# LPA/AD11-K

DIAGNOSTIC TEST
MD-11-DRLPK-A

EP-DRLPK-A-DL

FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

EOF1DRLPDRSEQ411 00010000 780223 IDENTPRPGT01 ECHDR1DRLPKASEQ 00010000 780223
SEQ 0001

Product Code: MAINDEC-11-DRLPK-A-D
Product Name: LPA/AD11-K DIAGNOSTIC TEST
Date: JAN 1978
Maintainer: Diagnostic Group

## 1.0 ABSTRACT

This diagnostic has two starting addresses: 200 for standard tolerances and 210 for tighter option test area tolerances.

This diagnostic tests the AD11K with or without a wraparound module (G5036).

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

W: The entire Wraparound test (requires G5036 module)
   a. Analog subtests
   b. Noise test
   c. Interchannel Settling test
   d. Differential Linearity and Relative Accuracy test

C: Calibration test only

N: Noise test only

S: Interchannel Settling only

L: Logic Subtests only

A: Auto test (requires G5036 module)

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

THIS PROGRAM IS A MODIFIED VERSION OF "MD-11-DZADL-B" IT WAS MODIFIED TO ENABLE THE OPERATOR TO CHECK OUT THE AD 11K OPTION WHEN IT IS ON THE LPA11-KX I/O BUS. NO RECABLING IS NEEDED. SOME TEST DONE IN THE ORIGINAL DIAGNOSTIC SUCH AS ARBITRATION TEST, WERE DELETED AS THEY COULD NOT BE CHECKED. IF THIS DIAGNOSTIC DOESN'T FIND A SUSPECTED PROBLEM, YOU MAY HAVE TO RUN "MD-11-DZADL-B" YOU SHOULD RUN "MD-11-DRLPA" BEFORE RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 10.

## 2.0 REQUIREMENTS

### 2.1 Equipment

PDP-11 family computer with 8K of memory
Teletype
AD11K Module
VT55 Terminal supported for graphic output
G5036 Wraparound Module

2.2 Storage

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

## 3.0 LOADING PROCEDURE

Procedure for loading normal binary tapes should be followed.

## 4.0 STARTING PROCEDURE

4.1 Control Switch Settings

Standard PDP-11 Format

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

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

## 5.0 OPERATING PROCEDURE

Start the diagnostic at 200 or 210. The program heading and the list of tests available, will be printed out followed by a message "Type the letter and carriage return for the desired test: . Then type the letter you want, according to the table listed and hit carriage return.

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

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

If "W" is typed, the program will type "xx AD11K's FOUND". Where xx is the number of AD11K's in octal. If the number is greater than 1, the test will be run successively on each AD11K. The program will run through the logic subtests, the Noise test on 8 edges, the Interchannel Settling test on 8 edges, and the Differential Linearity and Relative Accuracy test. A G5036 wraparound module is required. The program supports AD11K expansion beyond 16. channels. To run this test on a group of channels other than 0-17, load 20,40, or 60 into location BASECH (1336) for channels 20-37, 40-57, 60-77.

If "C" is typed, the program will run the calibration test and will loop on that test until the operator halts it. If a certain AD11K is to be tested, its status register address must be loaded into $BASE (1250), and its vector address must be loaded into the low byte of $VECT1 (1244) (the high byte containing the priority).

If "N" is typed, the program will run the Noise test tagged "BEGINN" and will loop on this test until the operator halts it. If a certain AD11K is to be tested its status register address must be loaded into $BASE (1250), and its vector address must be loaded into the low byte of $VECT1 (1244) (the high byte containing the priority).

If "S" is typed, the program will run the Interchannel Settling test tagged "BEGINS" and will loop on this test until the operator halts it. At the beginning of this test, the operator must respond to the statements asking for the "FROM" channel and the "TO" channel by typing in the channel value in octal and hitting carriage return. If a certain AD11K is to be tested its status register address must be loaded into $BASE (1250), and its vector address must be loaded into $VECT1 (1244) (the high byte containing the priority).

If "A" is typed, the program will execute the logic tests, analog tests, noise, settle and differential linearity. At the beginning of the test the program will type "XX AD11K's Found". Where XX IS THE NUMBER OF AD11K's in octal If the number is greater than 1, the test will be run successively on each AD11K. The program supports AD11K expansion beyond 16. channels. To run this test on a group of channels other than 0-17, load 20,40, or 60 into location BASECH (1336) for channels 20-37, 40-57, 60-77.

If "L" is typed, the program will execute the logic tests, printing "END PASS" when it has completed an entire pass. At the beginning of the test the program will type "XX AD11K's Found". Where XX is the number of AD11K's in octal If the number is greater than 1, the test will be run successively on each AD11K.

## 6.0 ERRORS

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

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

## 7.0 MISCELLANEOUS

### 7.1 Execution Time

Execution time for each of the tests is:

| | |
|---|---|
| Calibration: | 8 conversions/5 seconds @ 110 baud |
| Wraparound Test: | 17 minutes first pass; 35 minutes for successive passes |
| Settling Test: | 1 minute |
| Noise Test: | 1 minute |
| Logic Test: | 1 minute |
| Auto Test: | 18 minutes first pass, 36 minutes for successive passes |

### 7.2 Status Register and Vector Addresses and Priority

When testing more than one AD11K, the difference in addresses is presently 40 for bus address and vector address. These values are in VADR (bus address) (1332) and VVCT (vector address) (1334). The first AD11K's status register address must be in $BASE (1250), its vector address must be in the low byte of $VECT1 (1244), and the priority must be in the high byte of $VECT1.

### 7.3 AD11K Priority

If AD11K is set for a priority other than 6, the high byte of $VECT1 (1244) must be adjusted accordingly (the low byte containing the vector address). If more than one AD11K is being tested, all must be set at the same priority.

7.4 Switch Register

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

7.5 VT55 Graphic Output

The screen display may be halted for examination bysetting bit 12. And then just hit continue to complete the program's execution.

7.6 USER LINK TO I/O DEVICE

A SPECIAL USER LINK HAS BEEN PROVIDED IN ORDER FOR THE OPERATOR TO EXAMINE OR MODIFY LOCATIONS ON THE LPA11-KX I/O BUS. (NOTE: THIS CANNOT BE DONE DIRECTLY.)

PROCEDURE:

1) START THE PROCESSOR AT LOCATION $UTK:

2) THE DIALOG TO EXAMINE A LOCATION IS AS FOLLOWS:

```
E OR D              "E"
DEVICE ADDRS=       "OCTAL ADDRS"
XXXXXX
```

WHERE XXXXXX IS THE CONTENTS OF THE SPECIFIED LOC.

3) THE DIALOG TO MODIFY A LOCATION IS AS FOLLOWS:

```
E OR D              "D"
DATA=               "DATA TO BE DEPOSITED"
```

4) THE PROGRAM WILL STAY IN THIS LOOP UNTIL THE OPERATOR IS FINISHED. AT THIS TIME THE PROCESSOR SHOULD BE HALTED.

NOTE: THE OPERATORS RESPONSE IS ENCLOSED IN QUOTES.

## 8.0 RESTRICTIONS

8.1 A G5036 wraparound module must be present when running the auto test and the wraparound test.

Switch on G5036 must be in '0' position.

**********

The wraparound (G5036) module must be connected as follows:
AD11K TO BC08R CONNECTION A-A, VV-VV
BC08R TO G5036 CONNECTION "UPSIDE-DOWN" A-VV, VV-A
**********

SEQ 0009

## 9.0 PROGRAM DESCRIPTION

### 9.1 Logic Tests

These 14 logic subtests run sequentially without further operator intervention after he/she has typed in the number of AD11K's to be tested. Its purpose is to check that each of the mux bits can be loaded and properly read back; that initialize clears the external start enable bit, the done bit, the interrupt enable bit, the overflow bit, the error flag, and the A/D start bit. It also checks that the A/D done flag sets at end of conversion and clears when the converted value is read. It checks the interrupt logic and the correct setting of the error flag.

9.2 Calibration Test

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

9.3 Differential Linearity

This test is to determine if a change in the input voltage represents a similar change in the resulting converted binary value.

9.4 Settling Test

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

9.5 Noise Test

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

9.6 Analog Tests

These 11 subtests check the channels and their output.

10. LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

DIAGNOSTICS FOR THE LPA11 ARE WRITTEN AT THREE LEVELS: (1) TOTAL PDP-11 SYSTEM, (2) LPA11 SYSTEM; AND, (3) LPA11 OPTIONS.

LEVEL 1, IS DESIGNED TO ISOLATE A FAILURE TO tHE LPA11 SYSTEM. ALL OPTIONS ON THE PDP-11 ARE EXERCISED.

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN

THE USER RUNS DRLPA HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY "LOOK" ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

LEVEL THREE DIAGNOSTICS AID IN DETERMINING IF THE ERROR WAS IN FACT ON THE OPTION THE DRLPA SPECIFIED. THE USER MAY "LOOP" ON THE ERROR. WITHIN LEVEL THREE, THERE ARE TWO GROUPS OF DIAGNOSTICS. THE FIRST GROUP REQUIRES NO "EXTRA" WORK BY THE USER IN ORDER TO RUN. GROUP "A" DIAGNOSTICS DO NOT CHECK ARBITRATION, AND REQUIRE EXTRA TIME FOR EXECUTION. THE SECOND GROUP (GROUP "B") REQUIRES THAT THE USER RECONFIGURE THE PDP-11 SYSTEM. THIS RECONFIGURATION INVOLVES CABLING THE UNIBUS TO THE LPA'S I/O BUS.

THE DIAGNOSTIC FOR THE M8254 FALLS INTO THE GROUP "B" CATAGORY.

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:



| OPTION | GROUP | DIAG. # | DIAG. TITLE |
|---|---|---|---|
| LPA11-KX | LEVEL 2 | MD-11-DRLPA | LPA11-K SYSTEM DIAG. |
| M8254 | "B" | MD-11-DRLPN | M8254 (IPBM) DIAG. |
| AA11-K | A | MD-11-DRLPB | AA11-K DIAG. |
| | B | MD-11-DZAAC | AA11-K DIAG. |
| AR11 | A | MD-11-DRLPC | LPA/AR11 DIAG. #1 |
| | A | MD-11-DRLPD | LPA/AR11 DIAG. #2 |
| | A | MD-11-DRLPE | LPA/AR11 DIAG. #3 |
| | B | MD-11-DZARA | AR11 DIAG. #1 |
| | B | MD-11-DZARB | AR11 DIAG. #2 |
| | B | MD-11-DZARC | AR11 DIAG. #3 |
| DR11-K | A | MD-11-DRLPF | LPA/DR11-K DIAG. |
| | B | MD-11-DZDRG | DR11-K DIAG. |
| KW11-K | A | MD-11-DRLPG | LPA/KW11-K DIAG. |
| | B | MD-11-DZKWK | KW11-K DIAG. |
| LPS11 | A | MD-11-DRLPH | LPA/LPS11 DIAG. #1 |
| | A | MD-11-DRLPI | LPA/LPS11 DIAG. #2 |
| | A | MD-11-DRLPJ | LPA/LPS11 DIAG. #3 |
| | B | MD-11-DZLPC | LPS11 DIAG. #1 |
| | B | MD-11-DZLPD | LPS11 DIAG. #2 |
| | B | MD-11-DZLPI | LPS11 DIAG. #3 |
| AD11-K | A | MD-11-DRLPK | LPA/AD11-K DIAG. |
| | B | MD-11-DZADL | AD11-K DIAG. |
| M8200-YC | B | MD-11-DZLPL | LPA/M8200-YC BASIC MICRO-CPU R/W TEST |
| | B | MD-11-DZLPM | LPA/M8200-YC JMP+ROM READ TEST |

```
                        .REM    [

                                        LPA.MAC

WELCOME, THIS DIAGNOSTIC IS ONE IN A SERIES OF DIAGNOSTIC
DESIGNED IN ORDER TO AID YOU IN TESTING THE LPA-11XX OPTION.
I HOPE THAT YOU HAVE READ THE DOCuMENTATION SECTION OF THIS
DIAGNOSTIC.  IF YOU HAVE,YOU KNOW ABOUT ALL OF THE DIAGNOSTICS
THAT ARE AVAILIBLE FOR TESiING THE LPA SYSTEM.

                                        GOOD LUCK !

                        [
                        .GLOBL  DRLPX2
```

```
                        .TITLE  MAINDEC-11-DRLPKA
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY VERA BREUER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
```

```
                    020000          SW13=   20000
                    010000          SW12=   10000
                    004000          SW11=   4000
                    002000          SW10=   2000
                    001000          SW09=   1000
                    000400          SW08=   400
                    000200          SW07=   200
                    000100          SW06=   100
                    000040          SW05=   40
                    000020          SW04=   20
                    000010          SW03=   10
                    000004          SW02=   4
                    000002          SW01=   2
                    000001          SW00=   1
                                    .EQUIV  SW09,SW9
                                    .EQUIV  SW08,SW8
                                    .EQUIV  SW07,SW7
                                    .EQUIV  SW06,SW6
                                    .EQUIV  SW05,SW5
                                    .EQUIV  SW04,Sw4
                                    .EQUIV  SW03,SW3
                                    .EQUIV  SW02,SW2
                                    .EQUIV  SW01,SW1
                                    .EQUIV  SW00,SW0

                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                    100000          BIT15=  100000
                    040000          BIT14=  40000
                    020000          BIT13=  20000
                    010000          BIT12=  10000
                    004000          BIT11=  4000
                    002000          BIT10=  2000
                    001000          BIT09=  1000
                    000400          BIT08=  400
                    000200          BIT07=  200
                    000100          BIT06=  100
                    000040          BIT05=  40
                    000020          BIT04=  20
                    000010          BIT03=  10
                    000004          BIT02=  4
                    000002          BIT01=  2
                    000001          BIT00=  1
                                    .EQUIV  BIT09,BIT9
                                    .EQUIV  BIT08,BIT8
                                    .EQUIV  BIT07,BIT7
                                    .EQUIV  BIT06,BIT6
                                    .EQUIV  BIT05,BIT5
                                    .EQUIV  BIT04,BIT4
                                    .EQUIV  BIT03,BIT3
                                    .EQUIV  BIT02,BIT2
                                    .EQUIV  BIT01,BIT1
                                    .EQUIV  BIT00,BIT0

                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
```

```
                    000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                    000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                    000014          TBITVEC=14              ;;"T" BIT
                    000014          TRTVEC= 14              ;;TRACE TRAP
                    000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                    000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                    000024          PWRVEC= 24              ;;POWER FAIL
                    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                    000034          TRAPVEC=34              ;;"TRAP" TRAP
                    000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
                    000064          TPVEC=  64              ;;TTY PRINTER VECTOR
                    000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                                    .SBTTL  OPERATIONAL SWITCH SETTINGS
                                    ;*
                                    ;*      SWITCH                  USE
                                    ;*      ------          --------------------
                                    ;*        15            HALT ON ERROR
                                    ;*        14            LOOP ON TEST
                                    ;*        13            INHIBIT ERROR TYPEOUTS
                                    ;*        12            HALT FOR VT55 DISPLAY
                                    ;*        11            INHIBIT ITERATIONS
                                    ;*        10            BELL ON ERROR
                                    ;*         9            LOOP ON ERROR
                                    ;*         8            LOOP ON TEST IN SWR<7:0>
                    170400          ABASE=  170400
                    140340          AVECT1= 140340
                    000300          APRIOR= 300
                                    .SBTTL  TRAP CATCHER
                            000000          .=0
                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                            000174          .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                    .SBTTL  STARTING ADDRESS(ES)
000200  000137  001714                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  002404                  JMP     @#BEG2          ;RESTART ADDRESS
000210  000137  001722                  JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  012074                         $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOp
000052                                  .=52
000052  000000                         .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
001000                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
001000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                 $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                 $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  002260                 $TSTM:  .WORD   1200.   ;;RUN TIM OF LONGEST TEST
001006  000764                 $PASTM: .WORD   500.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  003244                 $UNITM: .WORD   1700.   ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                         .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

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

                                ;;**********************************************************************
                                .EVEN
001174                  001174  $MAIL:                          ;;APT MAILBOX
001174  000000                  $MSGTY: .WORD   AMSGTY           ;;MESSAGE TYPE CODE
001176  000000                  $FATAL: .WORD   AFATAL           ;;FATAL ERROR NUMBER
001200  000000                  $TESTN: .WORD   ATESTN           ;;TEST NUMBER
001202  000000                  $PASS:  .WORD   APASS            ;;PASS COUNT
001204  000000                  $DEVCT: .WORD   ADEVCT           ;;DEVICE COUNT
```

