5101
BIT7  = 000200           21#    556     617     659     698     711     725     737     783    1771    3511
BIT8  = 000400           21#    479     692     787     788
BIT9  = 001000           21#    101
BPTVEC= 000014           21#
BUFEND  101306          5493#
BUFFER  061306          2697    2719*  2732    2783    2931*  2932*  2933*  2934*  2935*  2936*  2946*  2947*  2987
                        2996    4362    4400*  4402*  4428*  4430*  4456*  4458*  4474    4502    4580    4592    4594
                        4616    4628    4629    4630    4652    4655    5492#
BUFF1   050663          4563    5364#
BUFF2   050711          4651    5374#
BUMPAD  033354          2311    2322    2332    4199#
BYPASS  033526          4214    4223#
B10     036760          4792#   4793
B11     037014          4802#   4803
CCHAN   046631          1821    1876    1934    2245    2361    5322#
CH      046477          2274    5312#
CHA     023176          1019*   1020*  1021    1050*  1051*  1052    1080*  1081*  1082    1225*  1226*  1227    1525*
                        2413*   2416   2418    2420*  2516*  2519*  2610   2682#  4355    4378
CHAN    046601          3018    4354    4687    5319#
CHANA   017754          2007*   2035   2066*   2074    2189#
CHANB   017756          2011*   2038   2069*   2080    2190#
CHANC   017760          2015*   2041   2191#   2192#
CHAND   017762          2019*   2044   2192#
CHANIS  013544          1322*   1323*  1324    1386    1463#
CHANL   001556          148#    583*    593*   1025*  1033    1035*  1056*  1091*  1092*  1104    1111    1113*  1174*
                        1185*   1229*  1230*   1237   1309*   1312*  1314   1333    1350    1366    1378    1394    1406
                        1418    1430    1455*   1456    1458   1460*   1492*  1596*  1610*  1650*  1664*  1696    1823*
                        1824    1879*  2146*   2147*   2182*  2183*   2210*  2224*  2228   2348*  2351   2353*  2364*
```

N 13

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41 PAGE 87-3
CVMNAC.P11   10-JUL-81 14:30   CROSS REFERENCE TABLE -- USER SYMBOLS   SEQ 0169

```
                        2708    3019    4324    4688    4694*   4696    4734*   5440    5445    5446
CHAPOS  044722          2005    5278#
CHB     023200          2524*   2555*   2593    2683#
CHBPOS  044745          2009    5279#
CHCPOS  044770          2013    5280#
CHDPOS  045013          2017    5281#
CHECK = 104414           526     552     559     569     578     662     703     716     802    5477#
CHKAGC  014146           868     901    1542#   1605
CHKGAN  014320           870    1583#
CHKIT = 104415           491     497     501     505     509     514     519    5478#
CHKTCC  014174           885     932    1548#   1659
CHKTCN  014534           887    1636#
CHPRIM  017766          2072*   2084*   2147    2171    2183    2194#
CHTABL  060156           750     777     988    1015    1024    1044    1055    1076    1092    1130    1146    1221    1230
                        1317    1322    1519    1542    1548    2525    2527    2596*   2597*   2615    2634    2651    2663
                        2678    3499*   3511*   5489#
CHXX    017764           865*    867     871*    872     882*    884     888*    889     896*    900     920*    921     927*
                         931     951*    952    1602    1639    1656    2071*   2077*   2082*   2083*   2088*   2172    2182
                        2193#
CH1     001544           143#   1135*   1136*   1137    1145*   1146*   1147    1149*   1150    1477*   1482*   1506    1508
                        1509*   1534    1936*   2033    2049*   2050    2071    2082    2093    2445*   4248    4250    4271
CH2     001546           144#   1150*   1151*   1478*   1483*   1492    1506    1509    1510*   1537    2033*   2036*   2039*
                        2042*   2137    2146    2446*   4244    4266
CK$WR = 104410          5453    5466    5476#
CLEAR1  034454          4364#   4366
CLEAR2  034520          4375#   4377
CLKBPR  001520           132#    700*
CLKCSR  001516           131#    701*
CLRCHS  017046          1937    2050#
CLRCHT  017040           847     850     853     856    2049#
COLECT  023262          2715#   2721
COMOD1  047544          2360    2385    5340#
COMPAR  036462           449     466     964     971     978     992    1001    1008    1029    1063    1107    1180    1189
                        1254    1283    1288    1812    4720#
COMPRA  036500          2184    4723#
CONTA1  017460          2034    2037    2040    2043    2135#
CONVCD  036346          1175    1178    4696#
CONVRT  036332           962     969     976     990     999    1006    4251    4693#
CONVR1  034604          4390    4392#
CONVR2  034720          4418    4420#
CONVR3  035034          4446    4448#
CONVTC  036340          1028    1062    1106    1493    1697    2373    4695#
CONV15  036642           447     464    4759#
CON4T   017634          2075    2078    2086    2089    2166#
CR    = 000015            21#   5471
CRLF  = 000200            21#    202    5471
CRWR    047151           757     765     798    1088    1166    1851    1911    1917    2026    2099    2104 .  2370    2379
                        5328#
C2      050621          4572    4657    5358#
C3      050627          4654    5359#
DAC     001570           153#   1702    1706    1738    1742    1752    1754    1860    1864    2377    2384*   2386    2389
                        4270    4275    4314*   4316*   4317    4336*
DASH    046453          4505    5310#
DDISP = 177570            21#     59     190
DECPNT  053610          4513*   4546*   5434#
DECPRT  026206          3446    3452#   3467
```