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

```
                                .SBTTL  ERROR POINTER TABLE

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

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

001256                          $ERRTB:


                                ;ITEM   1
001256  014253                          EM1                     ;STATUS REG. ERROR
001260  014413                          DH1                     ;ERRPC STREG EXPECTED ACTUAL
001262  014576                          DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
001264  014636                          DF1

                                ;ITEM   2
001266  014301                          EM2                     ;FAILED TO INTERRUPT
001270  014534                          DH3                     ;ERRPC STREG ACTUAL
001272  014626                          DT3                     ;$ERRPC, STREG,  $BDDAT
001274  014636                          DF1

                                ;ITEM   3
001276  014331                          EM3                     ;UNEXPECTED INTERRUPT
001300  014534                          DH3                     ;ERRPC STREG
001302  014626                          DT3                     ;$ERRPC, STREG
001304  014636                          DF1

                                ;ITEM   4
001306  014362                          EM4                     ;ERROR ON A/D CHANNEL
001310  014451                          DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
001312  014610                          DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,$READ,$BDDAT
001314  014636                          DF1
```

```
                                .SBTTL          MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
001316  170400          STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
001320  170402          ADBUFF: ABASE+2                 ;ADDRESS OF A/D BUFFER
001322  000300          BASEBR: APRIOR                  ;INTERRUPT PRIORITY LEVEL
001324  140342          VECTR1: AVECT1+2
001326  000040          VADR:   40                      ;INCREMENT FOR BUS ADDRESS
001330  000040          VVCT:   40                      ;INCREMENT FOR VECTOR ADDRESS
001332  000000          BASECH: 0                       ;BASE CHANNEL
001334  000060          KBVECT: 60
001336  000000          WIDE:   0                       ;NO. OF WIDE STATES
001340  000000          NARROW: 0                       ;NO. OF NARROW STATES
001342  000000          FIRST:  0
001344  000000          SKIPST: 0                       ;NO. OF SKIPPED STATES
001346  000000          TEMp:   0                       ;WORK AREA
001350  000000          CH1:    0                       ;FIRST CHANNEL
001352  000000          CH2:    0                       ;SECOND CHANNEL
001354  000000          NBEXT:  0                       ;NO. OF AD11K'S TO BE TESTED
001356  000000          NMBEXT: 0                       ;NO. OF AD11K'S TO BE TESTED
001360  000000          DUMMY:  0                       ;DUMMY CHANNEL
001362  000000          CHANL:  0                       ;CHANNEL VALUE
001364  000000          TADDR:  0                       ;TEST ADDRESS
001366  000000          RNA:    0                       ;RANDOM
001370  000000          RNB:    0                       ;NUMBER
001372  000000          RNC:    0                       ;VALUES
001374  000000          RMS:    0                       ;RMS NOISE VALUE
001376  000000          PEAK:   0                       ;PEAK NOISE VALUE
001400  000000          FLAG:   0                       ;VT55 FLAG
001402  000000          SPREAD: 0                       ;DEVIATION FROM THE NOMINAL
001404  000000          DAC:    0                       ;SAR VALUE
001406  000000          DELAY:  0                       ;TIME DELAY COUNTER
001410  000000          EDGE:   0                       ;EDGE VALUE
001412  000000          BITPNT: 0
001414  000000          MIN:    0                       ;MIN VALUE
001416  000000          WFTEST: 0                       ;OPTION TEST AREA FLAG
001420  000000          MAX:    0                       ;MAX VALUE
001422  000000          PERCNT: 0                       ;PERCENT FOR SAR ROUTINE
001424  000000          OUT:    0
001426  000000          MYTEMP: 0
001430  000000          EDINT:  0
001432  000000          $TEMp1: 0
001434  000000          $TEMP2: 0
                                ;
                                ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO MAY BE
                                ;                               CHANGED BY THE USER TO REFLECT
                                ;                               A DIFFERENT KMC-11 ADDR. THE
                                ;                               REST OF THE ADDRESSES WILL
                                ;                               BE CHANGED BY THE PROGRAM.
                                ;
001436                  LPCI:
001436  170460          KMADO:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.

001440                  LPMR:
001440  170461          KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
```

```
                                LPCO:
001442  170462                  KMAD2:  .WORD   170460+2                ;>PATCH         <;
001444                          LPSO:
001444  170463                  KMAD3:  .WORD   170460+3                ;>THIS AREA     <
001446                          LPADL:
001446  170464                  KMAD4:  .WORD   170460+4                ;
001450                          LPADH:
001450  170465                  KMAD5:  .WORD   170460+5                ;>DO NOT        <
001452                          LPMS1:
001452  170466                  KMAD6:  .WORD   170460+6                ;>PATCH         <
001454                          LPMS2:
001454  170467                  KMAD7:  .WORD   170460+7                ;>THIS AREA     <

001456  000340                  VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
001460  000344                  VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2

001462  000004                  VERSN:  .WORD   4               ;CURRENT VERSION NUMBER OF MICROCODE.

001464  000000                  .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
001466  000020                          .BLKW   16.             ;/BY INIT.

001526                          UNEXP:
001526  012737  001542  001162          MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
001534  005237  001103                  INC     $ERFLG
001540  104003                          ERROR   3
001542  005037  001162          1$:     CLR     $ESCAPE         ;RETURN ESCAPE TO NORMAL
001546  000002                          RTI                     ;UNEXPECTED INTERRUPT
```

```
                                        .SBTTL  CONTROL A AND C DECODERS
001550  010046                  ISERV:  MOV     R0,-(SP)            ;SAVE R0
001552  017700  177370                  MOV     @$TKB,R0            ;GET CHARACTER
001556  042700  177600                  BIC     #177600,R0
001562  120027  000003                  CMPB    R0,#3               ;IS IT ↑C?
001566  001010                          BNE     1$
001570  104401  012244                  TYPE    ,CMSG               ;ECHO CHARACTER
001574  012706  001100                  MOV     #STACK,SP
001600  004737  011362                  JSR     PC,RST              ;RESET & SET INTRPT. EN.
001604  000137  002404                  JMP     BEG2
001610  120027  000001          1$:     CMPB    R0,#1               ;IS IT ↑A?
001614  001010                          BNE     2$
001616  104401  012237                  TYPE    ,AMSG               ;ECHO CHARACTER
001622  012706  001100                  MOV     #STACK,SP
001626  004737  011362                  JSR     PC,RST              ;RESET & SET INTRPT. EN.
001632  000177  177526                  JMP     @TADDR              ;RETURN TO TEST
001636  120027  000007          2$:     CMPB    R0,#7               ;IS IT ↑G?
001642  001021                          BNE     NONE
001644  023727  001140  177570          CMP     SWR,#177570         ;HARDWARE SWREG?
001652  001415                          BEQ     NONE
001654  104401  012251                  TYPE    ,GMSG               ;ECHO CHARACTER
001660  017746  177254                  MOV     @SWR,-(SP)      ;;SAVE @SWR FOR TYPEOUT
                                        ;;TYPE SWREG
001664  104403                  TYPOS   ;;GO TYPE--OCTAL ASCII
001666     006                  .BYTE   6                       ;;TYPE 6 DIGITS
001667     001                  .BYTE   1                       ;;TYPE LEADING ZEROS
001670  104401  012431                  TYPE    ,SLASH
001674  104407                  RDOCT                           ;READ NEW VALUE
001676  012677  177236                  MOV     (SP)+,@SWR          ;LOAD NEW SWREG VALUE
001702  012600                  POPR0:  MOV     (SP)+,R0
001704  000002                  RETURN: RTI
001706  104401  012235          NONE:   TYPE    ,QUEST              ;TYPE "?"
001712  000773                          BR      POPR0
```

```
                                        .SBTTL          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
001714  005037  001416          BEGIN:  CLR     WFTEST
001720  000403                  BR      RBEG
001722  012737  000001  001416  BEGIN2: MOV     #1,WFTEST
001730                          RBEG:   ;RESET
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001730  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001734  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001736  022706  001140                  CMP     #$WR,R6 ;;DONE?
001742  001374                          BNE     .-6             ;;LOOP BACK IF NO
001744  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001750  012737  015234  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001756  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001764  012737  015512  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001772  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002000  012737  021276  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002006  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002014  012737  021352  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002022  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002030  013737  012054  012046          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
002036  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002042  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002046  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002054  012737  002054  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002062  012737  002062  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002070  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002074  012737  002130  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002102  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002110  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002116  022777  177777  177014          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002124  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002126  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002130  012716  002136          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002134  000002                          RTI
002136  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002144  012737  000174  001142          MOV     #DISPREG,DISPLAY
002152  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002156  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
002162  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002170  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002172  012737  001216  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002200                          67$:

                                ;
                                ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
                                ;
```

```
                                                MOV     R0,-(SP)
                                                MOV     R1,-(SP)
                                                MOV     KMADD,R0        ;GET KMC-11 ADDRESS.
                                                MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.

                                        68$:    INC     R0              ;UPDATE ADDR.
                                                MOV     R0,(1)+         ;WRITE ADDR.
                                                CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
                                                BNE     68$             ;NO - DO NEXT ADDR.
                                                CLR     .DVLS           ;CLR ADDR. LIST.
                                                MOV     (SP)+,R1
                                                MOV     (SP)+,R0
```

```
                                        CLR     FLAG            ;CLEAR VT55 FLAG
                                        TST     @#42                    ;IS IT CHAINED?
                                        BNE     REST1
                                .SBTTL          DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
                                        BIC     #100,@$TKS
                                        TYPE    ,CO             ;TYPE ASCIZ STRING
                                        JSR     PC,VTFLG        ;GET A CHARACTER
                                        CMP     R0,#33
                                        BNE     NOVT55          ;NO VT55 PRESENT
                                        JSR     PC,VTFLG        ;GET A CHARACTER
                                        CMP     R0,#57
                                        BNE     NOVT55          ;NO VT55 PRESENT
                                        JSR     PC,VTFLG        ;GET A CHARACTER
                                        CMP     R0,#103
                                        BEQ     VT55            ;VT55 IS PRESENT
                                        CMP     R0,#105
                                        BNE     NOVT55
                                VT55:   INC     FLAG
```

```
                                        .SBTTL      DIALOGUE TO DETERMINE WHICH TEST TO RUN
002332  104401  014034               NOVT55: TYPE    ,HEAD1
002336                               REST1:  ;RESET
002336  004737  005376                       JSR     PC,FIXONE               ;INITIALIZE ADDRESSES
002342  013700  001334                       MOV     KBVECT,R0
002346  012720  001550                       MOV     #ISERV,(R0)+
002352  012710  000340                       MOV     #340,(R0)
002356  012737  062341  001366               MOV     #62341,RNA              ;RANDOM NO, VARIABLES
002364  012737  142315  001370               MOV     #142315,RNB
002372  012737  127623  001372               MOV     #127623,RNC
002400  004737  011650                       JSR     PC,WFADJ                ;STANDARD OR OPTION TEST TOLERANCES?
002404                               BEG2:   ;RESET                  ;RESTART ADDRESS
002404  012706  001100                       MOV     #STACK,SP       ;RESET STACK IN CASE RESTARTED
002410  005737  000042                       TST     @#42                    ;IS IT CHAINED?
002414  001402                               BEQ     1$
002416  000137  005114                       JMP     BEGL                    ;GO TO LOGIC TESTS
002422  104401  013477               1$:     TYPE    ,MSG71
002426  104406                       TRYAG:  RDLIN
002430  052777  000100  176506               BIS     #100,@$TKS
002436  005037  177776                       CLR     PSW
002442  012600                               MOV     (SP)+,R0                ;READ ANSWER
002444  142710  000040                       BICB    #40,(R0)
002450  121027  000101                       CMPB    (R0),#'A                ;IS IT A?
002454  001002                               BNE     1$              ;;NO, TRY C
002456  000137  005156                       JMP     BEGINA                  ;GO TO AUTO TEST
002462  121027  000103               1$:     CMPB    (R0),#'C                ;IS IT C?
002466  001002                               BNE     2$              ;;NO, TRY L
002470  000137  004656                       JMP     BEGINC                  ;GO TO CALIBRATION TEST
002474  121027  000114               2$:     CMPB    (R0),#'L                ;IS IT L?
002500  001002                               BNE     3$              ;;NO, TRY N
002502  000137  005114                       JMP     BEGL                    ;GO TO LOGIC TESTS
002506  121027  000116               3$:     CMPB    (R0),#'N                ;IS IT N?
002512  001002                               BNE     4$              ;;NO, TRY S
002514  000137  005540                       JMP     BEGINN                  ;GO TO NOISE TEST
002520  121027  000123               4$:     CMPB    (R0),#'S                ;IS IT S?
002524  001002                               BNE     5$              ;;NO, TRY W
002526  000137  005610                       JMP     BEGINS                  ;GO TO SETTLE TEST
002532  121027  000127               5$:     CMPB    (R0),#'W                ;IS IT W?
002536  001002                               BNE     6$              ;;NO,TRY AGAIN
002540  000137  005250                       JMP     BEGINW                  ;GO TO WRAPAROUND TEST
002544  104401  012235               6$:     TYPE    ,QUEST
002550  000726                               BR      TRYAG                   ;WAIT FOR CHARACTER
002552  013737  001250  001126       TESTAD: MOV     $BASE,$BDDAT    ;SETUP TO TEST FOR AD11K'S
002560  005037  001464                       CLR     .DVLS
002564  005037  001466                       CLR     .DVLS+2
002570  005037  001354                       CLR     NBEXT                   ;CLEAR AD11K COUNTER
002574                               1$:                             ;ADDRESS AD11K

                                     ;*      MOV     $GDDAT,@$BDDAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG $BDDAT
002604  005737  017450                       TST     $AERR           ;DEVICE EXSIST? =0,YES
002610  001006                               BNE     2$              ;=1,NO.

002612  005237  001354               .       INC     NBEXT           ;INCREMENT AD11K COUNTER
002616  063737  001326  001126       .       ADD     VADR,$BDDAT     ;GET NEXT AD11K
```

```
   610  002624  000763                          BR      1$              ;;TRY NEXT AD11K
```

```
                                        2$:
611  002626
612  002626  013746  001354                    MOV     NBEXT,-(SP)     ;;SAVE NBEXT FOR TYPEOUT
613                                                                      ;;TYPE NUMBER OF AD11K'S
614  002632  104403                           TYPOS                     ;;GO TYPE--OCTAL ASCII
615  002634     002                           .BYTE   2                 ;;TYPE 2 DIGIT(S)
616  002635     000                           .BYTE   0                 ;;SUPPRESS LEADING ZEROS
617  002636  104401  013037                   TYPE    ,MSG50
618  002642  005337  001354                   DEC     NBEXT           ;ADJUST AD11K COUNT
619  002646  013737  001354  001356           MOV     NBEXT,NMBEXT    ;KEEP COUNT OF NUMBER
620  002654  000207                           RTS     PC
621
622  002656  005000                  VTFLG:   CLR     R0              ;TEST FOR PRESENCE
623  002660  105777  176260          1$:      TSTB    @$TKS           ;OF VT55
624  002664  100404                           BMI     2$              ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
625  002666  005300                           DEC     R0
626  002670  001373                           BNE     1$              ;;
627  002672  005726                           TST     (SP)+           ;POP A WORD OFF STACK
628  002674  000616                           BR      NOVT55          ;;NO VT55 PRESENT
629  002676  017700  176244          2$:      MOV     @$TKB,R0
630  002702  042700  177600                   BIC     #177600,R0              ;TEST VT55 CODE
631  002706  000207                           RTS     PC
```

```
                                BEGINL:
                                ;;*****************************************************************
                                ;*TEST 1         FLOAT A ONE THRU MULTIPLEXER BITS
                                ;;*****************************************************************
002710  012737  002710  001106  TST1:   MOV     #TST1,$LPADR
002716  012737  002710  001110          MOV     #TST1,$LPERR
002724  012737  000400  001124          MOV     #BIT8,$GDDAT            ;LOAD FIRST BIT
002732  004737  003400          2$:     JSR     PC,TESTIT
002736  104001                          ERROR   1                       ;FAILED TO LOAD + READ BIT
002740  006137  001124          1$:     ROL     $GDDAT                  ;GET NEXT BIT
002744  023727  001124  040000          CMP     $GDDAT,#BIT14           ;FINISHED?
002752  001367                          BNE     2$              ;;NO,GO TO NEXT TEST

                                ;;*****************************************************************
                                ;*TEST 2         LOAD AND READ BACK INTERRUPT ENABLE BIT6
                                ;;*****************************************************************
002754  000004                  TST2:   SCOPE
002756  012777  001526  176472          MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
002764  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD EXPECTED DATA
002772  004737  003400                  JSR     PC,TESTIT
002776  104001                          ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE

                                ;;*****************************************************************
                                ;*TEST 3         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
                                ;;*****************************************************************
003000  000004                  TST3:   SCOPE
003002  012737  000040  001124          MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
003010  004737  003400                  JSR     PC,TESTIT
003014  104001                          ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB

                                ;;*****************************************************************
                                ;*TEST 4         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
                                ;;*****************************************************************
003016  000004                  TST4:   SCOPE
003020  012737  000020  001124          MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
003026  004737  003400                  JSR     PC,TESTIT
003032  104001                          ERROR   1                       ;FAILED TO LOAD + READ EXT. START ENABLE

                                ;;*****************************************************************
                                ;*TEST 5         LOAD AND READ BACK ERROR FLAG BIT15
                                ;;*****************************************************************
003034  000004                  TST5:   SCOPE
003036  012737  100000  001124          MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
003044  004737  003400                  JSR     PC,TESTIT
003050  104001                          ERROR   1                       ;FAILED TO LOAD + READ ERROR FLAG
```

```
                                ;;*****************************************************************
                                ;*TESt 6         TEST  DONE FLAG  SETS AND BITO CLEARS ON END OF CONV.
                                ;;*****************************************************************
003052  000004                tST6:   SCOPE
003054  012700  001000                MOV     #BIT9,R0        ;STALL TIME COUNTER

                                ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
003070  005237  001426                INC     MYTEMP

                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003104  012737  000200  001124        MOV     #BIT7,$GDDAT    ;LOAD EXPECTED
003112  005300                1$:     DEC     R0              ;STALL
003114  001376                        BNE     1$              ;TIME

                                ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
003126  042737  100000  001426        BIC     #BIT15,MYTEMP

                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003144  004737  003410                JSR     PC,TESt
003150  104001                        ERROR   1               ;A/D DONe FLAG FAILED TO SET;BITO FAILED TO CLEAR

                                ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
003162  013700  001426                MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.

                                ;;*****************************************************************
                                ;*TEST 7         TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
                                ;;*****************************************************************
003166  000004                tST7:   SCOPE
003170  012737  000001  001426        MOV     #BITO,MYTEMP

                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003206  005037  001124                CLR     $GDDAT
003212                          1$:

                                ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
003222  105737  001426                TSTB    MYTEMP
003226  100371                        BPL     1$

                                ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
003240  013700  001426                MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
003244  004737  003410                JSR     PC,TESt
003250  104001                        ERROR   1               ;DONE FLAG FAILED TO CLEAR
```

```
                                        ;;*****************************************************************
                                        ;*TEST 10       TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
                                        ;;*****************************************************************
003252  000004                  T$T10:  SCOPE
003254  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003262  012737  000001  001426          MOV     #BIT0,MYTEMP
                                
003300                          1$:     MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
                                
                                        MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
003310  105737  001426                 TSTB    MYTEMP
003314  100371                         BPL     1$
003316  012737  100200  001124  2$:    MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
003324  012737  000001  001426          MOV     #BIT0,MYTEMP
                                
                                        MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003342  012700  001000                  MOV     #BIT9,R0        ;WAIT FOR 2ND
003346  005300                  3$:     DEC     R0              ;CONVERSION TO END
003350  001376                          BNE     3$
003352  004737  003410          4$:     JSR     PC,TEST
003356  104001                          ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
                                                                ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
                                
                                        MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