```
DECRPB   030200      3843    3851#
DECRPC   030202      3762*   3852#   4155*
DECRPT   030046      3806    3827#
DECTYP   037044      4815#   5479
DELAY    001572       154#   1803*   1807*   4382*   4469*
DELAY1   034602       207*    212*   4391#
DELAY2   034716       208*    213*   4419#
DELAY3   035032       209*    214*   4447#
DELAY4   034362      4343#   4344    5481
DELCLR   036234      4561    4573    4658    4668#
DELY   = 104420       212     213     214    5481#
DF1      053760        71      73      75      77      79      81      83      85      87      89      91      93      95
                     5447#
DH1      053165        71      85    5424#
DH12     053454        87      89      91      93    5430#
DH15     053520        95    5431#
DH2      053231        77    5425#
DH3      053315        73      75    5426#
DH5      053351        79    5427#
DH6      053375        81    5428#
DH7      053416        83    5429#
DIFLIN   034370      1526    2417    4349#
DISPLA   001142        59#    190*   2292*   5453*   5466*
DISPRE   000174        48#    190
DIST     060466      4367    4485*   4564    5491#
DIVI     024744      3095#   3300    3324
DOT      052123       308    5409#
DRVCSR   001522       133#
DRVDIR   001526       135#    847     850     853     856    2471
DRVDOR   001524       134#    787*    788*    847*    850*    853*    856*   1771*
DSWR   = 177570        21#     59     190
DT1      053614        71      85    5439#
DT12     053720        87      89      91      93    5445#
DT15     053736        95    5446#
DT2      053630        77    5440#
DT3      053650        73      75    5441#
DT5      053662        79    5442#
DT6      053674        81    5443#
DT7      053704        83    5444#
DUMMY    001554       147#   1186*   1696*   1824*   2368*   4266*   4271*   4319
DVMIEE   032720      3530    3727    3734    3739    3743    3747    3769    3775    3784    3793    4103#   4157
DVMTMP   031173      3729    3850    3894#
DVMVI    031121      3736    3845    3892#
DVMVO    031047      3786    3841    3890#
DWRFAD   041457       245     285    5225#
DWRFAG   041605       255    5227#
DWRFAM   041533       250    5226#
DWRFTP   041662       263    5228#
DWRMAP   042101       278    5231#
DWTSTP   030225       240     370    3865#
ECHAN    041220      2343    2408    5219#
EDGE     001574       155#   1494*   1698*   1735*   1739*   1857*   1861*   2374*   4280    4329
EDGFLG   034130      1504*   1707*   4282    4288#
EMTVEC=  000030        21#    190*
EM1      052213        71    5411#
EM10     052617        85    5418#
```

Symbol													
EM11	052663	87	5419#										
EM12	052737	89	5420#										
EM13	053005	91	5421#										
EM14	053053	93	5422#										
EM15	053117	95	5423#										
EM2	052251	73	5412#										
EM3	052311	75	5413#										
EM4	052352	77	5414#										
EM5	052413	79	5415#										
EM6	052474	81	5416#										
EM7	052572	83	5417#										
ENCOM	043447	4178	4186	5254#									
ENDTST	046423	1914	2000	2397	5308#								
ERCHKG	024524	2921	2965	3028#									
ERDIV	046152	2856	2907	3126	3310	3335	5300#						
ERIEE	040663	4151	5213#										
ERMSG	047505	455	472	1208	1723	1847	.1871	2394	3040	4155	4257	4519	4536	4550
		4609	5338#										
ERMUL	046220	3192	3207	3284	5302#								
EROVF	046274	2750	2813	2832	5304#								
ERRADR	015362	1691*	1759#	4265*									
ERRTOT	050555	5104	5356#										
ERRVEC=	000004	21#	190*	380*	399*	400*	3523	3524*	3546*	3664*	4216*	5453*	
ERSQR	046337	3401	3423	5306#									
EVER	001614	163#	306*	401	413*	414*	416	418	421				
EXCNOI	046103	3013	5298#										
EXTMSG	040276	5100	5101#										
EXTST	044421	795	5271#										
FIRST	001534	139#	4373*	4497	4499*								
FIXADR	033450	4200	4215#										
FIXONE	033454	216	423	1875	1931	2205	2337	2401	2425	4216#			
FOUND1	050001	405	5345#										
FOUND2	050024	408	5346#										
FPANL	017450	2118	2127#										
GADBUF	001510	128#	4746										
GAIN5	045253	1888	5285#										
GAIN5M	045321	1896	5287#										
GAIN50	045276	1892	5286#										
GANP5	045346	1330	5288#										
GAN5D	045434	1363	5290#										
GAN5P	045401	1347	5289#										
GAN5T	045470	1375	5291#										
GCHAN	043124	2211	2249	5249#									
GCONVT	036526	444	461	4733#									
GETDAT	036012	4619#	4627										
GHLF	045232	1884	1900	5284#									
GLD0	017770	1636*	1643	1669*	1671	2195#							
GLD1	017772	1637*	1647	1670*	2196#								
GLD2	017774	2197#											
GNS =	****** U	202	2369	2378	3429	3444	3447	5476	5477	5478	5479	5480	5481
GSTOL	030507	3708	3871#										
GSTOL0	030517	3704*	3873#										
GSTOL1	030521	3705*	3875#										
GSTOL2	030516	3703*	3872#										
GSTREG	001506	127#	4737*	4744*									
GS00	030530	3575	3624	3877#									

```
GS01     030570        3588    3633    3878#
GS10     030630        3601    3642    3879#
GS11     030670        3612    3651    3880#
GTSWR =  104407         202     335    5476#
GVECT    001512         129#   4741*
GVECT1   001514         130#   4742*
HAFMSG   047732        4547    5344#
HALF     035456        4539    4541#
HIDLY1   034664        4398    4407    4408#
HIDLY2   035000        4426    4435    4436#
HIDLY3   035114        4454    4463    4464#
HT    =  000011          21#   5471
HTOLO    032544        3700*   4073#
HTOL1    032546        3702*   4074#
HTOL2    032542        3698*   4072#
HUNS     053607        4824*   4837*   4838    4840*   4844*   4851    5433#
IBCSR    033226        3547    4113*   4126*   4132*   4135*   4144*   4150*   4163#
IBDAT    033230        4128*   4138    4141*   4143*   4164#
IBV0     033232        4103*   4165#
IBV0A    033234        4104*   4166#
IBV1     033236        4105*   4167#
IBV1A    033240        4106*   4168#
IBV2     033242        4107*   4169#
IBV2A    033244        4108*   4170#
IBV3     033246        4109*   4171#
IBV3A    033250        4110*   4172#
IDONTK   042166        2680    5235#
IGND     047112        1836    5327#
INIEE    031310        3531    3902#
INITVT   050731         201    5380#
INRNGE   035174        4481    4484#
IOTRD    040354          42     399    4216    5136#
IOTVEC=  000020          21#    190*
IVOLT    047211        1850    5329#
KBVECT   001500         123#
KWAD     001604         159#    275     709     894     925
KWBPR    001504         125#    713*    904*    935*
KWCSR    001502         124#    714*    907*    915*    938*    946*
K152     037412        4888#   4920    4926
K20      022140        2484    2497#   5444
K270     037402        4884#   4918
K275     037406        4886#   4928
K340     022142        2479    2498#   5444
K400     037414        4889#   4930
K516     037404        4885#   4922    4924
K60      022136        2474    2496#   5444
K72      037410        4887#   4916
LAST     035310        4495    4508#
LDSETG   014104        1152    1532#   2447
LDO1CH   014222        1554#   1583    1638    2520
LEDOFF   042704        2103    5241#
LEDON    042660        2095    5240#
LESS     036746        4787    4789#
LF    =  000012          21#   5471
LIMITE   037746        4856    4873    5022#
LIMITM   037706        4865    5004#
```

```
LIMITS   037644        4854    4858    4986#
LINEA    050534        4602    5355#
LOAD     036212        4653    4656    4660#
LOADY    036726        4567    4569    4663    4783#
LOAD0    036216        4661#   4665
LODLY1   034656        4396    4406#
LODLY2   034772        4424    4434#
LODLY3   035106        4452    4462#
LODTAB   026404        2550    2557    2565    2568    2572    3497#
LO2      036210        4615    4659#
LSB      046507        4243    5314#
LSBMSG   046444        4507    5309#
LTOL0    032426        3699*   3704    4046#
LTOL1    032430        3701*   3705    4047#
LTOL2    032424        3697*   3703    4045#
MADR     043355        4176    5252#
MASKNM   001620         165#   4201*   4203    4217*   5454    5455
MAX      001606         160#   4635*   4640    4642*   4646
MAXTST   036110        4638    4640#
MDASH    042162        2542    3481    5234#
MDIF     043060        2571    5246#
MESGD    050604        5109    5357#
MIN      001600         157#   4634*   4637.   4639*   4645
MINUS    042154        4285    4762    4817    5232#
MLSB     050644        1792    2388    3050    5361#
MLSBAT   050652        4685    5362#
MNCAD0   001436         104#   4208    4210
MNCTM    027226        3574    3587    3600    3611    3623    3632    3641    3650    3680#
MNCTM0   030206        3526    3681*   3694*   3695*   3724*   3725*   3731*   3732*   3755*   3756*   3768*   3781*   3855#
MNCTM1   030210        3722*   3856#
MOFSET   050631        1204    5360#
MPRMP    043100        2556    5247#
MSE      043040        2549    2581    5245#
MSG16    050162        4562    5350#
MSG18    050050        4650    5347#
MSG20    050110        4349    5349#
MSG21    050507        4579    5354#
MTCMP    043112        2562    5248#
MTEST    002566         224     233#    332
MTESTA   002634         238     242     244#    3668
MTEST0   003046         230     281     298#
MTEST1   003052         222     299#    336     375     4192
MULTI    025224        2738    3168#   3255    3263    3273    3359    3378    3407    3431
MVCT     043411        4183    5253#
MOLSB    047024        1855    5325#
NAR      035364        4522    4524#
NARMSG   047601        4526    5341#
NARROW   001532         138#   4372*   4489*   4524    4541
NBEXT    001550         145#   4199    4213*   4222*
NEXT1    034564        4385#   4468
NMBEXT   001552         146#    421*    422*   2339*   2403*   2427*   4222    5107
NOIJMP   013550        1296    1467#
NOIMSG   041252        1301    5220#
NOITST   012410        1295    1299#   2350
NOITS1   012470        1308    1311    1314#   1461    2354
NOLSTN   041023        3549    5216#
```

NONOIS	040640	2975	5212#											
NOTNAR	035230	4488	4494#											
NOTNEW	035674	4589	4594#											
NOTOK1	034650	4404#	4410											
NOTOK2	034764	4432#	4438											
NOTOK3	035100	4460#	4466											
NOTPOF	042025	271	5230#											
NOTSIE	040746	3522	3667	5215#										
NXTCMP	036074	4636#	4644											
NXTCVT	034250	4319#	4333											
NXTSTA	035636	4581#	4595											
NXTY1	035572	4566#	4571											
NXT8	036056	4632#	4649											
OFFSAV	015334	1250	1744*	1746#										
OFFSET	015252	1187	1735#	1841										
OFSET	041374	1199	5223#											
OKAYD	037070	4820	4822#											
OKAY1	034674	4401	4403	4405	4411#									
OKAY2	035010	4429	4431	4433	4439#									
OKAY3	035124	4457	4459	4461	4467#									
OKMSG	046620	453	470	1213	1719	2396	2977	3033	3045	3053	4237	4261	4523	4540
		4554	4613	5321#										
ONES	053612	3472*	3473	4822*	4829*	4830	4832*	4846*	5436#					
OUT	001612	162#	4370*	4482*	4528	4531	4534	4543						
OUTMSG	047677	4533	5343#											
PEAK	001562	150#												
PEAKN	024360	2945	2952	2986#										
PEAKNN	024240	2945#												
PERCNT	001610	161#	4308*	4309*	4310*	4311*	4312*	4334						
PIRQ =	177772	21#												
PIRQVE=	000240	21#												
PKNOI	046565	1342	1358	1446	2968	5318#								
PLUS	035656	4586	4588#											
PLUSR2	036734	4784	4786#											
POS	037056	4816	4819#											
POSITV	047272	1790	5331#											
POSNOI	036302	4682	4684#											
PRGAIN	026050	3049	3427#											
PRIAC	020002	2064*	2072	2200#										
PRIBD	020004	2067*	2084	2201#										
PRIME1	050763	298	5392#											
PRI4A	023202	1371	1383	1399	1411	1423	1435	2697#						
PRMS	024152	2900	2915#											
PR0 =	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#												
PSONOI	036314	2976	3051	4687#										
PSW =	177776	21#												
PUSHAG	042731	2098	5242#											
PWRMSG	056464	5468	5469#											

```
PWRVEC= 000024         21#      190*     5468*
QUEST   042156        374      3483     5233#
QUIET   001624        167#      307*     364*     1142     1195     1299     1328     1345     1361     1373     1389     1401     1413
                     1425      1448     1469     1523     1708     2973     3043     4240
RBEG    001720        182       186      188#
RBEG1   001724        184       189#
RDCHR = 104411       5449      5476#
RDCHXY  014472       1592      1607     1619#
RDLIN = 104412        286       309      758      766      799      1089     1167     1826     1837     1852     1912     1918     2027
                     2100      2105     2371     2380     3484     5451     5476#
RDOCT = 104413       1822      1877     1935     2209     2212     2215     2246     2250     2341     2344     2362     2406     2409
                     2429      2432     2436     2439     3552     3569     4179     4187     5476#
RDTCXY  014746       1646      1661     1676#
RDTEMP  031372       3728      3914#
RDVIN   031540       3735      3937#
RDVINL  032520       3744      4063#
RDVINP  032410       3740      4039#
RDVINU  032636       3748      4088#
RDVOUA  031642       3770      3957#
RDVOUB  031766       3776      3970#
RDVOUC  032104       3785      3980#
RDVOUD  032176       3794      3996#
RDVOUE  032310       4020#     4158
READ    035146       4475#     4509
RELACC  035622       4560      4577#
RESVEC= 000010         21#      203      204*     215*
RET     034126       4283      4287#
RETERR  014316       1572      1574#
RETURN  001650        174#     2258     2712     4320     4704     4741
RET1    001660        176#     4383
RET2    001664        175       177#
RMPTXT  042363       2404      5237#
RMS     001560        149#
RMSNOI  046551       1337      1353     1441     1714     2924     3048     5317#
RMSPEK  015010       1335      1340     1351     1356     1439     1444     1687#
RMS2    023470       2783#     2817
RW1     043224       2208      5250#
RW3     043250       2214      5251#
SADTST  044011        234      5265#
SAGTST  044072        235      1910     1933     5266#
SARSUB  034170       1699      1703     1736     1740     1858     1862     2375     2382     4267     4272     4308#
SCAL0   030730       3818      3881#
SCAL1   030740       3818      3882#
SCAL2   030751       3818      3883#
SCAL3   030762       3818      3884#
SCAL4   030773       3818      3819     3885#
SCAL5   031004       3819      3886#
SCAL6   031015       3819      3887#
SCAL7   031026       3819      3888#
SCAL8   031037       3819      3889#
SCHAN   041164       2340      2405     3551     5218#
SCLCAD  030034       3714      3822#
SCLDAC  030022       3710      3821#
SCLMSG  030166       3711*     3715*    3848#
SCLOCK  042774        274      5244#
SCLTXT  027776       3711      3818#
```