003370  013700  001426                  MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
```

```
                                                SCOPE
003374  000004                                  RTS     PC              ;RETURN TO TEST SECTION
003376  000207
                                        ;;SUBROUTINE FOR LOGIC TESTS;;
003400                                  TESTIT:
                                        ;*      MOV     $GDDAT,@STREG   ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG
003410                                  TEST:
                                        ;*      MOV     @STREG,$BDDAT   ;/READ DEVICE REG STREG,PUT DATA IN $BDDAT.
003420  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;COMPARE RESULTS
003426  001002                                  BNE     RETERR          ;;ERROR RETURN
003430  062716  000002                          ADD     #2,(SP)         ;BUMP RETURN ADDRESS TO gET AROUND ERROR
003434  000207                          RETERR: RTS     PC
```

```
                                .SBTTL          WRAPAROUND TEST SECTION
003436                          WRAP:
                                ;;*****************************************************************
                                ;*TEST 11        TEST CH14 GROUND
                                ;;*****************************************************************
000240                          TST11:  NOP
012737  000010  001160                  MOV     #10,$TIMES          ;;DO 10 ITERATIONS
012737  000011  001102                  MOV     #$TN-1,$TSTNM
012737  003776  001110                  MOV     #TST17,$LPERR
012737  003776  001106                  MOV     #TST17,$LPADR
004537  011072                          JSR     R5,CONVRT                   ;DO 8 CONVERSIONS
000014                                  14
004537  011314                          JSR     R5,COMPAR                   ;COMPARE RESULTS
004000                                  4000                        ;NOMINAL
011726                                  V50                         ;TOLERANCE
104004                                  ERROR   4                   ;ERROR-CH14 NOT GROUND-AD11K MUST BE IN
                                                                    ;SINGLE-ENDED CONFIGURATION,G5036 WRAPAROUND
                                                                    ;MODULE MUST BE PRESENT,CHECK CONNECTION A-VV,VV-A


                                ;;*****************************************************************
                                ;*TEST 12        TEST CONVERSION FROM EXT. START
                                ;;*****************************************************************
000004                          TST12:  SCOPE
012737  000010  001160                  MOV     #10,$TIMES          ;;DO 10 ITERATIONS
005737  001332                          TST     BASECH                      ;TESTING AN AM?
001044                                  BNE     TST13               ;;YES, GOTO NEXT TEST
012737  000020  001426                  MOV     #BIT4,MYTEMP

                                ;*      MOV     MYTEMP,@STREG       ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
012700  001000                  ;*      MOV     #BIT9,R0                    ;TIME DELAY COUNTER
012737  000220  001124                  MOV     #BIT7!BIT4,$GDDAT           ;LOAD EXPECTED
012737  000200  001426                  MOV     #200,MYTEMP

                                ;*      MOV     MYTEMP,@ADBUFF      ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
                                                                            ;WRAPAROUND MODULE PRESENT
005300                          1$:     DEC     R0
001376                                  BNE     1$
004737  003410                          JSR     PC,TEST
104001                                  ERROR   1                           ;FAILED TO DO CONVERSION FROM EXT. START

                                ;*      MOV     @ADBUFF,MYTEMP      ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
013700  001426                  ;*      MOV     MYTEMP,R0           ;/PUT CONVERTED VALUE IN R0.
005037  001426                          CLR     MYTEMP

                                ;*      MOV     MYTEMP,@STREG       ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG


                                ;;*****************************************************************
                                ;*TEST 13        TEST CH0 GROUND
                                ;;*****************************************************************
000004                          TST13:  SCOPE
012737  000010  001160                  MOV     #10,$TIMES          ;;DO 10 ITERATIONS
004537  011072                          JSR     R5,CONVRT                   ;CONVERT 8 TIMES
```

```
   815  003652  000000                    0
   816  003654  004537  011314            JSR     R5,COMPAR       ;COMPARE RESULTS
   817  003660  004000                    4000                    ;NOMINAL
   818  003662  011720                    V1                      ;TOLERANCE
   819  003664  104004                    ERROR   4               ;ERROR ON A/D CHANNEL
```

```
                                ;;*****************************************************************
                                ;*TEST 14        TEST CH1 GROUND
                                ;;*****************************************************************
003666  000004                  TST14:  SCOPE
003670  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003676  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
003702  000001                          1                               ;CHANNEL 1
003704  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
003710  004000                          4000                            ;NOMINAL
003712  011724                          V10                             ;TOLERANCE
003714  104004                          ERROR   4                       ;ERROR ON A/D CHANNEL

                                ;;*****************************************************************
                                ;*TEST 15        TEST CH2 +1 VOLT
                                ;;*****************************************************************
003716  000004                  TST15:  SCOPE
003720  012737  000010  001160          MOV     #10,$TImES      ;;DO 10 ITERATIONS
003726  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
003732  000002                          2                               ;CHANNEL 2
003734  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
003740  004632                          4632                            ;NOMINAL
003742  011726                          V50                             ;TOLERANCE
003744  104004                          ERROR   4                       ;ERROR ON A/D CHANNEL
                                                                        ;AD11K MUST BE SET UP FOR +OR- 5V OR +OR- 5.12V

                                ;;*****************************************************************
                                ;*TEST 16        TEST CH3 +2.5 VOLTS
                                ;;*****************************************************************
003746  000004                  TST16:  SCOPE
003750  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003756  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
003762  000003                          3                               ;CHANNEL 3
003764  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
003770  006000                          6000                            ;NOMINAL
003772  011734                          V240                            ;TOLERANCE
003774  104004                          ERROR   4                       ;ERROR ON A/D CHANNEL

                                ;;*****************************************************************
                                ;*TEST 17        TEST CH4 -2.5 VOLTS
                                ;;*****************************************************************
003776  000004                  TST17:  SCOPE
004000  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004006  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004012  000004                          4                               ;CHANNEL 4
004014  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004020  002000                          2000                            ;NOMINAL
004022  011734                          V240                            ;TOLERANCE
004024  104004                          ERROR   4
```

```
                                ;;*****************************************************************
                                ;*TEST 20       TEST VERNIER OFFSET DAC ON CH12
                                ;;*****************************************************************
004026  000004                  TST20:  SCOPE
004030  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004036  005037  001426                  CLR     MYTEMP

                                ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
004052  004737  004646                  JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
004056  004537  011072                  JSR     R5,CONVRT               ;CONV. CH12, DIRECT VERNIER DAC
004062  000012                          12
004064  013704  001346                  MOV     TEMP,R4                 ;SAVE VALUE IN R4
004070  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004074  002376                          2376                            ;WITH -1.875 VOLTS
004076  011732                          V115                            ;TOLERANCE OF 10%
004100  104004                          ERROR   4
004102  005037  001420                  CLR     MAX
004106  012702  000001                  MOV     #1,R2
004112  010237  001426          1$:     MOV     R2,MYTEMP               ;SET UP NEXT VERNIER DAC VALUE

                                ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
004126  004737  004646                  JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
004132  004537  011072                  JSR     R5,CONVRT               ;CONVERT IT
004136  000012                          12
004140  005737  001420                  TST     MAX
004144  001010                          BNE     2$
004146  023727  001346  004000          CMP     TEMP,#4000
004154  002404                          BLT     2$
004156  005237  001420                  INC     MAX
004162  010237  001414                  MOV     R2,MIN
004166  020227  000200          2$:     CMP     R2,#200
004172  001003                          BNE     3$
004174  013737  001346  004266          MOV     TEMP,4$
004202  013703  001346          3$:     MOV     TEMP,R3                 ;SAVE VALUE
004206  160437  001346                  SUB     R4,TEMP                 ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
004212  010304                          MOV     R3,R4                   ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
004214  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004220  000006                          6                               ;WITH 15 MILLIVOLTS(1 DAC LSB)
004222  011736                          V5
004224  104004                          ERROR   4
004226  005202                          INC     R2
004230  020227  000400                  CMP     R2,#400                 ;DONE?
004234  001326                          BNE     1$                      ;NO-DO NEXT VERNIER DAC VALUE
004236  004737  020422                  JSR     PC,$RESET
004242  052777  000100  174674          BIS     #100,@$TKS
004250  004737  004646                  JSR     PC,DAWAIT               ;LET DAC SETTLE
004254  004537  011072                  JSR     R5,CONVRT               ;CONVERT IT
004260  000012                          12
004262  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004266  000000                  4$:     0
004270  011722                          V2
004272  104004                          ERROR   4
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21        TEST CH13 +2.5 VOLTS
                                        ;;*****************************************************************
004274  000004                  TST21:  SCOPE
004276  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004304  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004310  000013                          13
004312  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004316  006000                          6000                    ;NOMINAL
004320  011730                          V144                    ;TOLERANCE
004322  104004                          ERROR   4
                                        ;;*****************************************************************
                                        ;*TEST 22        TEST CH17 +4V
                                        ;;*****************************************************************
004324  000004                  TST22:  SCOPE
004326  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004334  004537  011072                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004340  000017                          17                      ;CHANNEL 17
004342  004537  011314                  JSR     R5,COMPAR               ;COMPARE RESULTS
004346  007146                          7146                    ;NOMINAL
004350  011734                          V240                    ;TOLERANCE
004352  104004                          ERROR   4               ;ERROR ON A/D CHANNEL
```

```
                                ;;*****************************************************************
                                ;*TEST 23        OFFSET ON CHO
                                ;;*****************************************************************
                        TST23:  SCOPE
004354  000004                 
004356  012737  000001  001160          MOV     #1,$TIMES               ;;DO 1 ITERATION
004364  013737  001332  001362          MOV     BASECH,CHANL                    ;LOAD CHANNEL
004372  013737  001332  001360          MOV     BASECH,DUMMY                    ;LOAD DUMmY
004400  012737  004001  001410          MOV     #4001,EDGE
004406  004537  006452                  JSR     R5,SARSUB
004412  000062                          50.
004414  013737  001404  001346          MOV     DAC,TEMP
004422  004537  006452                  JSR     R5,SARSUB
004426  000062                          50.
004430  063737  001404  001346          ADD     DAC,TEMP
004436  162737  000062  001346          SUB     #62,TEMP
004444  013700  001414                  MOV     MIN,R0
004450  006300                          ASL     R0
004452  160037  001346                  SUB     R0,TEMP
004456  104401  013703                  TYPE    ,MOFSET                 ;TYPE ASCIZ STRING
004462  013702  001346                  MOV     TEMP,R2
004466  004737  011504                  JSR     PC,DECTYP
004472  104401  013716                  TYPE    ,MLSB                   ;TYPE ASCIZ STRING
004476  004537  011314                  JSR     R5,COMPAR               ;IS RESULT WITHIN LIMITS?
004502  000000                          0
004504  011740                          VSOD
004506  000401                          BR      OFFERR                  ;NO-ERROR
004510  000403                          BR      OFFOK                   ;YES-OK
004512  104401  012505          OFFERR: TYPE    ,ERMSG
004516  000402                          BR      TST24                   ;;GO TO NEXT TEST
004520  104401  012474          OFFOK:  TYPE    ,OKMSG
```

```
                                ;;*****************************************************************
                                ;*TEST 24        NOISE TEST ON 8 EDGES
                                ;;*****************************************************************
                        TST24:  SCOPE
004524  000004
004526  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004534  012737  000116  001346          MOV     #116,TEMP               ;DAC VALUE
004542  004537  010664                  JSR     R5,NOI8                 ;NOISE AT -FULL SCALE
004546  000015                          15
004550  004537  010664                  JSR     R5,NOI8                 ;NOISE AT MID-RANGE
004554  000007                          7
004556  004537  010664                  JSR     R5,NOI8                 ;NOISE AT +FULL SCALE
004562  000016                          16

                                ;;*****************************************************************
                                ;*TEST 25        SETTLE TEST ON 8 EDGES
                                ;;*****************************************************************
                        TST25:  SCOPE
004564  000004
004566  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004574  004537  006122                  JSR     R5,SET8                 ;SETTLE-POSITIVE DIRECTION
004600  000015                          15
004602  000016                          16
004604  012737  000116  001346          MOV     #116,TEMP
004612  004537  006122                  JSR     R5,SET8                 ;SETTLE-NEGATIVE DIRECTION
004616  000016                          16
004620  000015                          15

                                ;;*****************************************************************
                                ;*TEST 26        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
                                ;;*****************************************************************
                        TST26:  SCOPE
004622  000004
004624  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004632  005737  001202                  TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
004636  001402                          BEQ     LEND
004640  004737  006750                  JSR     PC,DIFLIN
004644  000207                  LEND:   RTS     PC                      ;RETURN TO TEST SECTION

004646  005000                  DAWAIT: CLR     R0
004650  105300                  1$:     DECB    R0
004652  001376                          BNE     1$
004654  000207                          RTS     PC
```

```
1011                                            .SBTTL  CALIBRATION TEST
1012  004656  012737  004656  001364  BEGINC: MOV     #BEGINC,TADDR          ;TEST ADDRESS IN TADDR
1013  004664  005037  001426                  CLR     MYTEMP
1014
1015                                    ;*      MOV     MYTEMP,@STREG  ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1016  004700  104401  013613                  TYPE    ,HEADS                 ;TYPE OUT HEADING
1017  004704  005037  177776                  CLR     PSW
1018  004710  017700  174224          1$:     MOV     @SWR,R0                ;READ CHANNEL FROM SWITCH REG.
1019  004714  042700  177700                  BIC     #177700,R0             ;ISOLATE MUX BITS
1020  004720  032777  020000  174212          BIT     #BIT13,@SWR            ;IS BIT 13 SET?
1021  004726  001005                          BNE     2$             ;;YES,SKIP TYPEOUT
1022  004730  104401  012317                  TYPE    ,CH
1023  004734  010046                          MOV     R0,-(SP)       ;;SAVE R0 FOR TYPEOUT
1024                                                                 ;;TYPE CHANNEL
1025  004736  104403                          TYPOS                  ;;GO TYPE--OCTAL ASCII
1026  004740     002                          .BYTE   2              ;;TYPE 2 DIGIT(S)
1027  004741     000                          .BYTE   0              ;;SUPPRESS LEADING ZEROS
1028  004742                          2$:
1029  004742  000300                          SWAB    R0                     ;SWITCH BYTES
1030  004744  010037  001426                  MOV     R0,MYTEMP
1031
1032                                    ;*      MOV     MYTEMP,@STREG  ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1033  004760  012702  000010                  MOV     #10,R2                 ;TYPEOUT COUNTER
1034  004764                          3$:
1035
1036
1037                                    ;*      MOV     @STREG,MYTEMP  ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
1038  004774  005237  001426                  INC     MYTEMP
1039
1040                                    ;*      MOV     MYTEMP,@STREG  ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1041  005010                          30$:
1042
1043                                    ;*      MOV     @STREG,MYTEMP  ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
1044  005020  105737  001426                  TSTB    MYTEMP
1045  005024  100371                          BPL     30$
1046
1047                                    ;*      MOV     @ADBUFF,MYTEMP ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
1048  005036  013700  001426                  MOV     MYTEMP,R0              ;/PUT CONVERTED VALUE IN R0.
1049  005042  032777  020000  174070          BIT     #BIT13,@SWR                    ;IS BIT 13 SET?
1050  005050  001403                          BEQ     4$                     ;NOT SET, TYPE OUT LIST
1051  005052  010077  174064                  MOV     R0,@DISPLAY                    ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
1052  005056  000714                          BR      1$                     ;REPEAT CONVERSION
1053  005060  104401  012322          4$:     TYPE    ,SPACE
1054  005064  010046                          MOV     R0,-(SP)       ;;SAVE R0 FOR TYPEOUT
1055                                                                 ;;PRINT OCTAL CONVERTED VALUE
1056  005066  104403                          TYPOS                  ;;GO TYPE--OCTAL ASCII
1057  005070     004                          .BYTE   4              ;;TYPE 4 DIGIT(S)
1058  005071     001                          .BYTE   1              ;;TYPE LEADING ZEROS
1059  005072  012701  010000                  MOV     #10000,R1
1060  005076  005301                  5$:     DEC     R1
1061  005100  001376                          BNE     5$
1062  005102  005302                          DEC     R2                     ;DECREMENT THE COUNTER
1063  005104  001327                          BNE     3$                     ;NO CARRIAGE RETURN
1064  005106  104401  001171                  TYPE    ,$CRLF                 ;CARRIAGE RETURN
```

```
  1065  005112  000676                        BR      1$                      ;REPEAT CONVERSION
```

```
1066                                            .SBTTL      LOGIC TEST SECTION
1067  005114  012737  005114  001364  BEGL:     MOV       #BEGL,TADDR       ;TEST ADDRESS
1068  005122  005037  001430                     CLR       EDINT
1069  005126  004737  002552                     JSR       PC,TESTAD         ;NO OF ADDITIONAL AD'S
1070  005132  004737  002710            1$:      JSR       PC,BEGINL         ;LOGIC TESTS
1071  005136  004737  005322                     JSR       PC,BUMPAD         ;MORE TO TEST?
1072  005142  000773                           BR        1$                ;TEST NEXT A/D
1073  005144  012737  005132  012016           MOV       #1$,AGTST         ;ADDRESS FOR EOP
1074  005152  000137  012020                     JMP       $EOP              ;TYPE END OF PASS
1075
1076                                            .SBTTL      AUTO TEST
1077  005156  012737  005156  001364  BEGINA:  MOV       #BEGINA,TADDR     ;TEST ADDRESS
1078  005164  005037  001430                     CLR       EDINT
1079  005170  005037  001202                     CLR       $PASS             ;CLEAR PASS COUNTER
1080  005174  004737  002552                     JSR       PC,TESTAD         ;NO. OF AD'S TO BE TESTED
1081  005200  004737  002710            1$:      JSR       PC,BEGINL         ;LOGIC TESTS
1082  005204  104401  012775                     TYPE      ,MEND             ;TYPE END OF LOGIC TEST
1083  005210  013746  001316                     MOV       $TREG,-(SP)       ;SAVE $TREG FOR TYPEOUT
1084  005214  104403                           TYPOS                       ;TYPE OCTAL NUMBER
1085  005216     006                           .BYTE     6                 ;TYPE 6 DIGITS
1086  005217     001                           .BYTE     1                 ;TYPE LEADING ZEROS
1087  005220  104401  001171                     TYPE      ,$CRLF            ;TYPE A CR,LF
1088  005224  004737  003436                     JSR       PC,WRAP
1089  005230  004737  005322                     JSR       PC,BUMPAD         ;TEST NEXT A/D
1090  005234  000761                           BR        1$                ;TEST NEXT AD
1091  005236  012737  005200  012016           MOV       #1$,AGTST         ;ADDRESS FOR EOP
1092  005244  000137  012020                     JMP       $EOP              ;TYPE END OF PASS
1093
1094                                            .SBTTL      WRAPAROUND TEST
1095  005250  012737  005250  001364  BEGINW:  MOV       #BEGINW,TADDR     ;TEST ADDRESS
1096  005256  005037  001430                     CLR       EDINT
1097  005262  005037  001202                     CLR       $PASS             ;CLEAR PASS COUNT
1098  005266  004737  002552                     JSR       PC,TESTAD         ;NO. OF AD'S TO BE TESTED
1099  005272  004737  003436            1$:      JSR       PC,WRAP           ;WRAPAROUND TESTS
1100  005276  005037  001430                     CLR       EDINT
1101  005302  004737  005322                     JSR       PC,BUMPAD         ;MORE A/D'S TO BE TESTED?
1102  005306  000771                           BR        1$                ;YES-GO TEST NEXT AD11K
1103  005310  012737  005272  012016           MOV       #1$,AGTST
1104  005316  000137  012020                     JMP       $EOP              ;INCREMENTS $PASS
```

```
1105                                        .SBTTL      DETERMINE IF MORE AD11K'S TO BE TESTED
1106  005322  005737  001354          BUMPAD: TST     NBEXT                   ;ADDITIONAL AD'S?
1107  005326  001421                          BEQ     FIXADR                  ;NO-INITIALIZE ADDRESSES
1108  005330  063737  001326  001316          ADD     VADR,STREG              ;SET UP NEW ST. REG.
1109  005336  063737  001326  001320          ADD     VADR,ADBUFF             ;SET UP NEW BUFFER ADDRESS
1110  005344  063737  001330  001456          ADD     VVCT,VECTOR             ;SET UP NEW VECTOR
1111  005352  063737  001330  001324          ADD     VVCT,VECTR1
1112  005360  005077  173740                  CLR     @VECTR1
1113  005364  005337  001354                  DEC     NBEXT                   ;ONE LESS AD11K
1114  005370  000441                          BR      BYPASS
1115  005372  062716  000002          FIXADR: ADD     #2,(SP)
1116  005376  013737  001250  001316  FIXONE: MOV     $BASE,STREG             ;RELOAD INITIAL ADDRESSES
1117  005404  013737  001250  001320          MOV     $BASE,ADBUFF
1118  005412  062737  000002  001320          ADD     #2,ADBUFF
1119  005420  013737  001244  001456          MOV     $VECT1,VECTOR
1120  005426  042737  170000  001456          BIC     #170000,VECTOR
1121  005434  113737  001245  001322          MOVB    $VECT1+1,BASEBR
1122  005442  105037  001323                  CLRB    BASEBR+1                ;CLEAR HIGH BYTE
1123  005446  013737  001456  001324          MOV     VECTOR,VECTR1
1124  005454  062737  000002  001324          ADD     #2,VECTR1
1125  005462  005077  173636                  CLR     @VECTR1
1126  005466  013737  001356  001354          MOV     NMBEXT,NBEXT            ;RESET COUNTER
1127                                        ;;LOAD .+2 AND HALT TRAP CATCH;;
1128  005474  012700  000216          BYPASS: MOV     #216,R0                 ;FILL .+2
1129  005500  012701  000214                  MOV     #214,R1                 ;LOAD HALT
1130  005504  020137  001334          1$:     CMP     R1,KBVECT
1131  005510  001410                          BEQ     2$
1132  005512  010021                          MOV     R0,(R1)+
1133  005514  005021                          CLR     (R1)+
1134  005516  010100                          MOV     R1,R0
1135  005520  005720                          TST     (R0)+
1136  005522  020027  001002                  CMP     R0,#1002
1137  005526  001366                          BNE     1$
1138  005530  000207                          RTS     PC                      ;TEST NEXT A/D
1139  005532  022021                  2$:     CMP     (R0)+,(R1)+
1140  005534  022021                          CMP     (R0)+,(R1)+
1141  005536  000762                          BR      1$
1142
1143
1144                                        .SBTTL      NOISE TEST, 1 EDGE
1145  005540  012737  005540  001364  BEGINN: MOV     #BEGINN,TADDR           ;TEST ADDRESS IN TADDR
1146  005546  104401  012126                  TYPE    ,NOIMSG                 ;ASK FOR CHANNEL
1147  005552  104401  013632                  TYPE    ,ASKCH
1148  005556  017737  173356  001350  1$:     MOV     @SWR,CH1                ;LOAD CHANNEL
1149  005564  042737  177700  001350          BIC     #177700,CH1
1150  005572  012737  000200  001346          MOV     #200,TEMP               ;LOAD DAC VALUE
1151  005600  004537  010400                  JSR     R5,NOITST               ;GO TO NOISE SUBROUTINE
1152  005604  001350                          CH1
1153  005606  000763                          BR      1$
```

```
                                        .SBTTL      INTERCHANNEL SETTLING TEST, 1 EDGE
005610  012737  005610  001364  BEGINS: MOV     #BEGINS,TADDR           ;TEST ADDRESS IN TADDR
005616  104401  012146                  TYPE    ,SETMSG                 ;ASK FOR CHANNELS
005622  104407                          RDOCT
005624  012637  001350                  MOV     (SP)+,CH1
005630  104401  012433                  TYPE    ,TOMSG
005634  104407                          RDOCT
005636  012637  001352                  MOV     (SP)+,CH2
005642  012737  000200  001346  BK3:    MOV     #200,TEMP               ;LOAD DAC
005650  013737  001352  001362          MOV     CH2,CHANL
005656  004737  006226                  JSR     PC,GETEDG               ;GET EDGE VALUES
005662  005002                          CLR     R2
005664  004737  006060                  JSR     PC,SET1A                ;SCALING = .02 LSB
005670  004737  006060                  JSR     PC,SET1A                ;MAKE IT .01 LSB
005674  100001                          BPL     POSR2
005676  005402                          NEG     R2
005700  010204                  POSR2:  MOV     R2,R4
005702  012737  000001  006450          MOV     #1,EDGFLG
005710  004737  005716                  JSR     PC,TYPSET
005714  000752                          BR      BK3
005716  004737  011504          TYPSET: JSR     PC,DECTYP
005722  104401  012327                  TYPE    ,LSB
005726  013746  001352                  MOV     CH2,-(SP)               ;;SAVE CH2 FOR TYPEOUT
                                                                        ;;TYPE CH
005732  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
005734     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
005735     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
005736  104401  013724                  TYPE    ,MAT                    ;TYPE ASCIZ STRING
005742  004737  006406                  JSR     PC,TYPEDG
005746  104401  012342                  TYPE    ,SETCH
005752  013746  001350                  MOV     CH1,-(SP)               ;;SAVE CH1 FOR TYPEOUT
                                                                        ;;TYPE CH
005756  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
005760     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
005761     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
005762  104401  012364                  TYPE    ,ATMSG
005766  013737  001350  006024          MOV     CH1,1$
005774  163737  001332  006024          SUB     BASECH,1$
006002  012737  000200  001426          MOV     #200,MYTEMP

006020  004537  011072          ;*      JSR     R5,CONVRT               ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
006024  000000                  1$:     0
006026  013746  001346                  MOV     TEMP,-(SP)              ;;SAVE TEMP FOR TYPEOUT
                                                                        ;;TYPE VALUE
006032  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
006034     004                          .BYTE   4                       ;;TYPE 4 DIGIT(S)
006035     001                          .BYTE   1                       ;;TYPE LEADING ZEROS
006036  020437  011746                  CMP     R4,VSET
006042  003003                          BGT     ERR
006044  104401  012474                  TYPE    ,OKMSG
006050  000207                          RTS     PC
```

```
1206  006052  104401  012505          ERR:    TYPE    ,ERMSG
1207  006056  000207                  RTS     PC
1208
1209
1210
1211                                  ;;SUBROUTINE FOR SETTLING TESTS;;
1212  006060  013737  001352  001360  SET1A:  MOV     CH2,DUMmY                ;LOAD DUMMY
1213  006066  004537  006452          JSR     R5,$ARSUB                ;DO SAR ROUTINE AT 50%
1214  006072  000062                  50.
1215  006074  063702  001404          ADD     DAC,R2          ;ADD RESULT TO R2
1216  006100  013737  001350  001360  MOV     CH1,DUMMY                ;CHANGE DUMMY VALUE
1217  006106  004537  006452          JSR     R5,$ARSUB                ;DO SAR ROUTINE AT 50%
1218  006112  000062                  50.
1219  006114  163702  001404          SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
1220  006120  000207                  RTS     PC              ;RETURN
1221
1222  006122  012537  001350  SET8:   MOV     (R5)+,CH1                ;GET FIRST CHANNEL
1223  006126  012537  001352          MOV     (R5)+,CH2                ;GET SECOND CHANNEL
1224  006132  063737  001332  001350  ADD     BASECH,CH1
1225  006140  063737  001332  001352  ADD     BASECH,CH2
1226  006146  004737  006226          JSR     PC,GETEDG                ;GET EDGE VALUES
1227  006152  005002                  CLR     R2
1228  006154  012703  000010          MOV     #10,R3          ;SET UP COUNTER
1229  006160  004737  006060  SETAA:  JSR     PC,SET1A                 ;GET SETTLE VALUES
1230  006164  005237  001410          INC     EDGE
1231  006170  005303                  DEC     R3
1232  006172  001372                  BNE     SETAA           ;REPEAT 8 TIMES
1233  006174  162737  000010  001410  SUB     #10,EDGE
1234  006202  005702                  TST     R2
1235  006204  100001                  BPL     R2POS
1236  006206  005402                  NEG     R2
1237  006210  010204          R2POS:  MOV     R2,R4
1238  006212  012737  000010  006450  MOV     #8.,EDGFLG
1239  006220  004737  005716          JSR     PC,TYPSET                ;TYPE OUT RESULTS
1240  006224  000205                  RTS     R5                       ;RETURN
```

```
1241                                    ;SUBROUTINE TO GET EDGE VALUE
1242                                    ;CALL=JSR   PC,GETEDG
1243                                    ;CONVERSIONS ON A/D CHANNEL 'CHANL'
1244                                    ;RESULT IN EDGE, USES R0
1245  006226                            GETEDG:
1246
1247                                    ;*      MOV     TEMP,@ADBUFF    ;/ PUT DATA FROM TEMP TO DEVICE REG ADBUFF
1248  006236  113700  001362                    MOVB    CHANL,R0                ;GET CHANNEL
1249  006242  000300                            SWAB    R0              ;SET UP A.D STATUS REG.
1250  006244  010037  001426                    MOV     R0,MYTEMP
1251
1252                                    ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1253  006260  012700  000100                    MOV     #100,R0         ;DAC SETTLING DELAY
1254  006264  005300                    1$:     DEC     R0
1255  006266  001376                            BNE     1$
1256  006270  005037  001410                    CLR     EDGE
1257  006274  012700  000010                    MOV     #10,R0
1258  006300                            CONV:
1259
1260
1261                                    ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
1262  006310  005237  001426                    INC     MYTEMP
1263
1264                                    ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1265  006324                            30$:
1266
1267                                    ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
1268  006334  105737  001426                    TSTB    MYTEMP
1269  006340  100371                            BPL     30$
1270
1271
1272                                    ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
1273  006352  063737  001426  001410            ADD     MYTEMP,EDGE
1274  006360  005300                            DEC     R0
1275  006362  001346                            BNE     CONV
1276  006364  006237  001410                    ASR     EDGE
1277  006370  006237  001410                    ASR     EDGE
1278  006374  006237  001410                    ASR     EDGE
1279  006400  005537  001410                    ADC     EDGE
1280  006404  000207                            RTS     PC
1281
1282                                    ;;SUBROUTINE TO TYPE EDGE VALUES;;
1283  006406  013703  001410            TYPEDG: MOV     EDGE,R3
1284  006412  010346                            MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
1285                                                                    ;;TYPE OCTAL VALUE OF EDGE
1286  006414  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
1287  006416     004                            .BYTE   4               ;;TYPE 4 DIGIT(S)
1288  006417     001                            .BYTE   1               ;;TYPE LEADING ZEROS
1289  006420  023727  006450  000001            CMP     EDGFLG,#1
1290  006426  001407                            BEQ     RET
1291  006430  062703  000007                    ADD     #7,R3
1292  006434  104401  013674                    TYPE    ,C1             ;TYPE ASCIZ STRING
1293  006440  010346                            MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
1294                                                                    ;;TYPE EDGE VALUE
```

```
1295  006442  104403                    TYPOS                   ;;GO TYPE--OCTAL ASCII
1296  006444  004                       .BYTE   4               ;;TYPE 4 DIGIT(S)
1297  006445  001                       .BYTE   1               ;;TYPE LEADING ZEROS
1298  006446  000207              RET:   RTS     PC
1299  006450  000000              EDGFLG: 0
```

```
                                        ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
                                        ;CALL=JSR   R5,SARSUB
                                        ;   XXX;XXX=PERCENT
                                        ;RESULT RETURNED IN 'DAC', USES R0,R1,R4
006452  012537  001422          SARSUB: MOV     (R5)+,PERCNT            ;GET PERCENT
006456  006337  001422                  ASL     PERCNT
006462  006337  001422                  ASL     PERCNT
006466  012737  000620  006746          MOV     #400.,CNNO              ;NO OF SAMPLES FOR SHORT PASS.
006474  032777  004000  172436          BIT     #BIT11,@SWR             ;USER WANT SHORT PASS?
006502  001010                          BNE     SAR1
006504  000407                          BR      SAR1                    ;ALWAYS USE SHORT SAMPLE COUNT.
006506  012737  003100  006746          MOV     #1600.,CNNO
006514  006337  001422                  ASL     PERCNT                  ;RESCALE PERCENT FOR 1600.
006520  006337  001422                  ASL     PERCNT                  ;POINTS PER BURST
006524  012737  000200  001412  SAR1:   MOV     #200,BITPNT             ;INITIALIZE BIT POINTER AT MSB
006532  005037  001404                  CLR     DAC                     ;INITIALIZE DAC VALUE
006536  004537  020740                  JSR     R5,$PUTS
006542  001316                          .WORD   STREG
006544  005000                  TRY:    CLR     R0
006546  063737  001412  001404          ADD     BITPNT,DAC              ;TRY BIT
                                        ;*      MOV     DAC,@ADBUFF     ;/ PUT DATA FROM DAC TO DEVICE REG ADBUFF
006564  012737  000100  001406          MOV     #100,DELAY
006572  005337  001406          1$:     DEC     DELAY                   ;STALL TIME
006576  001375                          BNE     1$
006600  013701  006746                  MOV     CNNO,R1                 ;SET UP FOR 1600. OR 400. CONVERSIONS
006604  113737  001362  001435          MOVB    CHANL,$TEMP2+1
006612  052737  000001  001434          BIS     #1,$TEMP2
006620  113737  001360  001433          MOVB    DUMMY,$TEMP1+1
006626  052737  000001  001432          BIS     #1,$TEMP1
006634                          NXTCVT:
006634  013777  001432  172604  $T6Mp:  MOV     $TEMP1,@KMAD4
006642  112777  000006  172572          MOVB    #6,@KMAD2
006650  122777  000377  172564  10$:    CMPB    #377,@KMAD2
006656  001374                          BNE     10$
006660  013777  001434  172560          MOV     $TEMP2,@KMAD4
006666  112777  000006  172546          MOVB    #6,@KMAD2
006674  122777  000377  172540  20$:    CMPB    #377,@KMAD2
006702  001374                          BNE     20$
006704  027737  172536  001410          CMP     @KMAD4,EDGE
006712  002001                          BGE     2$
006714  005200                          INC     R0                      ;COUNT RESULTS .LT. EDGE
006716  005301                  2$:     DEC     R1
006720  001345                          BNE     NXTCVT
006722  020037  001422                  CMP     R0,PERCNT
006726  003003                          BGT     SHIFT
006730  163737  001412  001404          SUB     BITPNT,DAC              ;TAKE THE BIT OUT
006736  006237  001412          SHIFT:  ASR     BITPNT
006742  001300                          BNE     TRY
006744  000205                          RTS     R5

006746  000000                  CNNO:   .WORD   0
```

```
1352                                        ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
1353  006750  104401  013120        DIFLIN: TYPE    ,MSG20
1354  006754  005037  001424                CLR     OUT
1355  006760  012700  022354                MOV     #BUFFER,R0
1356  006764  012701  010000                MOV     #4096.,R1       ;4096 WORDS FOR HISTOGRAM
1357  006770  005020                CLEAR1: CLR     (R0)+           ;CLEAR BUFFER AREA
1358  006772  005301                        DEC     R1
1359  006774  001375                        BNE     CLEAR1
1360  006776  012700  021534                MOV     #DIST,R0        ;DISTRIBUTION BUFFER POINTER
1361  007002  012701  000310                MOV     #200.,R1        ;200. WORDS FOR DISTRIBUTION
1362  007006  005003                        CLR     R3
1363  007010  005037  001424                CLR     OUT
1364  007014  005037  001336                CLR     WIDE
1365  007020  005037  001340                CLR     NARROW
1366  007024  005037  001342                CLR     FIRST
1367  007030  005037  001344                CLR     SKIPST
1368  007034  005020                CLEAR2: CLR     (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
1369  007036  005301                        DEC     R1
1370  007040  001375                        BNE     CLEAR2
1371  007042  012700  000011        CHANNL: MOV     #11,R0          ;CHANNEL 11
1372  007046  063700  001332                ADD     BASECH,R0
1373  007052  000300                        SWAB    R0                              ;LOAD MUX BITS
1374  007054  004537  020740                JSR     R5,$PUTS
1375  007060  001316                        .WORD   STREG
1376  007062  010037  001426                MOV     R0,MYTEMP
1377
1378                                  ;*    MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1379  007076  010037  001432                MOV     R0,$TEMP1
1380  007102  052737  000001  001432        BIS     #1,$TEMP1
1381  007110  012700  001440                MOV     #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
1382  007114  012777  001704  172334        MOV     #RETURN,@VECTOR
1383  007122  012701  007776        AGAIN:  MOV     #4094.,R1
1384  007126  004737  011010        NEXT:   JSR     PC,RANDY        ;GET RANDOM NUMBER
1385  007132  013702  001366                MOV     RNA,R2
1386  007136  042702  177760                BIC     #177760,R2      ;MASK IT TO 4 BITS ONLY
1387  007142  001402                        BEQ     CONVR
1388  007144  005302                DELAY3: DEC     R2              ;STALL
1389  007146  001376                        BNE     DELAY3          ;TIME
1390  007150                        CONVR:
1391  007150  013777  001432  172270 $TBF4:  MOV     $TEMP1,@KMAD4
1392  007156  112777  000006  172256        MOVB    #6,@KMAD2
1393  007164  122777  000377  172250 31$:    CMPB    #377,@KMAD2
1394  007172  001374                        BNE     31$
1395  007174  017702  172246                MOV     @KMAD4,R2
1396  007200  001413                        BEQ     DELAY1          ;IGNORE IF =0
1397  007202  020227  007777                CMP     R2,#7777        ;IGNORE IF =7777
1398  007206  001413                        BEQ     DELAY2
1399  007210  006302                        ASL     R2
1400  007212  005262  022354                INC     BUFFER(R2)      ;MAKE HISTOGRAM
1401  007216  100013                        BPL     OKAY
1402  007220  012762  077777  022354        MOV     #077777,BUFFER(R2)      ;PREVENT OVERFLOW
1403  007226  000407                        BR      OKAY
1404  007230  020227  007777        DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
1405  007234  001400                        BEQ     DELAY2          ;WITH DUMMY INSTR.
```

```
1406  007236  005201                DELAY2: INC     R1
1407  007240  005263  001346                INC     TEMP(R3)
1408  007244  100403                        BMI     NOTOK
1409  007246  005301                OKAY:   DEC     R1
1410  007250  001326                        BNE     NEXT
1411  007252  000403                        BR      AROUND
1412  007254  005037  001346        NOTOK:  CLR     TEMP
1413  007260  000772                        BR      OKAY
1414  007262  005300                AROUND: DEC     R0
1415  007264  001316                        BNE     AGAIN
1416  007266  012700  007776                MOV     #4094.,R0
1417  007272  012701  022356                MOV     #BUFFER+2,R1
1418  007276  012102                READ:   MOV     (R1)+,R2        ;GET STATE WIDTH
1419  007300  006202                        ASR     R2              ;1 LSB = 800.
1420  007302  006202                        ASR     R2
1421  007304  006202                        ASR     R2
1422  007306  005502                        ADC     R2              ;1 LSB = 100.
1423  007310  020227  000310                CMP     R2,#200.        ;OUT OF RANGE?
1424  007314  002403                        BLT     INRNGE
1425  007316  005237  001424                INC     OUT             ;YES - INCREMENT COUNTER
1426  007322  000423                        BR      TYPBAD
1427  007324  006302                INRNGE: ASL     R2
1428  007326  005262  021534                INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
1429  007332  006202                        ASR     R2
1430  007334  020227  000062                CMP     R2,#50.         ;IS IT 1/2 LSB?
1431  007340  002007                        BGE     NOTNAR
1432  007342  005237  001340                INC     NARROW
1433  007346  005702                        TST     R2              ;IS IT A SKIPPED STATE?
1434  007350  001002                        BNE     31$
1435  007352  005237  001344                INC     SKIPST
1436  007356  000405                31$:    BR      TYPBAD
1437  007360  020227  000226        NOTNAR: CMP     R2,#150.        ;IS IT 1.5 LSB?
1438  007364  003426                        BLE     LAST
1439  007366  005237  001336                INC     WIDE
1440  007372  005737  001342        TYPBAD: TST     FIRST
1441  007376  001004                        BNE     60$
1442  007400  005237  001342                INC     FIRST
1443  007404  104401  012277                TYPE    ,STATE
1444  007410  010103                60$:    MOV     R1,R3
1445  007412  162703  022356                SUB     #BUFFER+2,R3
1446  007416  006203                        ASR     R3
1447  007420  010346                        MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
1448                                                                ;;TYPE STATE
1449  007422  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
1450  007424      004                       .BYTE   4               ;;TYPE 4 DIGIT(S)
1451  007425      001                       .BYTE   1               ;;TYPE LEADING ZEROS
1452  007426  104401  012273                TYPE    ,DASH
1453  007432  004737  011504                JSR     PC,DECTYP
1454  007436  104401  012264                TYPE    ,LSBMSG
1455  007442  005300                LAST:   DEC     R0
1456  007444  001314                        BNE     READ
1457  007446  112737  000177  014572        MOVB    #177,DECPNT
1458  007454  013702  001344                MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
1459  007460  004737  011504                JSR     PC,DECTYP               ;TYPE IT
```

```
1460   007464   104401   012522                    TYPE    ,SKPMSG         ;TYPE MESSAGE
1461   007470   005737   001344                    TST     $KIPST
1462   007474   001403                             BEQ     1$
1463   007476   104401   012505                    TYPE    ,ERMSG          ;TYPE "ERROR"
1464   007502   000402                             BR      NAR
1465   007504   104401   012474          1$:       TYPE    ,OKMSG          ;TYPE #OK#
```

```
1466  007510  013702  001340          NAR:    MOV     NARROW,R2               ;GET NO. OF NARROW STATES
1467  007514  004737  011504                  JSR     PC,DECTYP               ;TYPE IT
1468  007520  104401  012544                  TYPE    ,NARMSG         ;TYPE MESSAGE
1469  007524  013702  001336                  MOV     WIDE,R2
1470  007530  063702  001424                  ADD     OUT,R2
1471  007534  004737  011504                  JSR     PC,DECTYP               ;TYPE NO. OF WIDE STATES
1472  007540  104401  012603                  TYPE    ,WIDMSG         ;TYPE MESSAGE
1473  007544  013702  001424                  MOV     OUT,R2
1474  007550  004737  011504                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE 2 LSB
1475  007554  104401  012642                  TYPE    ,OUTMSG         ;TYPE MESSAGE
1476  007560  005737  001424                  TST     OUT
1477  007564  001403                          BEQ     11$
1478  007566  104401  012505                  TYPE    ,ERMSG          ;TYPE "ERROR"
1479  007572  000402                          BR      HALF
1480  007574  104401  012474          11$:    TYPE    ,OKMSG          ;TYPE "OK"
1481  007600  013702  001340          HALF:   MOV     NARROW,R2
1482  007604  063702  001336                  ADD     WIDE,R2
1483  007610  063702  001424                  ADD     OUT,R2
1484  007614  010200                          MOV     R2,R0
1485  007616  004737  011504                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE LIMITS
1486  007622  112737  000056  014572          MOVB    #56,DECPNT
1487  007630  104401  012675                  TYPE    ,HAFMSG
1488  007634  020027  000051                  CMP     R0,#41.         ;COMPARE IT TO NOMINAL
1489  007640  003403                          BLE     21$
1490  007642  104401  012505                  TYPE    ,ERMSG          ;TYPE "ERROR"
1491  007646  000402                          BR      SWDIST
1492  007650  104401  012474          21$:    TYPE    ,OKMSG          ;TYPE "OK"
1493  007654  005737  001400          SWDIST: TST     FLAG            ;VT55?
1494  007660  001426                          BEQ     RELACC
1495  007662  004737  010342                  JSR     PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
1496  007666  104401  013152                  TYPE    ,MSG16
1497  007672  104401  013753                  TYPE    ,BUFF1          ;TYPE BUFF1-PRINT GRID
1498  007676  012700  021534                  MOV     #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
1499  007702  012701  000310                  MOV     #200.,R1        ;GO 200. TIMES UP TO 2 LSB
1500  007706  012002          NXTY1:  MOV     (R0)+,R2
1501  007710  004737  011402                  JSR     PC,LOADY
1502  007714  005002                          CLR     R2
1503  007716  004737  011402                  JSR     PC,LOADY
1504  007722  005301                          DEC     R1
1505  007724  001370                          BNE     NXTY1
1506  007726  104401  013676                  TYPE    ,C2             ;TYPE ASCIZ STRING
1507  007732  004737  010342                  JSR     PC,DELCLR
1508
```

```
                                        ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

                                RELACC: CLR     R1              ;RUNNING ERROR = 0
                                        CLR     R3              ;MAXIMUM ERROR = 0
                                        TYPE    ,MSG21
                                        MOV     #BUFFER+2,R0
                                NXTSTA: MOV     (R0),R2         ;STATE WIDTH = R2
                                        SUB     #800.,R2        ;STATE WIDTH ERROR IN R2
                                        ADD     R2,R1           ;UPDATE RUNNING ERROR
                                        MOV     R1,(R0)+        ;SAVE IN BUFFER
                                        MOV     R1,R4           ;SAVE IN R4 ALSO
                                        BPL     PLUS            ;IS IT POSITIVE?
                                        NEG     R4              ;NO - MAKE IT POSITIVE
                                PLUS:   CMP     R4,R3           ;CHECK AGAINST PREVIOUS MAX. ERROR
                                        BLE     NOTNEW          ;NOT A NEW MAXIMUM
                                        MOV     R4,R3           ;UPDATE MAXIMUM IN R3
                                        MOV     R0,R5
                                        SUB     #BUFFER+2,R5
                                        ASR     R5              ;R5=EDGE VALUE AT MAX. RELACC
                                NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
                                        BNE     NXTSTA          ;NO - REPEAT
                                        ASR     R3              ;RESCALE FROM 1 LSB = 800. SCALING
                                        ASR     R3              ;TO 1 LSB = 100. SCALING
                                        ASR     R3
                                        ADC     R3
                                        MOV     R3,R2
                                        JSR     PC,DECTYP
                                        TYPE    ,LINEA
                                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                                ;;TYPE VALUE
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   4               ;;TYPE 4 DIGIT(S)
                                        .BYTE   1               ;;TYPE LEADING ZEROS
                                        TYPE    ,SLASH                  ;PRINT '/'
                                        INC     R5
                                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                                ;;TYPE VALUE