```
SCLXTX  030010       3715    3819#   5275#
SCM     044531       1883    2129    5275#
SDDIF   044235        225     756    5268#
SDMDIF  044353        226    5270#
SDMSE   044305        253    5269#
SDSE    044165        248     764    5267#
SEL     046717       1825    5323#
SELGT   047274       3564    5332#
SELGTA  047336       3568    5333#
SETASK  026446       2629    2646    2658    2673    3511#   3515
SETCH   046522       4247    5315#
SETINT  034132        608     635    4232    4290#
SETMSG  041302       1471    5221#
SETPMS  041432       5224#
SETTLE  013662       1154    1479    1484    1488#   1511    2449
SETTPM  041343       1144    5222#
SET1A   034002       1496    1498    4265#
SHIFT   034352       4335    4337#
SKIPST  001536        140#   4374*   4492*   4514    4517
SKPMSG  047522       4516    5339#
SLASH   046616       4604    5320#
SPACE   046502        448     465    1802    2294    3840    3844    3846    3849    5313#
SPRAC   017776       2065*   2073    2198#
SPRBD   020000       2068*   2085    2199#
SPREAD  001566        152#    584*    594*   2073*   2085*   4721*   4727    5440    5446
SQRR    024020       2873#   2905
SQR0    024726       2848*   2851    3070#   3254    3319    3369*   3389*
SQR1    024730       2849*   2852    3071#   3262    3298    3370*   3390*   3391*   3412*
SQR2    024732       2850*   2853    3072#   3271    3299    3371*   3392*   3413*   3414*
SQUARE  025654       2795    3352#
SRLOOP  017566       1084    1140    1160    2152#
SRM     044602       1901    2130    5276#
STACK = 001100         21#    190
STALL   022144        557     661     702     715     789    1772    2501#   4808    4810
STATE   046457       4500    5311#
STKLMT= 177774         21#
STREG   001456        114*    480*    481     523*    529*    530*    531     538*    540     548*    555*    558*    564*
                      565     574*    575     585*    586     595*    596     611*    612     614     615*    626*    638*
                      639     641*    646*    653*    655*    657     660*    669*    670*    671*    672     678     681*
                      684*    685*    686     692*    693     699*    706*    712*    719*    784*    801*    805*    847*
                      850*    853*    856*    898*    905*    908     910     914     929*    936*    939     941     945
                     1241*   1242    1269*   1271*   1272    1569*   1570    1620*   1621*   1622    1677*   1678*   1679
                     1774*   1775*   1776    1882*   1920*   1921    2057*   2058    2107*   2108    2138*   2139*   2140
                     2169*   2173*   2174    2229*   2230*   2231    2244*   2276*   2280*   2523*   2543*   2544    2588*
                     2589    2711*   2715*   3763*   4208*   4211    4212    4218*   4321*   4327*   4381*   4392*   4420*
                     4448*   4699*   4707*   5204*   5439    5440    5441    5443    5445    5446
SVM     044652        259    1905    2131    5277#
SWDIST  035540       4553    4559#
SWR     001140         59#    190     202     386    2152    2154    2156    2158    2257*   2259    2272    2290    3011
                     4668    5449*   5453    5466    5468*
SWREG   000176         49#    190     202    5449
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#    2156
SW09  = 001000          21#    2154
SW1   = 000002          21#
SW10  = 002000          21#
SW11  = 004000          21#
SW12  = 010000          21#    386
SW13  = 020000          21#    3011
SW14  = 040000          21#    2152
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   043651        3479    5260#
TCHANE  022652        2518    2606#
TCHANK  022200        2206    2308    2338    2402    2426    2516#
TCHANL  022212        2318    2329    2519#
TCHANM  022216        2517    2520#
TCHANN  022206        2518#   2522
TCHK    015524        1797#   1843    1867
TEMP    001540         141#    241     371    418*     446     463    1176   1179*   1249*   1282*   1287*   1494   1698
                      1738*   1742*   1743*  1744     1788    1860*   1864*  1865*   2374    2377*   2384    3565   4253
                      4404*   4409*   4432*  4437*    4460*   4465*   4703*  4709*   4712*   4713*   4714*   4715*  4722
                      4736*   4746*   4749*  4750*    4751*   4752*   5442
TEMPH   024704        2730*   2745*   2746*  2763*    2788    3061#   3314*  3328
TEMPL   024702        2729*   2744*   2762*  2787     3060#
TEMP1   001542         142#    183*    188*   197      221
TENS    053611        4823*   4833*   4834   4836*    4845*   5435#
TEST    014274        1570#   5477
TESTAD  003452         378#   2307    2316   2327
TESTID= 104417        1886    1890    1894   1898     1903    1907    5480#
TESTIT  014266        1569#   5478
TESTR2  037110        4826#   4831    4835   4839     4842
THOUS   053606        4825*   4841*   4843*  4847     4849    5432#
TKVEC = 000060          21#   5449*
TMCH    030223        3766*   3864#
TMFLAG  030214        3859#
TMGO    030221        3828*   3838*   3862#
TMG1    030220        3827*   3835*   3837*  3839     3861#
TMHEAD  030331        3804    3866#
TMOUT   030216        3860#
TM15    044513         460    5273#
TOFF    015502        1205    1788#   1842   1866
TOGOOD  024334        2752    2973#
TOOBAD  024332        2751    2814    2857   2908     2917    2961    2971#
TOOBIG  024416        2748    2812    2831   2833     2855    2906    3007#   3125    3191    3206    3283    3309   3334
                      3400    3422
```

```
TPAB      001432           100#     261*    268     1217
TPGN00    045525          1391     5292#
TPGN01    045564          1403     5293#
TPGN10    045623          1415     5294#
TPGN11    045662          1427     5295#
TPGVAL    030212          3684*    3712    3829    3831    3833    3858#
TPOFFK    037522          1233     4937#
TPOFFS    037562          1234     4956#
TPOF00    037416          4895#    4956
TPOF01    037420          4896#    4957
TPOF02    037422          4897#    4958
TPOF03    037424          4898#    4959
TPOF04    037426          4899#    4960
TPOF05    037430          4900#    4961
TPOF06    037432          4901#    4962
TPOF07    037434          4902#    4963
TPOF10    037436          4903#    4964
TPOF11    037440          4904#    4965
TPOF12    037442          4905#    4966
TPOF13    037444          4906#    4967
TPOF14    037446          4907#    4968
TPOF15    037450          4908#    4969
TPOF16    037452          4909#    4970
TPOF17    037454          4910#    4971
TPRMP     016324          1917#    5480
TPSCHN    027224          3553*    3555*   3556*   3669#   3685    3759
TPSWOF    045721          1086     5296#
TPSWON    046011          1164     5297#
TPTA      001430            99#     264    1073    1123    2604    2669
TPVAL     013546          1101*    1238*   1464#   5446
TPVALS    037456          1093     4915#
TPVEC =   000064            21#
TP15      044504           443     5272#
TRAPVE=   000034            21#     190*
TRFRO     040636          5143*    5208#
TRTO      040634          5136*    5137*   5138    5190    5196    5207#
TRTVEC=   000014            21#
TRY       034226          4315#    4338
TSETUP    016642          1949     1962    1975    1988    2004#
TSRT1     017124          2045     2046    2064#
TSTAD     043725          2477     5262#
TSTADM    043747          2482     5263#
TSTAG     043771          2487     5264#
TSTDAC    015336          1701     1705    1752#   4269    4274
TSTHLD    017300          1939     1941    1943    1945    2092#
TSTSDF    015364           724      728     736     740     760     768    1762#
TST1      003746           431#     433
TST10     004366           507#
TST11     004402           512#
TST12     004416           517#
TST13     004432           521#
TST14     004470           528#
TST15     004542           535      537#
TST16     004614           544      547#
TST17     004652           554#
TST2      004112           454      459#
```

```
TST20   004716        562#
TST21   004760        572#
TST22   005010        581#
TST23   005054        589     591#
TST24   005132        600     602#
TST25   005312        629#
TST26   005466        654#
TST27   005542        667#
TST3    004204        435     437     439     471     477#
TST30   005640        682#
TST31   005724        695#
TST32   006004        708#
TST33   006064        710     721#
TST34   006126        723     733#
TST35   006170        735     745#
TST36   006314        747     749     753     755     774#
TST37   006510        776     780     782     846#
TST4    004272        489#
TST40   006706        847     849#
TST41   007104        850     852#
TST42   007302        853     855#
TST43   007500        856     859#
TST44   007550        875#
TST45   007620        893#
TST46   010006        895     924#
TST47   010174        926     955#
TST5    004306        493#
TST50   010206        960#    961
TST51   010256        968#
TST52   010306        975#
TST53   010336        982#
TST54   010414        998#
TST55   010444        1005#
TST56   010474        1013#
TST57   010636        1017    1034    1042#
TST6    004336        499#
TST60   011000        1048    1070#
TST61   011260        1072    1074    1078    1112    1120#
TST62   011530        1122    1124    1161    1163    1171#
TST63   011616        1184#
TST64   011764        1198    1211    1215#
TST65   012236        1218    1220    1223    1266#
TST66   012364        1293#
TST67   013550        1468#
TST7    004352        503#
TST70   014024        1485    1518#
TST71   014072        1520    1522    1524    1528#
TST8    036122        4641    4643#
TXTP2   045036        258     1951    5282#
TXTP3   045134        1977    5283#
TYPBAD  035242        4483    4493    4497#
TYPBN = 104406        5111    5476#
TYPDC = 104416        1791    2387    4242    4506    4515    4525    4529    4532    4545    4601    4684    4776    5479#
TYPDEC  026132        3428    3442#
TYPDIG  026276        3435    3468    3471#
TYPDS = 104405        407     5100    5106    5186    5467    5476#
```

```
TYPE  = 104401        201    202    225    226    233    234    235    236    248    253    258    259    260
                      271    283    284    298    308    374    405    408    443    448    453    455    460
                      465    470    472    756    757    764    765    795    798   1086   1088   1144   1164
                     1166   1199   1204   1208   1213   1301   1306   1330   1347   1363   1375   1391   1403
                     1415   1427   1450   1471   1476   1713   1719   1723   1790   1792   1800   1802   1821
                     1825   1832   1836   1839   1847   1850   1851   1854   1855   1871   1876   1883   1884
                     1888   1892   1896   1900   1901   1905   1909   1910   1911   1914   1917   1932   1933
                     1934   2000   2005   2009   2013   2017   2023   2026   2095   2098   2099   2103   2104
                     2119   2208   2211   2214   2245   2249   2274   2294   2301   2340   2343   2360   2361
                     2369   2370   2378   2379   2385   2388   2394   2396   2397   2404   2405   2408   2428
                     2431   2435   2438   2444   2477   2482   2487   2533   2538   2542   2580   2595   2680
                     2975   2977   3013   3014   3018   3022   3047   3050   3052   3429   3444   3447   3473
                     3479   3481   3483   3491   3551   3568   3666   3706   3708   3804   3839   3840   3841
                     3844   3845   3846   3847   3849   3850   3851   4151   4176   4178   4183   4186   4243
                     4245   4247   4249   4260   4285   4349   4354   4358   4500   4505   4507   4516   4519
                     4523   4526   4530   4533   4536   4540   4547   4550   4554   4562   4563   4572   4579
                     4602   4604   4609   4613   4650   4651   4654   4657   4678   4685   4687   4762   4777
                     4817   4849   4851   5100   5104   5109   5112   5183   5187   5189   5191   5449   5466
                     5467   5468   5471   5472   5474   5475   5476#
TYPEDG  034066       4246   4280#  4686
TYPITA  017416       2006   2010   2014   2018   2116#
TYPOC = 104402       3017   4177   5449   5467   5476#
TYPON = 104404       5476#
TYPOS =.104403        797   1202   1304   1474   2275   2295   2536   2541   2579   3020   3480   3482   4185
                     4244   4248   4253   4281   4286   4352   4356   4504   4603   4606   4688   5188   5190
                     5476#
TYPOUT  037204       4827   4843#
TYPRP   036274       1716   4681#
TYPSET  033624       1505   4236#
UNEXP   001626        169#   494
UNITBD  001622        166#   797   1201   1303   1473   2535   4351   5439   5440   5441   5442   5443   5445
                     5446   5456*  5459*
UPCHAN  013500       1321   1384   1436   1449   1453#
VARLT1  037750       4861   5025#
VARLT2  040012       4863   5044#
VARLT3  040114       4869   5082#
VARLT4  040054       4871   5064#
VCM     037700       2392   5001#
VECTOR  001464        117#   494*   609*   621*  2258*  2712*  4210*  4221*  4225*  4226   4228   4230   4320*
                     4383*  4704*  5188   5192*  5196*  5197*  5198   5200
VECTR1  001466        118#   495*   610*   621    622*  2713*  4226*  4227*  4705*  5192   5194*  5198*  5199*
VECTR2  001470        119#   636*   648*  4228*  4229*  5193*  5200*  5201*  5202
VECTR3  001472        120#   637*   648    649*  4230*  4231*  5193   5195*  5202*  5203*
VLIN    037704       4607   5003#
VMULH   024714       2731*  2788*  2789*  2792*  2794*  2882*  2883*  2884*  2889   2902   2955*  2957*  3032
                     3065#  3096*  3120*  3155*  3157*  3176   3180*  3182*  3200   3226*  3228*  3253*  3304
                     3328*  3352   3357*  3376*  3442   3449*  3451*  3452   3457   3458*  3459*  3463*
VMULHS  024710       3032*  3034   3063#
VMULL   024712       2736*  2737*  2787*  2791*  2793*  2879*  2885*  2886   2901   2956*  2958*  3031   3064#
                     3097*  3141*  3146*  3154*  3156*  3179*  3181*  3198   3225*  3227*  3252*  3314   3353
                     3358*  3375*  3405*  3427*  3433*  3448*  3450*
VMULLS  024706       3031*  3037   3062#
VNP     037646       1447   4988#
VNPAG0  037652       1343   4990#
VNPAG1  037656       1359   4992#
VNR     037644       1442   4987#
```