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   4               ;;TYPE 4 DIGIT(S)
                                        .BYTE   1               ;;TYPE LEADING ZEROS
                                        CMP     R3,VLIN
                                        BLE     41$
                                        TYPE    ,ERMSG
                                        BR      42$
                                41$:    TYPE    ,OKMSG
                                42$:    TST     FLAG            ;VT55?
                                        BEQ     L02
                                        MOV     #BUFFER,R0
                                        MOV     #4096.,R1
```

```
1558  010120  011002                GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
1559  010122  006202                        ASR     R2              ;RESCALE IT TO 1 LSB = 100.
1560  010124  006202                        ASR     R2
1561  010126  006202                        ASR     R2
1562  010130  005502                        ADC     R2
1563  010132  062702  000166              ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
1564  010136  010220                        MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
1565  010140  005301                        DEC     R1
1566  010142  001366                        BNE     GETDAT
1567  010144  012700  022354              MOV     #BUFFER,R0
1568  010150  012704  022354              MOV     #BUFFER,R4
1569  010154  012705  022356              MOV     #BUFFER+2,R5
1570  010160  012701  001000              MOV     #512.,R1
1571  010164  012702  000007        NXT8: MOV     #7.,R2
1572  010170  012003                        MOV     (R0)+,R3
1573  010172  010337  001414              MOV     R3,MIN          ;MINIMUM
1574  010176  010337  001420              MOV     R3,MAX          ;MAXIMUM
1575  010202  012003                  NXTCMP: MOV     (R0)+,R3
1576  010204  020337  001414              CMP     R3,MIN
1577  010210  002002                        BGE     MAXTST
1578  010212  010337  001414              MOV     R3,MIN          ;NEW MINIMUM
1579  010216  020337  001420        MAXTST: CMP     R3,MAX
1580  010222  003402                        BLE     TST8
1581  010224  010337  001420              MOV     R3,MAX          ;NEW MAXIMUM
1582  010230  005302                  TST8: DEC     R2
1583  010232  001363                        BNE     NXTCMP
1584  010234  013724  001414              MOV     MIN,(R4)+
1585  010240  013725  001420              MOV     MAX,(R5)+
1586  010244  022425                        CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
1587  010246  005301                        DEC     R1
1588  010250  001345                        BNE     NXT8
1589  010252  104401  013060              TYPE    ,MSG18
1590  010256  104401  014001              TYPE    ,BUFF2          ;TYPE BUFF2
1591  010262  012700  022354              MOV     #BUFFER,R0
1592  010266  004737  010320              JSR     PC,LOAD
1593  010272  104401  013701              TYPE    ,C3             ;TYPE ASCIZ STRING
1594  010276  012700  022356              MOV     #BUFFER+2,R0
1595  010302  004737  010320              JSR     PC,LOAD
1596  010306  104401  013676              TYPE    ,C2             ;TYPE ASCIZ STRING
1597  010312  004737  010342              JSR     PC,DELCLR
1598  010316  000207                  L02: RTS     PC
1599  010320  012701  001000        LOAD: MOV     #512.,R1
1600  010324  012002                LOADO: MOV     (R0)+,R2
1601  010326  005720                        TST     (R0)+
1602  010330  004737  011402              JSR     PC,LOADY
1603  010334  005301                        DEC     R1
1604  010336  001372                        BNE     LOADO
1605  010340  000207                        RTS     PC
```

```
 1606  010342  005000                    DELCLR: CLR     R0
 1607  010344  012701  000020                    MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
 1608  010350  005300                    1$:     DEC     R0
 1609  010352  001376                            BNE     1$
 1610  010354  005301                            DEC     R1
 1611  010356  001374                            BNE     1$
 1612  010360  032777  010000  170552            BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
 1613  010366  001401                            BEQ     2$              ;;DON'T HALT FOR DISPLAY
 1614  010370  000000                            HALT
 1615  010372  104401  014021            2$:     TYPE    ,VTINIT
 1616  010376  000207                            RTS     PC
 1617                                    ;;NOISE SUBROUTINE;;
 1618  010400  013537  001362            NOITST: MOV     @(R5)+,CHANL            ;LOAD CHANNEL
 1619  010404  013737  001362  001360            MOV     CHANL,DUMMY             ;LOAD DUMmY CHANNEL
 1620  010412  004737  006226                    JSR     PC,GETEDG               ;GET EDGE VALUE
 1621  010416  004737  010572                    JSR     PC,NOIA                 ;GET RMS AND PEAK VALUES
 1622  010422  012737  000001  006450            MOV     #1,EDGFLG
 1623  010430  004737  010436                    JSR     PC,TYPRP                ;TYPE RMS AND PEAK VALUES
 1624  010434  000205                            RTS     R5
 1625
 1626
 1627
 1628
 1629
 1630                                    ;;TYPE RMS AND PEAK VALUES;;
 1631  010436  104401  012371            TYPRP:  TYPE    ,NOI
 1632  010442  005737  001374                    TST     RMS
 1633  010446  100002                            BPL     POSRMS
 1634  010450  005037  001374                    CLR     RMS                     ;RMS<0,SET RMS=0
 1635  010454  005737  001376            POSRMS: TST     PEAK
 1636  010460  100002                            BPL     POSPEA
 1637  010462  005037  001376                    CLR     PEAK                    ;PEAK<0,SET PEAK=0
 1638  010466  013702  001374            POSPEA: MOV     RMS,R2
 1639  010472  004737  011504                    JSR     PC,DECTYP
 1640  010476  104401  012744                    TYPE    ,MESR
 1641  010502  013702  001376                    MOV     PEAK,R2
 1642  010506  004737  011504                    JSR     PC,DECTYP
 1643  010512  104401  012757                    TYPE    ,MESP
 1644  010516  004737  006406                    JSR     PC,TYPEDG
 1645  010522  104401  012401                    TYPE    ,CHAN
 1646  010526  013746  001362                    MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
 1647                                            ;;TYPE CHANL
 1648  010532  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
 1649  010534     002                            .BYTE   2               ;;TYPE 2 DIGIT(S)
 1650  010535     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
 1651  010536  023737  001374  011742            CMP     RMS,VNR         ;WITHIN LIMITS?
 1652  010544  003007                            BGT     ER
 1653  010546  023737  001376  011744            CMP     PEAK,VNP                ;WITHIN LIMITS?
 1654  010554  003003                            BGT     ER
 1655  010556  104401  012474                    TYPE    ,OKMSG
 1656  010562  000207                            RTS     PC
 1657  010564  104401  012505            ER:     TYPE    ,ERMSG
 1658  010570  000207                            RTS     PC
```

```
1659                                        ;;SUBROUTINES FOR NOISE TEST;;
1660  010572  005037  001374          NOIA: CLR     RMS                 ;CLEAR RMS VLAUE
1661  010576  005037  001376                CLR     PEAK                ;CLEAR PEAK VALUE
1662  010602  004537  006452          NOI1: JSR     R5,SARSUB           ;DO SAR ROUTINE AT 16%
1663  010606  000020                        16.
1664  010610  063737  001404  001374        ADD     DAC,RMS       ;ADD RESULT TO RMS
1665  010616  004537  006452                JSR     R5,SARSUB           ;DO SAR ROUTINE AT 84%
1666  010622  000124                        84.
1667  010624  163737  001404  001374        SUB     DAC,RMS       ;SUBTRACT RESULT FROM RMS
1668  010632  004537  006452                JSR     R5,SARSUB           ;DO SAR ROUTINE AT 1%
1669  010636  000001                        1
1670  010640  063737  001404  001376        ADD     DAC,PEAK            ;ADD RESULT TO PEAK
1671  010646  004537  006452                JSR     R5,SARSUB           ;DO SAR ROUTINE AT 99%
1672  010652  000143                        99.
1673  010654  163737  001404  001376        SUB     DAC,PEAK            ;SUBTRACT RESULT FROM PEAK
1674  010662  000207                        RTS     PC                  ;RETURN
1675
1676  010664  012537  001362          NOI8: MOV     (R5)+,CHANL         ;GET CHANNEL VALUE
1677  010670  063737  001332  001362        ADD     BASECH,CHANL
1678  010676  013737  001362  001360        MOV     CHANL,DUMMY         ;LOAD DUMMY CHANNEL
1679  010704  004737  006226                JSR     PC,GETEDG           ;GET EDGE VALUES
1680  010710  005037  001374                CLR     RMS           ;CLEAR RMS VALUE
1681  010714  005037  001376                CLR     PEAK          ;CLEAR PEAK VALUE
1682  010720  012737  000010  011006        MOV     #10.,10$      ;SET UP COUNTER
1683  010726  004737  010602          1$:   JSR     PC,NOI1       ;GET NOISE VALUES
1684  010732  005237  001410                INC     EDGE
1685  010736  005337  011006                DEC     10$
1686  010742  001371                        BNE     1$            ;REPEAT 8 TIMES
1687  010744  162737  000010  001410        SUB     #10,EDGE
1688  010752  006237  001374                ASR     RMS           ;SCALE IT TO 1 LSB=100.
1689  010756  005537  001374                ADC     RMS
1690  010762  006237  001376                ASR     PEAK
1691  010766  005537  001376                ADC     PEAK
1692  010772  012737  000010  006450        MOV     #8.,EDGFLG
1693  011000  004737  010436                JSR     PC,TYPRP                    ;TYPE RESULTS
1694  011004  000205                        RTS     R5            ;RETURN
1695  011006  000000                10$:    0                     ;COUNTER
1696
1697
1698                                        ;;RANDOM NUMBER GENERATOR;;
1699  011010  063737  001370  001366  RANDY: ADD     RNB,RNA
1700  011016  063737  001372  001366        ADD     RNC,RNA
1701  011024  005537  001366                ADC     RNA
1702  011030  063737  001366  001370        ADD     RNA,RNB
1703  011036  063737  001372  001370        ADD     RNC,RNB
1704  011044  005537  001370                ADC     RNB
1705  011050  063737  001366  001372        ADD     RNA,RNC
1706  011056  063737  001370  001372        ADD     RNB,RNC
1707  011064  005537  001372                ADC     RNC
1708  011070  000207                        RTS     PC
```

```
                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
011072  012500                  CONVRT: MOV     (R5)+,R0                ;GET CHANNEL VALUE
011074  063700  001332                  ADD     BASECH,R0
011100  010037  001362                  MOV     R0,CHANL
011104  000300                          SWAB    R0
011106  005037  001346                  CLR     TEMP
                                        ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
011122  010037  001426                  MOV     R0,MYTEMP
                                        ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
011136  012700  010000                  MOV     #10000,R0
011142  005300                  2$:     DEC     R0
011144  001376                          BNE     2$
011146  012777  001704  170302          MOV     #RETURN,@VECTOR                 ;LOAD VECTOR
011154  012700  000010                  MOV     #10,R0                  ;SET UP COUNTER
011160                          1$:
                                        ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
011170  052737  000001  001426          BIS     #1,MYTEMP
                                        ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
011206  005001                          CLR     R1
011210  105201                  10$:    INCB    R1
011212  001007                          BNE     11$
011214  012737  000200  001124          MOV     #BIT7,$GDDAT                    ;EXPECT DONe TO sET BY NOW
011222  013737  001426  001126          MOV     MYTEMP,$BDDAT
011230  104001                          ERROR   1                               ;DONE FAILED TO sET ON A/D
011232                          11$:
                                        ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
011242  105737  001426                  TSTB    MYTEMP
011246  100360                          BPL     10$
                                        ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
011260  063737  001426  001346          ADD     MYTEMP,TEMP
                                                                        ;WAIT FOR CONVERSION
                                                                        ;READ BUFFER
011266  005300                          DEC     R0
011270  001333                          BNE     1$                      ;DO 8 TIMES
011272  006237  001346                  ASR     TEMP                    ;AVERAGE VALUE
011276  006237  001346                  ASR     TEMP
011302  006237  001346                  ASR     TEMP
011306  005537  001346                  ADC     TEMP
011312  000205                          RTS     R5                      ;RETURN

                                        ;COMPARE $GDDAT AND $BDDAT;;
011314  012537  001124          COMPAR: MOV     (R5)+,$GDDAT                    ;GET GOOD DATA
011320  013537  001402                  MOV     @(R5)+,SPREAD                   ;GET SPREAD
011324  013737  001346  001126          MOV     TEMP,$BDDAT                     ;GET BAD(ACTUAL) DATA
011332  013701  001126                  MOV     $BDDAT,R1
011336  013700  001124                  MOV     $GDDAT,R0
```

```
 1763  011342  160100                        SUB     R1,R0           ;GET DIFFERENCE
 1764  011344  100001                        BPL     7$
 1765  011346  005400                        NEG     R0
 1766  011350  020037  001402        7$:     CMP     R0,SPREAD               ;COMPARE IT TO SPREAD
 1767  011354  003001                        BGT     10$             ;GO TO ERROR PRINTOUT
 1768  011356  005725                        TST     (P5)+           ;BUMP RETURN POINTER AROUND ERROR CALL
 1769  011360  000205                10$:    RTS     R5
```

```
1770                                    ;SUBROUTINE TO RESET & SET INTRPT. EN.;
1771  011362  004737  020422            RST:    JSR     PC,$RESET
1772  011366  052777  000100  167550            BIS     #100,@$TKS
1773  011374  005037  177776                    CLR     PSW
1774  011400  000207                            RTS     PC
1775
1776
1777
1778                                    ;SUBROUTINE LOADY;
1779  011402  005702                    LOADY:  TST     R2              ;ROUTINE TO LOAD VLAUE INTO R2
1780  011404  100001                            BPL     PLUSR2          ;AS A VT55 Y-VALUE
1781  011406  005002                            CLR     R2
1782  011410  020227  000353            PLUSR2: CMP     R2,#235.
1783  011414  002402                            BLT     LESS
1784  011416  012702  000353                    MOV     #235.,R2
1785  011422  010203                    LESS:   MOV     R2,R3
1786  011424  042702  177740                    BIC     #177740,R2
1787  011430  052702  000040                    BIS     #40,R2
1788  011434  105777  167510            B10:    TSTB    @$TPS           ;PRINT CHARACTER
1789  011440  100375                            BPL     B10
1790  011442  110277  167504                    MOVB    R2,@$TPB
1791  011446  006203                            ASR     R3
1792  011450  006203                            ASR     R3
1793  011452  006203                            ASR     R3
1794  011454  006203                            ASR     R3
1795  011456  006203                            ASR     R3
1796  011460  042703  177770                    BIC     #177770,R3
1797  011464  052703  000040                    BIS     #40,R3
1798  011470  105777  167454            B11:    TSTB    @$TPS           ;PRINT CHARACTER
1799  011474  100375                            BPL     B11
1800  011476  110377  167450                    MOVB    R3,@$TPB
1801  011502  000207                            RTS     PC
1802
1803
```

```
1804                                    ;;SUBROUTINE TO tYPE DECIMAL VALUE;;
1805                                    ;;IN R2 AS X.XX;;
1806  011504  005702                    DECTYP: TST     R2              ;TESt VALUE TO BE TYPED
1807  011506  100003                            BPL     POS
1808  011510  104401  012233                    TYPE    ,MINUS                  ;TYPE MINUS SIGN
1809  011514  005402                            NEG     R2
1810  011516  020227  001747            POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
1811  011522  003402                            BLE     OKAYD
1812  011524  012702  001747                    MOV     #999.,R2
1813  011530  105037  014574            OKAYD:  CLRB    ONES            ;CLEAR ONES
1814  011534  105037  014573                    CLRB    TENS            ;CLEAR TENs
1815  011540  105037  014571                    CLRB    HUNS            ;CLEAR HUNS
1816  011544  005702                    TESTR2: TST     R2                      ;CONVERT VALUE TO A DECiMAL VALUE
1817  011546  001424                            BEQ     TYPOUT
1818  011550  005302                            DEC     R2
1819  011552  105237  014574                    INCB    ONES
1820  011556  123727  014574  000012            CMPB    ONES,#10.
1821  011564  001367                            BNE     TESTR2
1822  011566  105037  014574                    CLRB    ONES
1823  011572  105237  014573                    INCB    TENS
1824  011576  123727  014573  000012            CMPB    TENs,#10.
1825  011604  001357                            BNE     TESTR2
1826  011606  105037  014573                    CLRB    TENS
1827  011612  105237  014571                    INCB    HUNS
1828  011616  000752                            BR      TESTR2
1829  011620  152737  000060  014571    TYPOUT: BISB    #60,HUNS                ;PREPARE FOR TYPOUT
1830  011626  152737  000060  014573            BISB    #60,TENS
1831  011634  152737  000060  014574            BISB    #60,ONES
1832  011642  104401  014571                    TYPE    ,HUNS           ;TYPE VALUE
1833  011646  000207                            RTS     PC
1834
1835  011650  012701  011742            WFADJ:  MOV     #VNR,R1                 ;SUBROUTINE TO SET UP LIMITS
1836  011654  005737  001332                    TST     BASECH                  ;TESTING AN AM11K?
1837  011660  001403                            BEQ     1$              ;;
1838  011662  012702  011774                    MOV     #VARLT3,R2              ;BASECH NOT ZERO, USE AM11K LIMITS
1839  011666  000410                            BR      3$              ;;
1840  011670  005737  001416            1$:     TST     WFTEST
1841  011674  001003                            BNE     2$
1842  011676  012702  011754                    MOV     #VARLT1,R2              ;WFTEST=0,USE NORMAL LIMITS
1843  011702  000402                            BR      3$
1844  011704  012702  011764            2$:     MOV     #VARLT2,R2              ;WFTEST=1,USE OPTION AREA LIMITS
1845  011710  012221                    3$:     MOV     (R2)+,(R1)+
1846  011712  005711                            TST     (R1)
1847  011714  100375                            BPL     3$
1848  011716  000207                            RTS     PC
```

```
1849  011720  000001                    V1:     1                       ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
1850  011722  000002                    V2:     2
1851  011724  000010                    V10:    10
1852  011726  000050                    V50:    50
1853  011730  000144                    V144:   144
1854  011732  000115                    V115:   115
1855  011734  000240                    V240:   240
1856  011736  000005                    V5:     5
1857  011740  000062                    V50D:   50.
1858
1859  011742  000000                    VNR:    0                       ;RMS NOISE LIMIT
1860  011744  000000                    VNP:    0                       ;PEAK NOISE LIMIT
1861  011746  000000                    VSET:   0                       ;INTER-CHANNEL SETTLING LIMIT
1862  011750  000000                    VLIN:   0                       ;RELATIVE ACCURACY ERROR LIMIT
1863  011752  100000                            BIT15
1864
1865  011754  000031                    VARLT1: 25.             ;.25 LSB,NORMAL LIMITS FOR SYSTEM
1866  011756  000310                            200.            ;2. LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
1867  011760  000144                            100.            ;1 LSB
1868  011762  000144                            100.            ;1 LSB
1869
1870  011764  000027                    VARLT2: 23.             ;.23 LSB,  TIGHTER LIMITS FOR OPTION
1871  011766  000226                            150.            ;1.5 LSB,  AREA USE ON SPEC TESTS
1872  011770  000132                            90.             ;.9 LSB
1873  011772  000132                            90.             ;.9 LSB
1874
1875  011774  000062                    VARLT3: 50.             ;.5 LSB,  LIMITS FOR AM11K TESTING
1876  011776  000310                            200.            ;2. LSB
1877  012000  000226                            150.            ;1.5 LSB
1878  012002  000226                            150.            ;1.5 LSB
1879
1880  012004  052777  000100  167132  AGATST: BIS     #100,@$TKS
1881  012012  000177  000000                JMP     @AGTST
1882  012016  001714                    AGTST:  BEGIN
```

```
                                        .SBTTL  END OF PASS ROUTINE
                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*TYPE "END PASS"
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO AGATST
                                        ;*IF IT IS DESIRED TO hAVE A BELL INDICATE THE "END OF PASS" LOCATION
                                        ;*$ENDMG CAN BE CHANGED TO 7.
012020                                  $EOP:
012020  000240                                  NOP
012022  005037  001102                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
012026  005037  001160                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
012032  005237  001202                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
012036  042737  100000  001202                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
012044  005327                                  DEC     (PC)+           ;;LOOP?
012046  000001                          $EOPCT: .WORD   1
012050  003015                                  BGT     $DOAGN          ;;YES
012052  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
012054  000001                          $ENDCT: .WORD   1
012056  012046                                  $EOPCT
012060  104401  012113                          TYPE    ,$ENDMG         ;;TYPE "END PASS"
012064  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
012070  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
012072  000005                                  RESET                   ;;CLEAR THE WORLD
012074  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
012076  000240                                  NOP                     ;;SAVE ROOM
012100  000240                                  NOP                     ;;FOR
012102  000240                                  NOP                     ;;ACT11
012104                                  $DOAGN:
012104  000137                                  JMP     @(PC)+          ;;RETURN
012106  012004                          $RTNAD: .WORD   AGATST
012110     377        377     000      $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
012113     015  042412  042116          $ENDMG: .ASCIZ  <15><12>/END PASS/
012120  050040  051501  000123
```

```
                                                .SBTTL          ASCII MESSAGES
012126  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST-- /
012134  020105  042524  052123
012142  026455  000040
012146  005015  042523  052124  SETMSG: .ASCIZ  <15><12>/SETTLING TEST-- TYPE DESIRED 'FROM' CHANNEL & CR: /
012154  044514  043516  052040
012162  051505  026524  020055
012170  054524  042520  042040
012176  051505  051111  042105
012204  023440  051106  046517
012212  020047  044103  047101
012220  042516  020114  020046
012226  051103  020072     000
012233     055     000          MINUS:  .BYTE   55,0
012235     077     000          QUEST:  .BYTE   77,0
012237     136     101     040  AMSG:   .BYTE   136,101,40,40,0
012242     040     000
012244     136     103     040  CMSG:   .BYTE   136,103,40,40,0
012247     040     000
012251     136     107     015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012254     012     123     127
012257     122     105     107
012262     072     000
012264  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
012272     000
012273     055  020055     000  DASH:   .ASCIZ  /-- /
012277     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012304  026455  053440  042111
012312  044124  005015     000
012317     103  000110          CH:     .ASCIZ  /CH/
012322  020040  020040     000  SPACE:  .ASCIZ  /  /
012327     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
012334  047117  041440  000110
012342  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
012350  047111  020107  051106
012356  046517  041440  000110
012364  040440  020124     000  ATMSG:  .ASCIZ  / AT /
012371     116  044517  042523  NOI:    .ASCIZ  /NOISE: /
012376  020072     000
012401     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
012406  040510  047116  046105
012414  000040
012416  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
012424  042516  005015     000
012431     057     000          SLASH:  .ASCIZ  #/#
012433     124  050131  020105  TOMsG:  .ASCIZ  /TYPE DESIRED 'TO' CHANNEL & CR: /
012440  042504  044523  042522
012446  020104  052047  023517
012454  041440  040510  047116
012462  046105  023040  041440
012470  035122  000040
012474  020040  020040  045517  CKMSG:  .ASCIZ  /    OK/<15><12>
012502  005015     000
```

```
1973  012505  040     025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
1974  012512  047522  025122  006452
1975  012520  000012
1976  012522  051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
1977  012530  042105  051440  040524
1978  012536  042524  051450  000051
1979  012544  047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
1980  012552  020127  036050  030440
1981  012560  031057  046040  041123
1982  012566  020051  052123  052101
1983  012574  024105  024523  005015
1984  012602  000
1985  012603  040     044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
1986  012610  024040  020076  020061
1987  012616  027461  020062  051514
1988  012624  024502  051440  040524
1989  012632  042524  051450  006451
1990  012640  000012
1991  012642  051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
1992  012650  051450  020051  044527
1993  012656  042504  020122  044124
1994  012664  047101  031040  046040
1995  012672  041123  000
1996  012675  040     052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
1997  012702  026505  044527  052104
1998  012710  024110  024523  047440
1999  012716  052125  044523  042504
2000  012724  025440  047440  020122
2001  012732  020055  027461  020062
2002  012740  051514  000102
2003  012744  046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
2004  012752  051515  020054  000     
2005  012757  040     051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
2006  012764  042520  045501  040440
2007  012772  020124  000
2008  012775  015     042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
2009  013002  047440  020106  047514
2010  013010  044507  020103  042524
2011  013016  052123  123
2012  013021  040     047117  040440  ONAD:   .ASCIZ  / ON AD11K AT /
2013  013026  030504  045461  040440
2014  013034  020124  000
2015  013037  040     042101  030461  MSG50:  .ASCIZ  / AD11K'S FOUND/<15><12>
2016  013044  023513  020123  047506
2017  013052  047125  006504  000012
2018  013060  005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
2019  013066  020062  051514  006502
2020  013074  005012  005012  005012
2021  013102  005012  005012  005012
2022  013110  030455  031057  051514          .ASCIZ  \-1/2LSB\
2023  013116  000102
2024
```

```
                                .EVEN
013120 044504 043106 051105 MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
013126 047105 044524 046101
013134 046040 047111 040505
013142 044522 054524 006472
013150 000012
013152 020040 020040 020040 MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
013160 020040 020040 020040
013166 020040 020040 020040
013174 020040 052123 052101
013202 026505 044527 052104
013210 020110 044504 052123
013216 044522 052502 044524
013224 047117 005015 005012
013232 020040 020043 043117        .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
013240 051440 040524 042524
013246 005123 005012 005012
013254 005012 005012 005012
013262 005012 005012 005012
013270 005012
013272 020040 020040 020040        .ASCII  /                                                STATE WIDTH (LSB)/<15>
013300 020040 020040 020040
013306 020040 020040 020040
013314 020040 020040 020040
013322 020040 020040 020040
013330 020040 020040 020040
013336 020040 020040 020040
013344 020040 020040 020040
013352 051440 040524 042524
013360 053440 042111 044124
013366 024040 051514 024502
013374 005015
013376 030040 020040 020040        .ASCIZ  # 0              1/2              1              1 1/2              2#
013404 020040 020040 020040
013412 020040 020040 027461
013420 020062 020040 020040
013426 020040 020040 020040
013434 020040 020061 020040
013442 020040 020040 020040
013450 020040 030440 030440
013456 031057 020040 020040
013464 020040 020040 020040
013472 020040 031040     000
013477     015 052012 050131 MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR DESIRED TEST: /
013504 020105 042514 052124
013512 051105 023040 041440
013520 020122 047506 020122
013526 042504 044523 042522
013534 020104 042524 052123
013542 020072     000
013545     122 046105 052101 MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
013552 053111 020105 041501
013560 052503 040522 054503
013566 006472 000012
```

```
 2079  013572  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
 2080  013600  054101  046511  046525
 2081  013606  040440  020124     000
 2082  013613     015  041412  046101  HEAD5:  .ASCII  <15><12>/CALIBRATION--/
 2083  013620  041111  040522  044524
 2084  013626  047117  026455
 2085  013632  051440  052105  041440  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
 2086  013640  040510  047116  046105
 2087  013646  044440  020116  053523
 2088  013654  020122  047514  020127
 2089  013662  054502  042524  005015
 2090  013670     000
 2091  013671     033  000132          CO:     .ASCIZ  <33><132>
 2092  013674  000055                  C1:     .ASCIZ  <55>
 2093  013676  031033     000          C2:     .ASCIZ  <33><62>
 2094  013701     112     000          C3:     .ASCIZ  <112>
 2095  013703     015  047412  043106  MOFSET: .ASCIZ  <15><12>/OFFSET =/
 2096  013710  042523  020124  000075
 2097  013716  046040  041123  000040  MLSB:   .ASCIZ  / LSB /
 2098  013724  040440  020124     000  MAT:    .ASCIZ  / AT /
 2099  013731     015  020012  047105  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
 2100  013736  042524  044522  043516
 2101  013744  052040  051505  020124
 2102  013752     000
 2103  013753     033     061     101  BUFF1:  .BYTE   33,61,101,61,111,62,114,41,60,45,63,51,66,55,71,61,74,110,41,40,112,0
 2104  013756     061     111     062
 2105  013761     114     041     060
 2106  013764     045     063     051
 2107  013767     066     055     071
 2108  013772     061     074     110
 2109  013775     041     040     112
 2110  014000     000
 2111  014001     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
 2112  014004     047     111     061
 2113  014007     104     050     065
 2114  014012     044     062     110
 2115  014015     040     040     102
 2116  014020     000
 2117  014021     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0
 2118  014024     112     033     061
 2119  014027     101     040     033
 2120  014032     062     000
 2121  014034  005015  046412  026504  HEAD1:  .ASCII  <15><12><12>#MD-11-DRLPK-A    AD11K/LPA-11 DIAGNOSTIC#<15><12>
 2122  014042  030461  042055  046122
 2123  014050  045520  040455  020040
 2124  014056  020040  042101  030461
 2125  014064  027513  050114  026501
 2126  014072  030461  042040  040511
 2127  014100  047107  051517  044524
 2128  014106  006503     012
 2129  014111     012  035101  040440          .ASCII  <12>/A: AUTO TESt/
 2130  014116  052125  020117  042524
 2131  014124  052123
 2132  014126  005015  035103  041440          .ASCII  <15><12>/C: CALIBRATION/
```

```
2133  014134  046101  041111  040522
2134  014142  044524  047117
2135  014146  005015  035114  046040          .ASCII  <15><12>/L: LOGIC TEST/
2136  014154  043517  041511  052040
2137  014162  051505     124
2138  014165     015  047012  020072          .ASCII  <15><12>/N: NOISE TEST/
2139  014172  047516  051511  020105
2140  014200  042524  052123
```

```
2141   014204   005015   035123   051440          .ASCII  <15><12>/S: SETTLE TEST/
2142   014212   052105   046124   020105
2143   014220   042524   052123
2144   014224   005015   035127   053440          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
2145   014232   040522   040520   047522
2146   014240   047125   020104   042524
2147   014246   052123   005015      000
2148   014253      015   051412   040524  EM1:    .ASCIZ  <15><12>/STATUS REG. ERROR/<15><12>
2149   014260   052524   020123   042522
2150   014266   027107   042440   051122
2151   014274   051117   005015      000
2152   014301      015   043012   044501  EM2:    .ASCIZ  <15><12>/FAILED TO INTERRUPT/<15><12>
2153   014306   042514   020104   047524
2154   014314   044440   052116   051105
2155   014322   052522   052120   005015
2156   014330      000
2157   014331      015   052412   042516  EM3:    .ASCIZ  <15><12>/UNEXPECTED INTERRUPT/<15><12>
2158   014336   050130   041505   042524
2159   014344   020104   047111   042524
2160   014352   051122   050125   006524
2161   014360   000012
2162   014362   005015   051105   047522  EM4:    .ASCIZ  <15><12>#ERROR ON A/D CHANNEL#<15><12>
2163   014370   020122   047117   040440
2164   014376   042057   041440   040510
2165   014404   047116   046105   005015
2166   014412      000
2167   014413      105   051122   041520  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/<15><12>
2168   014420   051440   051124   043505
2169   014426   042440   050130   041505
2170   014434   042524   020104   041501
2171   014442   052524   046101   005015
2172   014450      000
2173   014451      105   051122   041520  DH2:    .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOL#RANCE  ACTUAL/
2174   014456   020040   052123   042522
2175   014464   020107   020040   044103
2176   014472   047101   042516   020114
2177   014500   047040   046517   047111
2178   014506   046101   020040   047524
2179   014514   042514   040522   041516
2180   014522   020105   040440   052103
2181   014530   040525   000114
2182   014534   051105   050122   020103  DH3:    .ASCIZ  /ERRPC       STREG    ACTUAL/<15><12>
2183   014542   020040   020040   051440
2184   014550   051124   043505   020040
2185   014556   020040   041501   052524
2186   014564   046101   005015      000
```

```
2187  014571  000                       HUNS:    .BYTE   0
2188  014572  056                       DECPNT:  .BYTE   56
2189  014573  000                       TENS:    .BYTE   0
2190  014574  000     000               ONES:    .BYTE   0,0
2191                                    .EVEN
2192
2193  014576  001116  001316  001124   DT1:     $ERrPC, STREG, $GDDAT, $BDDAT,0
2194  014604  001126  000000
2195  014610  001116  001316  001362   DT2:     $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
2196  014616  001124  001402  001126
2197  014624  000000
2198  014626  001116  001316  001126   DT3:     $ERRPC,STREG,$BDDAT,0
2199  014634  000000
2200
2201  014636  000000                    DF1:     0
2202
2203
2204
2205
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*****************************************************************
                                .ENABL  LSB

                                .DSABL  LSB


                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;

014640  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014642  016666  000004  000002  MOV     4(SP),2(SP)     ;;SAVE THE PS
014650  105777  164270  1$:     TSTB    @$TKS           ;;WAIT FOR
014654  100375                  BPL     1$              ;;A CHARACTER
014656  117766  164264  000004  MOVB    @$TKB,4(SP)     ;;READ THE TTY
014664  042766  177600  000004  BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
014672  026627  000004  000023  CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014700  001013                  BNE     3$              ;;BRANCH IF NO
014702  105777  164236  2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014706  100375                  BPL     2$              ;;LOOP UNTIL ITS THERE
014710  117746  164232          MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014714  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
014720  022627  000021          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
014724  001366                  BNE     2$              ;;IF NOT DISCARD IT
014726  000750                  BR      1$              ;;YES, RESUME
014730  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
014736  002407                  BLT     4$              ;;BRANCH IF YES
014740  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
014746  003003                  BGT     4$              ;;BRANCH IF YES
014750  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
014756  000002          4$:     RTI                     ;;GO BACK TO USER
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

014760  010346          $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
014762  012703  015066  1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
014766  022703  015076  2$:     CMP     #$TTYIN+80.,R3  ;;BUFFER FULL?
014772  101405                  BLOS    4$              ;;BR IF YES
014774  104405                  RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
014776  112613                  MOVB    (SP)+,(R3)      ;;GET CHARACTER
015000  122713  000177  10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
015004  001003                  BNE     3$              ;;SKIP IF NOT
015006  104401  001170  4$:     TYPE    ,$QUES          ;;TYPE A '?'
015012  000763                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
```

```
2260  015014  111337  015064          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2261  015020  104401  015064                  TYPE    ,9$
2262  015024  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2263  015030  001356                          BNE     2$              ;;LOOP IF NOT RETURN
2264  015032  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2265  015036  104401  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
2266  015042  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2267  015044  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2268  015046  016666  000004  000002          MOV     4(SP),2(SP)     ;;       FIRST ASCII CHARACTER ON IT
2269  015054  012766  015066  000004          MOV     #$TTYIN,4(SP)
2270  015062  000002                          RTI                     ;;RETURN
2271  015064     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2272  015065     000                          .BYTE   0               ;;TERMINATOR
2273  015066  000010             $TTYIN:     .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2274  015076  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2275  015103     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2276  015110  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2277  015116  020075     000
2278  015121     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2279  015126  036440  000040
```

```
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

015132  011646          $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
015134  016666  000004  000002  MOV     4(SP),2(SP)             ;;INPUT NUMBER
015142  010046                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
015144  010146                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
015146  010246                  MOV     R2,-(SP)                ;;PUSH R2 ON STACK
015150  104406          1$:     RDLIN                           ;;READ AN ASCIZ LINE
015152  012600                  MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
015154  005001                  CLR     R1                      ;;CLEAR DATA WORD
015156  005002                  CLR     R2
015160  112046          2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
015162  001412                  BEQ     3$                      ;;IF ZERO GET OUT
015164  006301                  ASL     R1                      ;;*2
015166  006102                  ROL     R2
015170  006301                  ASL     R1                      ;;*4
015172  006102                  ROL     R2
015174  006301                  ASL     R1                      ;;*8
015176  006102                  ROL     R2
015200  042716  177770          BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
015204  062601                  ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
015206  000764                  BR      2$                      ;;LOOP
015210  005726          3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
015212  010166  000012          MOV     R1,12(SP)               ;;SAVE THE RESULT
015216  010237  015232          MOV     R2,$HIOCT
015222  012602                  MOV     (SP)+,R2                ;;POP STACK INTO R2
015224  012601                  MOV     (SP)+,R1                ;;POP STACK INTO R1
015226  012600                  MOV     (SP)+,R0                ;;POP STACK INTO R0
015230  000002                  RTI                             ;;RETURN
015232  000000          $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

015234                          $SCOPE:
015234  032777  040000  163676  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
015242  001114                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
015244  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
015246  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015252  012737  015272  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
015260  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
015264  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015270  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
015272  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
015274  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015300  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
015302                          6$:;#####END OF CODE FOR THE XOR TESTER#####
015302  032777  000400  163630          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
015310  001404                          BEQ     2$              ;;BR IF NO
015312  127737  163622  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
015320  001465                          BEQ     $OVER           ;;BR IF YES
015322  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
015326  001421                          BEQ     3$              ;;BR IF NO
015330  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015336  101015                          BHI     3$              ;;BR IF NO
015340  032777  001000  163572          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
015346  001404                          BEQ     4$              ;;BR IF NO
015350  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015356  000446                          BR      $OVER
015360  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
015364  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015370  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015372  032777  004000  163540  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015400  001011                          BNE     1$              ;;BR IF YES
015402  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015406  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015410  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015414  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015422  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015424  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015432  013737  015510  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015440  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
                                .SBTTL  ERROR HANDLER ROUTINE
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

015512                          $ERROR:
015512  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
015516  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015520  013777  001102  163414          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015526  032777  002000  163404          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015534  001402                          BEQ     1$              ;;NO - SKIP
015536  104401  001164                  TYPE    ,$BELL          ;;RING BELL
015542  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015546  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015552  162737  000002  001116          SUB     #2,$ERRPC
015560  117737  163332  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015566  032777  020000  163344          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015574  001004                          BNE     20$             ;;SKIP TYPEOUTS
015576  004737  015706                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
015602  104401  001171                  TYPE    ,$CRLF
015606                          20$:
015606  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015614  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
015616  113737  001114  015630          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
015624  004737  016342                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
015630  000             21$:    .BYTE   0
015631  000                     .BYTE   0
015632  000777          22$:    BR      22$             ;;APT ERROR LOOP
015634  005777  163300  2$:     TST     @SWR            ;;HALT ON ERROR
015640  100001                  BPL     3$              ;;SKIP IF CONTINUE
015642  000000                  HALT                    ;;HALT ON ERROR!
015644  032777  001000  163266  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
015652  001402                  BEQ     4$              ;;BR IF NO
015654  013716  001110          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
015660  005737  001162  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
015664  001402                  BEQ     5$              ;;BR IF NONE
```

```
2426  015666  013716  001162                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
2427  015672                        5$:
2428  015672  022737  012074  000042        CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
2429  015700  001001                        BNE     6$              ;;BRANCH IF NO
2430  015702  000000                        HALT                    ;;YES
2431  015704                        6$:
2432  015704  000002                        RTI                     ;;RETURN
2433                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2434
2435                                ;;*****************************************************************
2436                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2437                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2438                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2439
2440  015706                        $ERRTYP:
2441  015706  104401  001171                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2442  015712  010046                        MOV     R0,-(SP)        ;;SAVE R0
2443  015714  005000                        CLR     R0              ;;PICKUP THE ITEM INDEX
2444  015716  153700  001114                BISB    @#$ITEMB,R0
2445  015722  001004                        BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
2446                                                                ;;TYPE THE PC OF THE ERROR
2447  015724  013746  001116                MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
2448                                                                ;;ERROR ADDRESS
2449  015730  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2450  015732  000426                        BR      6$              ;;GET OUT
2451  015734  005300                1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
2452  015736  006300                        ASL     R0              ;;      WORK FOR THE ERROR TABLE
2453  015740  006300                        ASL     R0
2454  015742  006300                        ASL     R0
2455  015744  062700  001256                ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
2456  015750  012037  015760                MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
2457  015754  001404                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
2458  015756  104401                        TYPE                    ;;TYPE THE "ERROR MESSAGE"
2459  015760  000000                2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
2460  015762  104401  001171                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2461  015766  012037  015776        3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
2462  015772  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
2463  015774  104401                        TYPE                    ;;TYPE THE "DATA HEADER"
2464  015776  000000                4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
2465  016000  104401  001171                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2466  016004  011000                5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
2467  016006  001004                        BNE     7$              ;;GO TYPE THE DATA
2468  016010  012600                6$:     MOV     (SP)+,R0        ;;RESTORE R0
2469  016012  104401  001171                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2470  016016  000207                        RTS     PC              ;;RETURN
2471  016020                        7$:
2472  016020  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
2473  016022  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2474  016024  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
2475  016026  001770                        BEQ     6$              ;;BR IF NO
2476  016030  104401  016036                TYPE    ,8$             ;;TYPE TWO(2) SPACES
2477  016034  000771                        BR      7$              ;;LOOP
2478  016036  020040      000       8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
2479          016042                        .EVEN
```

```
                                .SBTTL  TYPE ROUTINE
                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
016042  105737  001157          $TYPE:  TSTB    $TPFLG           ;;IS THERE A TERmINAL?
016046  100002                  1$:     BPL     1$               ;;BR IF YES
016050  000000                          HALT                     ;;HALT HERE IF NO TERmINAL
016052  000430                          BR      3$               ;;LEAVE
016054  010046                          MOV     R0,-(SP)         ;;SAVE R0
016056  017600  000002                  MOV     @2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
016062  122737  000001  001214          CMPB    #APTENV,$ENv     ;;RUNNING IN APT MODE
016070  001011                          BNE     62$              ;;NO,GO CHECK FOR APT CONSOLE
016072  132737  000100  001215          BITB    #APTSPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
016100  001405                          BEQ     62$              ;;NO,GO CHECK FOR CONSOLE
016102  010037  016112                  MOV     R0,61$           ;;SETUP MESSAGE ADDRESS FOR APT
016106  004737  016332                  JSR     PC,$ATY3         ;;SPOOL MESSAGE TO APT
016112  000000                  61$:    .WORD   0                ;;MESSAGE ADDRESS
016114  132737  000040  001215  62$:    BITB    #APTCSUP,$ENvM   ;;APT CONSOLE SUPPRESSED
016122  001003                          BNE     60$              ;;YES,SKIP TYPE OUT
016124  112046                  2$:     MOVB    (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016126  001005                          BNE     4$               ;;BR IF IT ISN'T THE TERmINATOR
016130  005726                          TST     (SP)+            ;;IF TERMINATOR POP IT OFF THE STACK
016132  012600                  60$:    MOV     (SP)+,R0         ;;RESTORE R0
016134  062716  000002          3$:     ADD     #2,(SP)          ;;ADJUST RETURN PC
016140  000002                          RTI                      ;;RETURN
016142  122716  000011          4$:     CMPB    #HT,(SP)         ;;BRANCH IF <HT>
016146  001430                          BEQ     8$
016150  122716  000200                  CMPB    #CRLF,(SP)       ;;BRANCH IF NOT <CRLF>
016154  001006                          BNE     5$
016156  005726                          TST     (SP)+            ;;POP  <CR><LF> EQUIV
016160  104401                          TYPE                     ;;TYPE A CR AND LF
016162  001171                          $CRLF
016164  105037  016320                  CLRB    $CHARCNT         ;;CLEAR CHARACTER COUNT
016170  000755                          BR      2$               ;;GET NEXT CHARACTER
016172  004737  016254          5$:     JSR     PC,$TYPEC        ;;GO TYPE THIS CHARACTER
016176  123726  001156          6$:     CMPB    $FILLC,(SP)+     ;;IS IT TIME FOR FILLER CHARS.?
016202  001350                          BNE     2$               ;;IF NO GO GET NEXT CHAR.
016204  013746  001154                  MOV     $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
                                                                 ;;AND THE NULL CHAR.
016210  105366  000001          7$:     DECB    1(SP)            ;;DOES A NULL NEED TO BE TYPED?
016214  002770                          BLT     6$               ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
2534  016216  004737  016254                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2535  016222  105337  016320                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2536  016226  000770                          BR      7$              ;;LOOP
2537
2538                                  ;HORIZONTAL TAB PROCESSOR
2539