```
VNRAG0   037650        1338    4989#
VNRAG1   037654        1354    4991#
VOLTS    044522        4777    5274#
VPAG2A   037664        1369    4995#
VPAG2B   037666        1370    4996#
VPAG3A   037674        1381    4999#
VPAG3B   037676        1382    5000#
VPTP0A   037712        1397    5008#
VPTP0B   037714        1398    5009#
VPTP1A   037722        1409    5012#
VPTP1B   037724        1410    5013#
VPTP2A   037732        1421    5016#
VPTP2B   037734        1422    5017#
VPTP3A   037742        1433    5020#
VPTP3B   037744        1434    5021#
VRAG2A   037660        1367    4993#
VRAG2B   037662        1368    4994#
VRAG3A   037670        1379    4997#
VRAG3B   037672        1380    4998#
VRTP0A   037706        1395    5006#
VRTP0B   037710        1396    5007#
VRTP1A   037716        1407    5010#
VRTP1B   037720        1408    5011#
VRTP2A   037726        1419    5014#
VRTP2B   037730        1420    5015#
VRTP3A   037736        1431    5018#
VRTP3B   037740        1432    5019#
VSET     037702        4238    4254    5002#
VTABLE   037622        1026    4975#
VTFLAG   001564        151#     279     339*    4559    4614
VTINIT   050750        4678    5387#
VTMSG    043454        2533    5183    5255#
VTMSG1   043531        5189    5257#
VTMSG2   043561        5191    5258#
VTMSG3   043500        5187    5256#
V0       037364        1285    1290    4877#
V1H      024720        2777*   2805*   2806*   2827*   2841*   2848    2874*   3067#   3099    3103*   3105*   3110*   3112*
                       3118    3132    3144
V1L      024716        2776*   2804*   2826*   2840*   2873*   3066#   3102*   3104*   3109*   3111*   3116    3130    3142
V10      037370        1814    4879#
V100D    037376         451     468    4882#
V12      037372         966     994    4880#
V2       037366        1182    4878#
V2H      024724        2779*   2809*   2810*   2829*   2843*   2850    3069#   3106    3117*   3118*   3131*   3132*   3140*
                       3143*   3144*   3299*   3320*
V2L      024722        2778*   2807*   2808*   2828*   2842*   2849    3068#   3116*   3130*   3139*   3142*   3298*   3319*
V326     037400         973     980    1003    1010    1031    1065    4883#
V50D     037374        1191    4881#
WFAD     022130         722    2466*   2476*   2492#
WFADJ    037260         217    4854#
WFADJ0   037330        4867#
WFAG     022134         438     775     847     850     853     856     983    2468*   2486*   2494#
WFAM     022132         734    1310    1480    2467*   2481*   2493#
WFCHK    022006        2305    2317    2328    2466#
WFTEST   001602         158#    181*    185*    187*    223     237     368*    403     434     696     746     785     986
                       1763    2469    3665*   4152*   4859    4867
```

```
WHICHU  055712        796     1200    1302    1472    2534    4233    4350    5184    5455#
WHICHV  055704        456      473    1209    1724    3023    3041    4258    4520    4537    4551    4610    5454#   5466
WIDE    001530        137#    1294*   1307    1315    1453    2349*   4371*   4496*   4527    4542
WIDMSG  047640       4530     5342#
WOWAGS  042274       2595     5236#
WRAP    010206        959#    2321    2331
WRAPX   010474        989     1012#
WRAPY   011530        985     1170#
XADJ    047010       1839     5324#
XBAR    023320       2731#    2755
XDEBUG= ****** U     4195     5482
XDIVI   025554       2875     3298#
XMULT   025404       2797     3245#
XMULO   024734       2804     3073#   3245*   3257*
XMUL1   024736       2806     3074#   3246*   3258*   3265*
XMUL2   024740       2808     3075#   3247*   3266*   3267*   3275*
XMUL3   024742       2810     3076#   3248*   3276*   3277*
YADJ    047255       1854     5330#
YESNO   043672        236     3491    5261#
$APTHD  001000         58#
$ASTAT= ****** U     5473
$ATYC   057334       5473#
$ATY1   057310       5473#
$ATY3   057316       5471     5473#
$ATY4   057326       5466     5473#
$AUTOB  001134         59#     202*    218     2509    5449
$BASE   001244         59#     378     4177    4182*   4218    4219    4220
$BDADR  001122         59#.
$BDDAT  001126         59#     378*    381     382*    481*    482*    483     531*    532*    534     540*    541*    543
                      588*    598*    599     614*    639*    672*    673*    674     688*    689*    847*    850*    853*
                      856*    914*    945*   1570*   1571    1594    1608    1624*   1625*   1648    1662    1681*   1682*
                     1778*   1779*   1780    1923*   1924*   1925    2142*   2143*   2144    2181*   2233*   2234*   2235
                     2473*   4722*   4724    5439    5440    5441    5444    5445    5446
$BELL   055254       5449#
$BIN    060056       5475#*
$CDW1   001250         59#
$CHARC  057060       5471#*
$CKSWR  054326       5449#    5476
$CMTAG  001100         59#     190
$CM3  = 000000         59#
$CNTLC  055260       5449#
$CNTLG  055272       5449#
$CNTLU  055265       5449#
$CPUOP  001216         59#
$CRLF   001165         59#     233     260     283    1306    1450    1476    1800    1909    1932    2301    2444    2538
                     3022    3762    3852    4358    5112    5449    5466    5467    5471
$DBLK   057300       5472#
$DEVCT  001200         59#
$DEVM   001246         59#
$DOAGN  040252       5100#
$DTBL   057270       5472#
$ENDAD  040242         56      202    5100#    5466
$ENDCT  040210        190     5100#
$ENDMG  040261       5100#
$ENULL  040256       5100#
$ENV    001210         59#     202     384     5466    5471    5473
```

B 15

CVMNA-C MNCAD/MNCAM/MNCAG/MNCTP DIAGNOSTIC   MACY11 30G(1063) 10-JUL-81 14:41   PAGE 87-17
CVMNAC.P11   10-JUL-81 14:30   CROSS REFERENCE TABLE -- USER SYMBOLS   SEQ 0183