2540  016230  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2541  016234  004737  016254          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2542  016240  132737  000007  016320          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2543  016246  001372                          BNE     9$              ;;TAB STOP
2544  016250  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2545  016252  000724                          BR      2$              ;;GET NEXT CHARACTER
2546  016254  105777  162670          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2547  016260  100375                          BPL     $TYPEC
2548  016262  116677  000002  162662          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2549  016270  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2550  016276  001003                          BNE     1$              ;;BRANCH IF NO
2551  016300  105037  016320                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2552  016304  000406                          BR      $TYPEX          ;;EXIT
2553  016306  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2554  016314  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2555  016316  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2556  016320  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2557  016322  000207          $TYPEX: RTS     PC
2558
2559                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2560
2561                                  ;;**************************************************************
2562  016324  112737  000001  016570  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2563  016332  112737  000001  016566  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2564  016340  000403                          BR      $ATYC
2565  016342  112737  000001  016570  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2566  016350                          $ATYC:
2567  016350  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2568  016352  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2569  016354  105737  016566                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2570  016360  001450                          BEQ     5$              ;;IF NOT:  BR
2571  016362  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2572  016370  001031                          BNE     3$              ;;IF NOT:  BR
2573  016372  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2574  016400  001425                          BEQ     3$              ;;IF NOT:  BR
2575  016402  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2576  016406  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2577  016414  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2578  016420  001375                          BNE     1$              ;;IF NOT:  WAIT
2579  016422  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2580  016426  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2581  016430  001376                          BNE     2$
2582  016432  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2583  016436  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2584  016440  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2585  016444  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2586  016452  000413                          BR      5$
2587  016454  017637  000004  016500  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
```

```
2588  016462  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2589  016470  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2590  016474  004737  016042                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
2591  016500  000000                  4$:     .WORD   0
2592  016502                          5$:
2593  016502  105737  016570          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2594  016506  001416                          BEQ     12$             ;;IF NOT:  BR
2595  016510  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
2596  016514  001413                          BEQ     12$             ;;IF NOT:  BR
2597  016516  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2598  016522  001375                          BNE     11$             ;;IF NOT:  WAIT
2599  016524  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
2600  016532  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2601  016540  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2602  016544  105037  016570          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2603  016550  105037  016567                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2604  016554  105037  016566                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2605  016560  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2606  016562  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2607  016564  000207                          RTS     PC              ;;RETURN
2608  016566     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
2609  016567     000          $LFLG:  .BYTE   0               ;;LOG FLAG
2610  016570     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
2611        016572                  .EVEN
2612        000200          APTSIZE=200
2613        000001          APTENV=001
2614        000100          APTSPOOL=100
2615        000040          APTCSUP=040
2616
2617                        ;*
2618                        ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
2619                        ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
2620                        ;*NEXT WE WILL INIT BOTH UPROCESSORS
2621                        ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
2622                        ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
2623                        ;*
2624                        ;*      CALL=   JSR     R5,$LPAI
2625                        ;*              .WORD   0               ;ADDR. OF DEVICE ADDRESS.
2626                        ;* ROUTINES REQUIRED:   .LOADLP
2627                        ;* PROGRAMS REQUIRED:   DRLPX2
2628                        ;*
2629
2630                        ;*              ;RETURNS WITH $AERR=1 IF SLAVE
2631                        ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
2632                        ;*
2633  016572          $LPAI:
2634  016572  013746  000004          MOV     4,-(SP)
2635
2636  016576  000413                  BR      31$             ;FIELD DOES NOT HAVE A BUS SWITCH TO
2637                                                          ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
2638                                                          ;BRANCH ARROUD THE NEXT CODE THAT
2639                                                          ;WORKS BASED ON A BUS SWITCH.
2640                                                          ;CODE LEFT IN HERE FOR IN HOUSE
2641                                                          ;PERSONAL WHO MAY PATCH THIS BRANCH
```

```
                                                        ;INSTRUCTION TO A <NOP> OCTAL <240>
                                                        ;IN ORDER TO RUN PROGRAM WITH A SWITCH.
                                                        ;
                                                        ;NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
                                                        ;TEST EQUIPMENT ONLY IT CONNECTS
                                                        ;THE UNIBUS TO THE I/O BUS FOR
                                                        ;CERTAIN TESTING.
016600  012737  016624  000004          MOV     #30$,4
016606  005237  170000                  INC     170000
016612  104401  016620                  TYPE    ,65$            ;;TYPE ASCIZ STRING
016616  000401                          BR      64$             ;;GET OVER THE ASCIZ
                              ;;65$:    .ASCIZ  <7>##
016622                        64$:
016622  000401                          BR      31$
016624  022626                30$:      CMP     (SP)+,(SP)+
016626  012637  000004        31$:      MOV     (SP)+,4         ;ALL THIS JUNK MUST BE REMOVED!!
016632  005037  017450                  CLR     $AERR
016636  004537  017452                  JSR     R5,$LOAD        ;LOAD MICRO-CODE.
016642  000000G                         .WORD   DRLPX2          ;FILE "DRLPX2.OBJ"
                                                        ;
016644  052777  040000  162564          BIS     #BIT14,@KMADO   ;ISSUE KMC+DMC INIT.
                                                        ;
016652                        1$:
                                                        ;"HANGS" HERE THEN KMC-11 ERROR.
016652  010146                          MOV     R1,-(SP)
016654  005001                          CLR     R1
016656  005201                2$:       INC     R1              ;STALL FOR DMC-UP
016660  001376                          BNE     2$
016662  012777  104000  162546          MOV     #BIT15!BIT11,@KMADO     ;SET RUN, AND ENABLE ARBITRATION.
016670  105201                25$:      INCB    R1
016672  001376                          BNE     25$
                                                        ;
016674  032777  000040  162534          BIT     #BIT5,@KMADO    ;SLAVE READY? (READING IPBM SR)
016702  001401                          BEQ     3$
                                                        ;FATAL LPA-11 ERROR SLAVE NOT READY.
016704  104000                          ERROR
                                                        ;
016706  012777  000004  162526  3$:     MOV     #4,@KMAD2       ;READ FAST PATH
016714                        4$:
016714  004537  020362                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
                                                        ;
016720  104000                          ERROR                   ;/TIME-OUT ERROR
                                                        ;/WE FAILED TO COMPLETE
                                                        ;/CURRENT OPERATION.
                                                        ;/CONTINUES IN THIS LOOP
                                                        ;/WOULD MAKE US "HANG" HERE
                                                        ;
016722  000774                          BR              4$
                                                        ;
                                                        ;/RETURNS HERE-FROM-TIMED OUT.
016724  122777  000377  162510          CMPB    #377,@KMAD2     ;WAIT TILL KMC DONE COMMAND.
016732  001370                          BNE     4$
016734  122777  000377  162504          CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
016742  001001                          BNE     35$
```

```
2696  016744  104000                          ERROR                   ;IPBM ERROR (SLAVE SIDE)
2697                                                                  ;YOU MUST RUN IPBM DIAGNOSTIC.
2698
2699  016746  122777  000004  162472  35$:    CMPB    #4,@KMAD4       ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
2700  016754  001543                          BEQ     5$              ;YES-CONTINUE.
2701  016756  005227  177777                  INC     #-1
2702  016762  001140                          BNE     5$
2703  016764  005227  177777                  INC     #-1
2704  016770  001135                          BNE     5$
2705  016772  104401  017000                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2706  016776  000440                          BR      66$             ;;GET OVER THE ASCIZ
2707                                  ;;67$:  .ASCIZ  <200>"W A R N I N G  THIS PROGRAM WAS DESIGNED TO RUN WITH VERsION 4"
2708  017100                          66$:
2709  017100  104401  017106                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2710  017104  000430                          BR      68$             ;;GET OVER THE ASCIZ
2711                                  ;;69$:  .ASCIZ  <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
2712  017166                          68$:
2713  017166  104401  017174                  TYPE    ,71$            ;;TYPE ASCIZ STRING
2714  017172  000434                          BR      70$             ;;GET OVER THE ASCIZ
2715                                  ;;71$:  .ASCIZ  <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
2716  017264                          70$:
2717
2718  017264  112737  177777  017416  5$:     MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
2719  017272  012501                          MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
2720  017274  021127  000000          6$:     CMP     (R1),#0         ;TERM REACHED?
2721  017300  001444                          BEQ     10$
2722  017302  105237  017416                  INCB    11$
2723  017306  113777  017416  162132          MOVB    11$,@KMAD4      ;FIFO DATA
2724  017314  004737  017420                  JSR     PC,20$          ;ISSUE SEND
2725  017320  112177  162122                  MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
2726  017324  004737  017420                  JSR     PC,20$          ;ISSUE SEND
2727  017330  112177  162112                  MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
2728  017334  004737  017420                  JSR     PC,20$
2729
2730  017340  032777  000002  162070  7$:     BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
2731  017346  001374                          BNE     7$              ;=1 NO DATA. =0 DATA.
2732  017350  112777  000002  162064          MOVB    #2,@KMAD2       ;READ FIFO.
2733
2734  017356                          8$:
2735  017356  004537  020362                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2736
2737  017362  104000                          ERROR                   ;/TIME-OUT ERROR
2738                                                                  ;/WE FAILED TO COMPLETE
2739                                                                  ;/CURRENT OPERATION.
2740                                                                  ;/CONTINUES IN THIS LOOP
2741                                                                  ;/WOULD MAKE US "HANG" HERE
2742
2743  017364  000774                          BR              8$
2744
2745                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
2746  017366  122777  000377  162046          CMPB    #377,@KMAD2     ;WAIT FOR READ.
2747  017374  001370                          BNE     8$
2748  017376  105777  162044                  TSTB    @KMAD4          ;WAS A ZERO RETURNED?
2749  017402  001734                          BEQ     6$              ;YES GET NEXT ADDR.
```

```
                                                        ;SLAVE WILL RETURN CODE 0 IF
017404  005237  017450                INC     $AERR     ;DEV PRESENT.   ELSE
                                                        ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
017410  005041                        CLR     -(1)      ;GET RID OF REFERENCE TO BAD ADDR.
017412  012601                10$:    MOV     (SP)+,R1
017414  000205                        RTS     R5        ;RETURN ALL ADDR. CHECKED.

017416  000000                11$:    .WORD   0         ;HOLDS DAC CODE PLUS OFFSET
                                                        ;TO SLAVES ADDR. TABLE.

017420  112777  000003  162014  20$:    MOVB    #3,@KMAD2 ;ISSUE FIFO WRITE
017426                        21$:
017426  004537  020362                JSR     R5, $TOUT ;-TOUT-CHECK FOR TIMEOUT

017432  104000                        ERROR             ;/TIME-OUT ERROR
                                                        ;/WE FAILED TO COMPLETE
                                                        ;/CURRENT OPERATION.
                                                        ;/CONTINUES IN THIS LOOP
                                                        ;/WOULD MAKE US "HANG" HERE

017434  000774                        BR          21$

                                                        ;/RETURNS HERE-FROM-TIMED OUT.
017436  122777  000377  161776        CMPB    #377,@KMAD2 ;KMC CODE WILL RETURN A "377"
017444  001370                        BNE     21$       ;WHEN DONE COMMAND.
017446  000207                        RTS     PC

017450  000000                $AERR:  .WORD   0         ;=0 IF ADDR. LIST OK,=1 IF BAD.

                                        ;*
                                        ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
                                        ;*      CALL =  JSR     R5,$LOAD
                                        ;*              .WORD   XX              ;ADDR. OF MICRO CODE.
                                        ;*              ;RETURNS HERE
                                        ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
                                        ;*

017452  010446                $LOAD:  MOV     R4,-(SP)  ;SAVE R4.
017454  010046                        MOV     R0,-(SP)  ;SAVE R0.
017456  012500                1$:     MOV     (5)+,R0   ;GET PROG. ADDR.
017460  005077  161752                CLR     @KMAD0    ;CLEAR CSR
017464  005077  161756                CLR     @KMAD4    ;CLEAR CRAM ADDR.
017470  052777  002000  161740  2$:     BIS     #2000,@KMAD0 ;SELECT CRAM.
017476  012077  161750                MOV     (0)+,@KMAD6 ;WRITE DATA.
017502  052777  020000  161726        BIS     #20000,@KMAD0 ;SET CRAM WRITE
017510  005077  161722                CLR     @KMAD0    ;DISABLE CRAM.
017514  005277  161726                INC     @KMAD4    ;UPDATE CRAM ADDR.
017520  021027  177777                CMP     (0),#-1   ;ALL DONE?
017524  001361                        BNE     2$        ;NO LOOP.
017526  005077  161714                CLR     @KMAD4    ;CLEAR CRAM ADDR.
017532  016500  177776                MOV     -2(5),R0  ;GET MICRO CODE ADDR.

017536  052777  002000  161672  3$:     BIS     #2000,@KMAD0 ;SELECT CRAM
017544  022077  161702                CMP     (R0)+,@KMAD6 ;DATA OK?
```

```
2804  017550  001013                        BNE     5$              ;NO - REPORT AN ERROR.
2805  017552  021027  177777                CMP     (0),#-1         ;ALL DONE?
2806  017556  001405                        BEQ     4$              ;YES - EXIT
2807  017560  005077  161652                CLR     @KMAD0          ;NO - DESELECT CRAM.
2808  017564  005277  161656                INC     @KMAD4          ;UPDATE CRAM ADDR.
2809  017570  000762                        BR      3$
2810
2811  017572  012600              4$:       MOV     (SP)+,R0        ;RESTORE R0
2812  017574  012604                        MOV     (SP)+,R4        ;RESTORE R4
2813  017576  000205                        RTS     R5              ;EXIT
2814
2815  017600                      5$:                               ;COME HERE ON LOAD ERROR
2816  017600  005745                        TST     -(5)
2817  017602  105204                        INCB    R4              ;UPDATE ERROR COUNTER.
2818  017604  100324                        BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
2819  017606  000000                        HALT                    ;MICRO CODE LOAD ERROR.
2820                                                                ;KMC-11 FAULT. YOU COULD TRY
2821  017610  000722                        BR      1$              ;TO PRESS CONTINUE TO GIVE IT
2822                                                                ;ANOTHER CHANCE, BUT I DOUBT
2823                                                                ;THAT THAT WOULD WORK. SINCE I'VE
2824                                                                ;ALREADY GIVEN IT 177 (OCTAL) CHANCES.
2825                                                                ;TRY RUNNING THE KMC-11 DIAGNOSTIC.
2826
2827
2828                                        ;
2829                                        ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
2830                                        ;*
2831                                        ;*      CALL =  JSR     R5,$TLKW
2832                                        ;*              .WORD   0               ;OFFSET OF DEVICE ADDR.
2833                                        ;*              .WORD   0               ;DATA TO BE WRITTEN
2834                                        ;*
2835  017612  010046              $TLKw:    MOV     R0,-(SP)        ;SAVE R0
2836  017614  012500                        MOV     (5)+,R0         ;GET DEVICE OFFSET
2837  017616  052700  000340                BIS     #340,R0         ;ADD WRITE CODE.
2838  017622  004737  020074                JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
2839  017626  010037  017720                MOV     R0,W1
2840  017632  010077  161610                MOV     R0,@KMAD4
2841  017636  112777  000005  161576        MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
2842  017644  004737  020074                JSR     PC,$LPw         ;WAIT FOR RDY
2843  017650  011537  017722                MOV     (5),W2
2844  017654  112577  161566                MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
2845
2846  017660  112777  000005  161554        MOVB    #5,@KMAD2       ;FP WRITE
2847  017666  004737  020074                JSR     PC,$LPw
2848  017672  111537  017724                MOVB    (5),W3
2849  017676  112577  161544                MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
2850  017702  112777  000005  161532        MOVB    #5,@KMAD2
2851  017710  004737  020074                JSR     PC,$LPW
2852  017714  012600                        MOV     (SP)+,R0
2853  017716  000205                        RTS     R5              ;EXIT DONe.
2854  017720  000000              W1:       0
2855  017722  000000              W2:       0
2856  017724  000000              W3:       0
2857
```

```
                                        ;*
                                        ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
                                        ;*
                                        ;*      CALL =  JSR     R5,$TLKR
                                        ;*              .WORD   0               ;OFFSET OF DEVICE
                                        ;*              ;RETURNS HERE
                                        ;*DATA IN WORD $DATR
                                        ;*
017726  010046                  $TLKr:  MOV     R0,-(SP)        ;SAVE R0
017730  012500                          MOV     (5)+,R0         ;GET OFFSET
017732  052700  000300                  BIS     #300,R0         ;ADD READ CODE
017736  004737  020074                  JSR     PC,$LPw         ;WAIT TILL READY
017742  110077  161500                  MOVB    R0,@KMAD4
017746  112777  000005  161466          MOVB    #5,@KMAD2       ;ISSUE WRITE FP
017754  004737  020074                  JSR     PC,$LPW
017760  010037  020070                  MOV     R0,RD1
017764                          1$:
017764  004537  020362                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
017770  104000                          ERROR                   ;/TIME-OUT ERROR
                                                                ;/WE FAILED TO COMPLETE
                                                                ;/CURRENT OPERATION.
                                                                ;/CONTINUES IN THIS LOOP
                                                                ;/WOULD MAKE US "HANG" HERE
017772  000774                          BR              1$
                                                                ;/RETURNS HERE-FROM-TIMED OUT.
017774  032777  000040  161434          BIT     #BIT5,@KMAD0    ;FAST PATH GOT DATA?
020002  001370                          BNE     1$
020004  112777  000004  161430          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
020012  004737  020074                  JSR     PC,$LPW
020016  117737  161424  020072          MOVB    @KMAD4,$DATR    ;GET LOW BYTE
020024                          2$:
020024  004537  020362                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
020030  104000                          ERROR                   ;/TIME-OUT ERROR
                                                                ;/WE FAILED TO COMPLETE
                                                                ;/CURRENT OPERATION.
                                                                ;/CONTINUES IN THIS LOOP
                                                                ;/WOULD MAKE US "HANG" HERE
020032  000774                          BR              2$
                                                                ;/RETURNS HERE-FROM-TIMED OUT.
020034  032777  000040  161374          BIT     #BIT5,@KMAD0    ;FAST PATH READY?
020042  001370                          BNE     2$
020044  112777  000004  161370          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
020052  004737  020074                  JSR     PC,$LPw
020056  117737  161364  020073          MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
020064  012600                          MOV     (SP)+,R0
020066  000205                          RTS     R5
020070  000000                  RD1:    0
```

```
2912  020072  000000                    $DATR:  .WORD   0
2913                                      ;
2914                                      ;THIS ROUTINE WAITS FOR KMC-CODE TO bECOME READY AS WELL
2915                                      ;AS FAST PATH TO BE READ.
2916                                      ;
2917                                      ;       CALL =  JSR     PC,$LPW
2918                                      ;
2919                                      ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
2920                                      ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
2921                                      ;
2922
2923  020074  010146                    $LPW:   MOV     R1,-(SP)        ;SAVE R1
2924  020076  005001                            CLR     R1
2925  020100  122777  000377  161334    1$:     CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
2926  020106  001403                            BEQ     2$
2927  020110  005201                            INC     R1              ;TIME OUT?
2928  020112  001372                            BNE     1$
2929  020114  000411                            BR      10$
2930
2931  020116  032777  000020  161312    2$:     BIT     #BIT4,@KMAD0    ;FAST PATH READ?
2932  020124  001403                            BEQ     3$
2933  020126  005201                            INC     R1              ;NO - TIME OUT?
2934  020130  001372                            BNE     2$
2935  020132  000402                            BR      10$             ;YES - REPORT AN ERROR
2936
2937  020134  012601                    3$:     MOV     (SP)+,R1        ;RESTORE R1
2938  020136  000207                            RTS     PC              ;EXIT
2939
2940  020140                            10$:
2941  020140  104401  020146                    TYPE    ,65$            ;;TYPE ASCIZ STRING
2942  020144  000407                            BR      64$             ;;GET OVER THE ASCIZ
2943                                    ;;65$:  .ASCIZ  <200>#LPA-11 FAULT#
2944  020164                            64$:
2945
2946  020164  000000                    11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
2947  020166  000776                            BR      11$             ;DIAGNOSTICS.
2948
2949
2950
2951                                      ;*
2952                                      ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
2953                                      ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
2954                                      ;*