```
$ENVM    001211        59#    190    5471    5473
$EOP     040154       398     412   2314    2325    2335    2357    2423    2451    3660    5100#
$EOPCT   040202       190*   5100#
$ERFLG   001103        59#    170*   5453*   5466*
$ERMAX   001115        59#    190*   5453*
$ERROR   055746       190    5466#
$ERRPC   001116        59#   5439    5440    5441    5442    5443    5444    5445    5446    5466*   5467
$ERRTB   001252        59#   5467
$ERRTY   056136      5466    5467#
$ERTTL   001112        59#    305*    457*    474*   1210*   1725*   3024*   3042*   3520*   3558*   4156*   4259*   4521*
                     4538*   4552*   4611*   5102    5105    5466*
$ESCAP   001162        59#    169*    172*    190*   5453*   5466
$ETABL   001210        59#
$ETEND   001252        58     59#
$FATAL   001172        59#   5473*
$FFLG    057554      5473#*
$FILLC   001156        59#   5471
$FILLS   001155        59#   5471
$GDADR   001120        59#
$GDDAT   001124        59#    479*    480     483     486*    487     490*    496*    500*    504*    508*    513*    518*
                      522*    533*    534     542*    543     549*    556*    563*    573*    582*    592*    599     616*
                      617*    640*    659*    668*    674     683*    698*    699     711*    712     783*    784     847*
                      850*    853*    856*    916*    917*    947*    948*   1569    1571    1593*   1594    1601*   1608
                     1647*   1648    1655*   1662    1773*   1780    1885*   1889*   1893*   1897*   1902*   1906*   1925
                     2074*   2080*   2135*   2144    2223*   2235    4720*   4723    5439    5440    5445    5446
$GET42   040232      5100#
$GTSWR   054416      5449#   5476
$HD    = 000000       20
$HIBTS   001000       58#
$HIOCT   055422      5451#*
$ICNT    001104        59#   5453*
$ILLUP   056456      5468#
$INTAG   001135        59#   5449*
$ITEMB   001114        59#   5466*   5467
$LF      001166        59#   5449    5466    5471
$LFLG    057553      5473#*
$LPADR   001106        59#    190*    431*    603*    630*    961*   1045*   5206    5453*
$LPERR   001110        59#    190*    433*    847*    850*    853*    856*    961*   1046*   1075*   1216*   1584*   1600*
                     1640*   1654*   2136*   2166*   5453*   5466
$MADR1   001222        59#
$MADR2   001226        59#
$MADR3   001232        59#
$MADR4   001236        59#
$MAIL    001170       58     59#    190     202    5453    5466    5471
$MAMS1   001220        59#
$MAMS2   001224        59#
$MAMS3   001230        59#
$MAMS4   001234        59#
$MBADR   001002       58#
$MFLG    057552      5473#*
$MNEW    055310      5449#
$MSGAD   001204        59#   5473*
$MSGLG   001206        59#   5473*
$MSGTY   001170        59#   5473*
$MSWR    055277      5449#
$MTYP1   001221        59#
```

```
$MTYP2   001225         59#
$MTYP3   001231         59#
$MTYP4   001235         59#
$MXCNT   055702       5453#
$NULL    001154         59#      5471
$NWTST=  000001        431#      459#     477#     489#     493#     499#     503#     507#     512#     517#     521#     528#     537#
                       547#      554#     562#     572#     581#     591#     602#     629#     654#     667#     682#     695#     708#
                       721#      733#     745#     774#     846#     849#     852#     855#     859#     875#     893#     924#     955#
                       960#      968#     975#     982#     998#    1005#    1013#    1042#    1070#    1120#    1171#    1184#    1215#
                      1266#     1293#    1468#    1518#    1528#
$OCNT    060000       5474#*
$OMODE   060002       5474#*
$OVER    055666       5453#
$PASS    001176         59#      190*     304*     436      754      781     1071     1121     1219     1521     2507     2521     5100*
                      5181      5453
$PASTM   001006         58#
$PWRAD   056452       5468#
$PWRDN   056312        190      5468#
$PWRMG   056446       5468#
$PWRUP   056364       5468#
$QUES    001164         59#     1832     5449     5466     5471
$RDCHR   054670       5449#     5476
$RDDEC=  ****** U     5476
$RDLIN   054760       5449#     5476
$RDOCT   055322       5451#     5476
$RDSZ =  000040       5449#
$RTNAD   040254       5100#
$R2A  =  ****** U     5476
$SAVRE=  ****** U     5476
$SAVR6   056462       5468#*
$SCOPE   055424        190      5453#
$SETUP=  000137         60#      190      202     5100     5449     5453     5466
$STUP =  177777         60#
$SVLAD   055632       5453#
$SVPC =  000220         56#
$SWR  =  165400          5#       20       22       59      190      431      459      477      489      493      499      503      507
                       512      517      521      528      537      547      554      562      572      581      591      602      629
                       654      667      682      695      708      721      733      745      774      846      849      852      855
                       859      875      893      924      955      960      968      975      982      998     1005     1013     1042
                      1070     1120     1171     1184     1215     1266     1293     1468     1518     1528     5100     5453     5466
                      5468
$SWREG   001212         59#      190
$SWRMK=  000C00         22      5453
$TESTN   001174         59#     5453*
$TIMES   001160         59#      190*     431*     459*     521*     547*     554*     562*     602*     629*     721*     733*     745*
                       774*     846*     849*     852*     855*     859*     875*     893*     924*     955*     960*     968*     975*
                       982*     998*    1005*    1013*    1042*    1070*    1120*    1171*    1184*    1215*    1266*    1293*    1468*
                      1518*     1528*    5100*    5453*
$TKB     001146         59#     1810     5449     5471
$TKCNT   053772       5449#*
$TKINT   054040        196      5449#
$TKQEN=  054040       5449#
$TKQIN   053774       5449#*
$TKQOU   053776       5449#*
$TKQSR   054000       5449#
$TKS     001144         59#      300*     525*     551*     568*     604*     631*     656*    1804*    1805     1811*    5101*    5449*
```

```
                        5471
$TKSRV  054110          5449#
$TN   = 000072             6#      20    431#    432     454    459#    471    477#    478    489#    493/    499#    503#
                         507#    512#    517#    521#    528#    535    537#    544     547#    554#    562#    572#    581#
                         589     591#    600     602#    629#    654#    667#    682#    695#    708#    710     721#    723
                         733#    735     745#    747     749     753     755     774#    776     780     782     846#    847
                         849#    850     852#    853     855#    856     859#    875#    893#    895     924#    926     955#
                         960#    961     968#    975#    982#    998#   1005#   1013#   1014    1017    1034    1042#   1048
                        1070#   1072    1074    1078    1112    1120#   1122    1124    1161    1163    1171#   1172    1184#
                        1198    1211    1215#   1218    1220    1223    1266#   1293#   1468#   1485    1518#   1520    1522
                        1524    1528#
$TPB    001152            59#   4794*   4804*   5471*
$TPFLG  001157            59#   5471
$TPS    001150            59#   4792    4802    5471
$TRAP   060060           190    5476#
$TRAP2  060102          5476#
$TRP  = 000021          5476#   5477#   5478#   5479#   5480#   5481#
$TRPAD  060114          5476#
$TSTM   001004           58#
$TSTNM  001102           59#     432*    478*    960*   1014*   1172*   2158    5100*   5144    5453*   5466
$TTYIN  055214          5449#
$TYPBN  060004          5475#   5476
$TYPDS  057064          5472#   5476
$TYPE   056530          192*    193*    194*    195*   5471#   5473    5476
$TYPEC  056742          5449    5471#
$TYPEX  057062          5471#
$TYPOC  057602          5474#   5476
$TYPON  057616          5474#   5476
$TYPOS  057556          5474#   5476
$UNIT   001202           59#    379*    383*    388     392     406     409     413     416     420*    5185    5442
$UNITM  001010           58#
$USWR   001214           59#
$VECT1  001240           59#    4184    4190*   4191*   4221
$VECT2  001242           59#
$XOFF = 000023          5471
$XON  = 000021          5471
$XTSTR  055436          5453#
$$GET4= 000000          5100#
$OFILL  060001          5474#*
$40CAT= ****** U        5453    5466
.     = 101310           28#     39     41#     43#     45#     47#     56#     57#     58#     59#     190     206    2459#
                        3891#   3893#   3895#   3896#   3897#   3911#   3933#   3934#   3954#   3967#   3968#   3977#   3978#
                        3993#   4018#   4036#   4059#   4086#   4097#   4117    5100    5449#   5453    5466    5467#   5468
                        5471    5472#   5473#   5489#   5491#   5492#
.$ASTA= ****** U        5473
.$X   = 001000           58#
```

```
ASCIZ       8#
BERROR   5463#    5466
COMMEN     21#
DUMWRN     62#
ENDCOM     21#
ERROR      21#     171      396      419      485      492      498      502      506      510      515      520      527      536      545
          553      560      570      579      590      601      618      643      663      676      691      704      717      803      847
          850      853      856      918      949      967      974      981      995     1004     1011     1032     1066     1110     1183
         1257     1286     1291     1597     1611     1651     1665     1782     1887     1891     1895     1899     1904     1908     2148
         2185     2237     2489     5147
ESCAPE     21#     169
GETPRI     21#
GETSWR     21#     202#
MULT       21#
NEWTST     21#     431      459      477      489      493      499      503      507      512      517      521      528      537      547
          554      562      572      581      591      602      629      654      667      682      695      708      721      733      745
          774      846      849      852      855      859      875      893      924      955      960      968      975      982      998
         1005     1013     1042     1070     1120     1171     1184     1215     1266     1293     1468     1518     1528
POP        21#    5451     5468     5472     5473
PUSH       21#    5451     5468     5472     5473
REPORT     21#
SCOPE      21#     459      477      489      493      499      503      507      512      517      521      528      537      547      554
          562      572      581      591      602      629      654      667      682      695      708      721      733      745      774
          846      849      852      855      859      875      893      924      955      968      975      982      998     1005     1013
         1042     1070     1120     1171     1184     1215     1266     1293     1468     1518     1528
SET        12#     961
SETPRI     21#    5449
SETTRA   5476#    5477     5478     5479     5480     5481
SETUP      21#     190
SKIP       21#     313      316      319      322      325      328      331      334      338      342      345      348      351      354
          357      360      363      367      435      439      452      454      469      471      488      535      544      589      600
          619      675      690      694      697      710      723      735      747      749      753      755      776      780      782
          786      790      794      847      850      853      856      895      926     1017     1034     1048     1072     1074     1078
         1112     1122     1124     1161     1163     1198     1207     1211     1218     1220     1223     1259     1311     1457     1459
         1485     1491     1507     1511     1513     1520     1522     1524     1595     1609     1649     1663     1695     1720     1722
         1753     1755     1781     1789     1798     1801     1806     1808     1809     1815     1829     1831     1833     1844     1845
         1868     1869     1926     2145     2236     2273     2470     2548     2575     2584     2590     3488     3490     3492     4669
         4671     4760     4768     4772     4839     4848     4860     4862     4866     5182
SLASH      21#    5146     5177
SPACE      21#
STARS      21#      56       58       59      431      459      477      489      493      499      503      507      512      517      521
          528      537      547      554      562      572      581      591      602      629      654      667      682      695      708
          721      733      745      774      846      849      852      855      859      875      893      924      955      960      968
          975      982      998     1005     1013     1042     1070     1120     1171     1184     1215     1266     1293     1468     1518
         1528     5100     5449     5451     5453     5466     5467     5468     5471     5472     5473     5474     5475     5476
SWRSU      21#     190#
TRMTRP   5476#
TYPBIN     21#
TYPDEC     21#    5100     5467
TYPNAM     21#     202
TYPNUM     21#
TYPOCS     21#     797     2275     2295     2541     2579     3480     3482     4185     4244     4248     4253     4281     4286     4504
         4603     4606     4688     5188     5190
TYPOCT     21#    4177     5449     5467
TYPTXT     21#    2369     2378     3429     3444     3447
XHOLDT    807#     847      850      853      856
```

```
$$CMRE      59#
$$CMTM      59#
$$ESCA      21#
$$NEWT      21#      431     459     477     489     493     499     503     507     512     517     521     528     537     547
            554      562     572     581     591     602     629     654     667     682     695     708     721     733     745
            774      846     849     852     855     859     875     893     924     955     960     968     975     982     998
           1005     1013    1042    1070    1120    1171    1184    1215    1266    1293    1468    1518    1528
$$SET     5476#    ·5477    5478    5479    5480    5481
$$SETM     190#
$$SKIP      21#      454     471     535     544     589     600     710     723     735     747     749     753     755     776
            780      782     847     850     853     856     895     926    1017    1034    1048    1072    1074    1078    1112
           1122     1124    1161    1163    1198    1211    1218    1220    1223    1485    1520    1522    1524
.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#     5473
.$CATC       7#
.$CMTA       7#       59
.$EOP        7#     5100
.$ERRO       7#     5466
.$ERRT       9#     5467
.$PARM       8#
.$POWE       8#     5468
.$RAND      10#
.$RDDE       7#
.$RDOC      10#     5451
.$READ       8#     5449
.$SAVE       8#
.$SCOP       8#     5453
.$SPAC       9#
.$SWDO       9#
.$TRAP       9#     5476
.$TYPB       8#     5475
.$TYPD      10#     5472
.$TYPE       9#     5471
.$TYPO       8#     5474


. ABS.  101310      000     CON   RW    ABS   GBL   D


ERRORS DETECTED:  0

CVMNAC,CVMNAC/CRF=CVMNAC
RUN-TIME: 30 22 2 SECONDS
RUN-TIME RATIO: 181/56=3.2
CORE USED: 28K  (55 PAGES)
```