2955                                      ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
2956                                      ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
2957                                      ;* THAT ADDRESS.
2958                                      ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
2959                                      ;* $TLKW
2960                                      ;*
2961
2962  020170  010046                    $OUTLP: MOV     R0,-(SP)        ;SAVE R0
2963  020172  010146                            MOV     R1,-(SP)        ;SAVE R1
2964
2965  020174  012700  001464                    MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
```

```
                                        CLR     R1
                                1$:     TST     (0)             ;TERMINATOR REACHED?
                                        BEQ     10$             ;YES NEXT STEP.
                                        CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
                                        BEQ     2$
                                        INC     R1
                                        BR      1$

                                2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
                                        TST     (5)+
                                        MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
                                        JSR     R5,$TLKW        ;DO WRITE
                                3$:     .WORD   0               ;DEVICE OFFSET
                                4$:     .WORD   0               ;DATA TO BE WRITTEN.
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R0
                                        RTS     R5
                                10$:    MOV     @(5),(0)+       ;SAVE ADDR.
                                        CLR     (0)
                                        JSR     R5,$LPAI
                                        .WORD   .DVLS
                                        BR      2$

                                        ;*
                                        ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
                                        ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
                                        ;*
                                        ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
                                        ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
                                        ;*WITH THE NEW ADDR.
                                        ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
                                        ;*$TLKR
                                        ;*      CALL THROUGH    MOVEI   DATA,ADDR.
                                        ;*              WHICH EQUALS:
                                        ;*                      JSR     R5,$INLP
                                        ;*                      .WORD   XX      ADDR OF DEVICE
                                        ;*                      .WORD   YY      ADDR TO sTORE READ DATA.

                                $INLP:  MOV     R0,-(SP)        ;SAVE R0
                                        MOV     R1,-(SP)        ;SAVE R1

                                        MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
                                        CLR     R1
                                1$:     TST     (0)             ;EOL REACHED?
                                        BEQ     10$             ;YES - DEFINE NEW ADDR.

                                        CMP     @(5),(0)+       ;ADDR. MATCH?
                                        BEQ     2$
                                        INC     R1
                                        BR      1$

                                2$:     MOV     R1,3$           ;SAVE LIST OFFSET
                                        TST     (5)+
                                        JSR     R5,$TLKR        ;GO READ DEVICE
```

```
3020                    020330                  $OFS=.
3021    020330  000000                  3$:     .WORD   0               ;OFFSET OF DEVICE
3022
3023    020332  013735  020072                  MOV     $DATR,@(5)+     ;STORE DATA.
3024    020336  012601                          MOV     (SP)+,R1        ;RESTORE R1
3025    020340  012600                          MOV     (SP)+,R0        ;RESTORE R2
3026    020342  000205                          RTS     R5              ;EXIT
3027
3028    020344  017520  000000          10$:    MOV     @(5),(0)+
3029    020350  005010                          CLR     (0)
3030    020352  004537  016572                  JSR     R5,$LPAI
3031    020356  001464                          .WORD   .DVLS
3032    020360  000756                          BR      2$
3033                                            ;*
3034                                            ;*$TOUT ROUTINE USED TO WATCH IF
3035                                            ;*      WE'RE IN A LOOP TOO-LONG
3036                                            ;*      CALL=   JSR R5, $TOUT
3037                                            ;*              ERROR X     ;RETURNS HERE ON TIMEOUT
3038                                            ;*              BR
3039                                            ;*              ;RETURNS HERE NO ERROR
3040                                            ;*
3041
3042    020362  020537  020416          $TOUt:  CMP     R5,$SAD         ;SAME ADDR?
3043    020366  001405                          BEQ     1$
3044    020370  010537  020416                  MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
3045    020374  005037  020420                  CLR     $CNT            ;CLR CNT AT ADDR.
3046    020400  000403                          BR      2$
3047    020402  005237  020420          1$:     INC     $CNT            ;OVERFLOW?
3048    020406  100402                          BMI     3$              ;YES-ERROR RETURN
3049    020410  062705  000004          2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
3050    020414  000205                  3$:     RTS     R5              ;RETURN.
3051
3052    020416  000000                  $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
3053    020420  000000                  $CNT:   .WORD   0               ;# OF TIMES AT ADDR.
3054
3055                                            ;*
3056                                            ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
3057                                            ;*USE FOR A RESET.  FIRST WE DO A RESET INSTRUCTION.
3058                                            ;*THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
3059                                            ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
3060                                            ;*
3061                                            ;*      CALL=JSR        PC,$RESET       ;REPLACES "RESET INSTRUCTION
3062                                            ;*                      ;RETURNS HERE.
3063                                            ;*
3064    020422  000005                  $RESET: RESET                   ;RESET THE WORLD.
3065
3066                                    ;*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
3067    020434  005737  017450                  TST     $AERR           ;IF NO ERROR,LOOP
3068    020440  001004                          BNE     10$             ;THERE WAS AN ERROR.
3069    020442  062737  000002  020456          ADD     #2,2$           ;UPDATE DEVICE ADDR.
3070                                                                    ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
3071                                                                    ;IF 2$ CONTAINED A VALID ADDR,WE
3072                                                                    ;MUST KEEP TRYING UNTIL WE GENERATE
3073                                                                    ;AN INVALID ADDR.
```

```
3074  020450  000764                        BR      $RESET
3075  020452                          10$:
3076  020452  000207                        RTS     PC
3077  020454  000000                  1$:   .WORD   0               ;JUNK LOC.
3078  020456  160000                  2$:   .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
3079
3080
3081                                        ;
3082                                        ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
3083                                        ;        IS NOT TIME DEPENDENT CODE SENCE
3084                                        ;        NOT USED TO GET SPECIFIC TIME BUT
3085                                        ;        JUST A LITTLE DELAY.
3086
3087                                        ;        THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
3088                                        ;        THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
3089                                        ;
3090                                        ;
3091                                        ;        CALL=   JSR PC,  SDELAY
3092                                        ;
3093  020460                          SDELAY:
3094  020460  005737  020542                TST     RTCCSR          ;CLOCK PRESENT?
3095  020464  100016                        BPL     10$
3096  020466  012737  000002  020532        MOV     #2,TIME
3097  020474  052777  000115  000040        BIS     #115,@RTCCSR    ;START CLOCK
3098  020502  005037  177776               CLR     PS
3099  020506  005737  020532          1$:   TST     TIME
3100  020512  001375                        BNE     1$
3101  020514  005077  000022                CLR     @RTCCSR         ;STOP CLOCK
3102
3103  020520  000207                        RTS PC
3104  020522  105237  020532          10$:  INCB    TIME
3105  020526  001375                        BNE     10$
3106  020530  000207                        RTS     PC
3107
3108  020532  000000                  TIME: .WORD   0
3109
3110  020534  005337  020532          CLKINT: DEC   TIME
3111  020540  000002                        RTI
3112  020542  000000                  RTCCSR: .WORD 0               ;CLOCK CSR IF USED.
3113                                        ;
3114                                        ;*
3115                                        ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
3116                                        ;*ANY DEVICE ON THE I/O BUS
3117                                        ;*USER MUST START AT THIS ADDR.
3118                                        ;*HE MUST SAY EITHER "E" FOR EXAMINE, OR "D" FOR DEPOSIT.
3119                                        ;*"E" IS DEFAULT.
3120                                        ;*NEXT, HE MUST SUPPLY AN ADDR.
3121                                        ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
3122                                        ;*WILL OCCUR.
3123
3124  020544                          $UTK:
3125  020544  005037  001464                CLR     .DVLS
3126  020550                          21$:
3127  020550  104401  020556                TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
3128  020554  000405                          BR      64$             ;;GET OVER THE ASCIZ
3129                                  ;;65$:    .ASCIZ  <200>#E OR D?#
3130  020570                          64$:
3131  020570  105777  160350          1$:     TSTB    @$TKS
3132  020574  100375                          BPL     1$
3133  020576  117737  160344  020720          MOVB    @$TKB,20$       ;GET INPUT
3134  020604  104401  020720                  TYPE,   20$             ;ECHO, NEXT MESSAGE.
3135  020610  142737  000240  020720          BICB    #240,20$        ;STRIP PARITY, LC
3136  020616  104407                          RDOCT                   ;GET ADDR.
3137  020620  012637  020716                  MOV     (SP)+,14$
3138  020624  123727  020720  000104          CMPB    20$,#'D         ;DEPOSIT?
3139  020632  001411                          BEQ     10$
3140
3141  020634  004537  020266                  JSR     R5,$INLP        ;GET DATA
3142  020640  020716                  2$:     .WORD   14$
3143  020642  020654                          .WORD   5$
3144
3145  020644  013746  020654                  MOV     5$,-(SP)        ;;SAVE 5$ FOR TYPEOUT
3146  020650  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3147  020652  000736                          BR      21$             ;LOOP.
3148  020654  000000                  5$:     .WORD   0
3149
3150  020656                          10$:
3151  020656  104401  020664                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3152  020662  000404                          BR      66$             ;;GET OVER THE ASCIZ
3153                                  ;;67$:    .ASCIZ  <200>#DATA= #
3154  020674                          66$:
3155  020674  104407                          RDOCT
3156  020676  012637  020714                  MOV     (SP)+,13$
3157
3158  020702  004537  020170          11$:    JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
3159  020706  020716                  12$:    .WORD   14$             ;DEVICE ADDR.
3160  020710  020714                          .WORD   13$             ;DATA
3161  020712  000716                          BR      21$
3162
3163  020714  000000                  13$:    .WORD   0
3164  020716  000000                  14$:    .WORD   0
3165  020720  100001  042504  044526  20$:    .ASCIZ  <1><200>#DEVICE ADDR= #
3166  020726  042503  040440  042104
3167  020734  036522  000040
3168                                          .EVEN
3169
3170
3171
3172                                  ;
3173                                  ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
3174                                  ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
3175                                  ;TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
3176                                  ;SAMPLE TAKEING PURPOSES.
3177                                  ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
3178                                  ;       A/D CSR IN BSEL 4,AND 5.
3179                                  ;       (2) HE MUST CALL THIS ROUTINE:
3180                                  ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
3181                                  ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
```

```
                                        ;                ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
                                        ;                              ;(UNTILL ONE DOES A RESET)
                                        ;
                                        ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
                                        ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR.!
                                        ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
                                        ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
                                        ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
                                        ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
                                        ;
020740  012537  020750          $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
020744  004537  020266                  JSR     R5,$INLP
020750  000000                  1$:     .WORD   0
020752  021046                          .WORD   10$
020754  113777  020330  160470          MOVB    $OFS,@KMAD6
020762  113777  020330  160464          MOVB    $OFS,@KMAD7
020770  013737  020750  021010          MOV     1$,2$
020776  062737  000002  021010          ADD     #2,2$
021004  004537  020266                  JSR     R5,$INLP
021010  000000                  2$:     .WORD   0
021012  021046                          .WORD   10$
021014  113777  020330  160422          MOVB    $OFS,@KMAD3
021022  152777  000340  160422          BISB    #340,@KMAD6
021030  152777  000300  160416          BISB    #300,@KMAD7
021036  152777  000300  160400          BISB    #300,@KMAD3
021044  000205                          RTS     R5
021046  000000                  10$:    .WORD   0
```

```
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                ;;**************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT

021050  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
021054  116637  000001  021273        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
021062  112637  021275                MOVB    (SP)+,$OMoDE+1  ;;NUMBER OF DIGITS TO TYPE
021066  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
021072  000406                        BR      $TYPON
021074  112737  000001  021273 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
021102  112737  000006  021275        MOVB    #6,$OMoDE+1     ;;SET FOR SIX(6) DIGITS
021110  112737  000005  021272 $TYPON: MOVB    #5,$OCnT        ;;SET THE ITERATION COUNT
021116  010346                        MOV     R3,-(SP)        ;;SAVE R3
021120  010446                        MOV     R4,-(SP)        ;;SAVE R4
021122  010546                        MOV     R5,-(SP)        ;;SAVE R5
021124  113704  021275                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
021130  005404                        NEG     R4
021132  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
021136  110437  021274                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
021142  113704  021273                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
021146  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
021152  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
021154  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
021156  000404                        BR      3$              ;;GO DO MSB
021160  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
021162  006105                        ROL     R5
021164  006105                        ROL     R5
021166  010503                        MOV     R5,R3
021170  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
021172  105337  021274                DECB    $OMoDE          ;;TYPE THIS DIGIT?
021176  100016                        BPL     7$              ;;BR IF NO
021200  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
021204  001002                        BNE     4$              ;;TEST FOR 0
```

```
3264   021206  005704                          TST     R4              ;;SUPPRESS THIS 0?
3265   021210  001403                          BEQ     5$              ;;BR IF YES
3266   021212  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3267   021214  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3268   021220  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3269   021224  110337  021270                  MOVB    R3,8$           ;;SAVE FOR TYPING
3270   021230  104401  021270                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
3271   021234  105337  021272          7$:     DECB    $OCNT           ;;COUNT BY 1
3272   021240  003347                          BGT     2$              ;;BR IF MORE TO DO
3273   021242  002402                          BLT     6$              ;;BR IF DON.
3274   021244  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3275   021246  000744                          BR      2$              ;;GO DO THE LAST DIGIT
3276   021250  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3277   021252  012604                          MOV     (SP)+,R4        ;;RESTORE R4
3278   021254  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3279   021256  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3280   021264  012616                          MOV     (SP)+,(SP)
3281   021266  000002                          RTI                     ;;RETURN
3282   021270     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3283   021271     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3284   021272     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3285   021273     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3286   021274  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  TRAP DECODER

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                        ;*GO TO THAT ROUTINE.

021276  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
021300  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
021304  005740                          TST     -(R0)           ;;BACKUP BY 2
021306  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
021310  006300                          ASL     R0              ;;POSITION FOR INDEXING
021312  016000  021332                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
021316  000200                          RTS     R0              ;;GO TO ROUTINE

                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

021320  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
021322  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWn
021330  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE "TRAP" INSTRUCTION.

                                        ;       ROUTINE
                                        ;       -------
021332  021320                  $TRPAD: .WORD   $TRAP2
021334  016042                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
021336  021074                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
021340  021050                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
021342  021110                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)

021344  014640                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
021346  014760                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
021350  015132                          $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
021352  012737  021516  000024  $PWRDN:  MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST UP
021360  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
021366  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
021370  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
021372  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
021374  010346                          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
021376  010446                          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
021400  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
021402  017746  157532                  MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
021406  010637  021522                  MOV     SP,$SAVR6         ;;SAVE SP
021412  012737  021424  000024          MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
021420  000000                          HALT
021422  000776                          BR      .-2               ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
021424  012737  021516  000024  $PWRUP:  MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWn
021432  013706  021522                  MOV     $SAVR6,SP         ;;GET SP
021436  005037  021522                  CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
021442  005237  021522          1$:     INC     $SAVR6            ;;WAIT FOR THE INC
021446  001375                          BNE     1$                ;;OF  WORD
021450  012677  157464                  MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
021454  012605                          MOV     (SP)+,R5          ;;POP STACK INTO R5
021456  012604                          MOV     (SP)+,R4          ;;POP STACK INTO R4
021460  012603                          MOV     (SP)+,R3          ;;POP STACK INTO R3
021462  012602                          MOV     (SP)+,R2          ;;POP STACK INTO R2
021464  012601                          MOV     (SP)+,R1          ;;POP STACK INTO R1
021466  012600                          MOV     (SP)+,R0          ;;POP STACK INTO R0
021470  012737  021352  000024          MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
021476  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
021504  104401                          TYPE                      ;REPORT THE POWER FAILURE
021506  021524          $PWRMG: .WORD   $POWER                    ;;POWER FAIL MESSAGE POINTER
021510  012716                          MOV     (PC)+,(SP)        ;;RESTART AT BEG2
021512  002404          $PWRAD: .WORD   BEG2                      ;;RESTART ADDRESS
021514  000002                          RTI
021516  000000          $ILLUP: HALT                              ;;THE POWER UP SEQUENCE WAS STARTED
021520  000776                          BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
021522  000000          $SAVR6: 0                                 ;;PUT THE SP HERE
021524  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
021532  000122
                                        .EVEN
                                .EVEN
021534  000310          DIST:   .BLKW   200.                      ;STATE-WIDTH DISTRIBUTION
022354  010000          BUFFER: .BLKW   4096.                     ;BUFFER AREA

        000001                  .END
```

```
ABASE = 170400            162#   267    308    358    359
ACDW1 = 000000            267    310
ACDW2 = 000000            267
ACPUOP= 000000            267    282
ADBUFF  001320            359#   700    717    745    796    803    876    889   1048   1109*  1117*  1118*  1195
                         1248   1273   1322   1717   1746
ADDW0 = 000000            267
ADDW1 = 000000            267
ADDW10= 000000            267
ADDW11= 000000            267
ADDW12= 000000            267
ADDW13= 000000            267
ADDW14= 000000            267
ADDW15= 000000            267
ADDW2 = 000000            267
ADDW3 = 000000            267
ADDW4 = 000000            267
ADDW5 = 000000            267
ADDW6 = 000000            267
ADDW7 = 000000            267
ADDW8 = 000000            267
ADDW9 = 000000            267
ADEVCT= 000000            267    273
ADEVM = 000000            267    309
AENV  = 000000            267    278
AENVM = 000000            267    279
AFATAL= 000000            267    270
AGAIN   007122           1383#  1415
AGATST  012004           1880#  1915
AGTST   012016           1073*  1091*  1103*  1881   1882#
AMADR1= 000000            267    295
AMADR2= 000000            267    299
AMADR3= 000000            267    302
AMADR4= 000000            267    305
AMAMS1= 000000            267    289
AMAMS2= 000000            267    297
AMAMS3= 000000            267    300
AMAMS4= 000000            267    303
AMSG    012237            451   1935#
AMSGAD= 000000            267    275
AMSGLG= 000000            267    276
AMSGTY= 000000            267    269
AMTYP1= 000000            267    290
AMTYP2= 000000            267    298
AMTYP3= 000000            267    301
AMTYP4= 000000            267    304
APASS = 000000            267    272
APRIOR= 000300            164#   267    360
APTCSU= 000040           2510   2615#
APTENV= 000001           2411   2503   2571   2613#
APTSIZ= 000200            516   2612#
APTSPO= 000100           2505   2573   2614#
AROUND  007262           1411   1414#
ASKCH   013632           1147   2085#
```

```
ASWREG= 000000            267     280
ATESTN= 000000            267     271
ATMSG   012364           1189    1956#
AUNIT = 000000            267     274
AUSWR = 000000            267     281
AVECT1= 140340            163#    267     306     361     424     425
AVECT2= 000000            267     307
BASEBR  001322            360#   1121*   1122*
BASECH  001332            364#    786     947     948    1191    1224    1225    1372    1677    1711    1836
BEGIN   001714            184     473#   1882
BEGINA  005156            580    1077#
BEGINC  004656            583    1012#
BEGINL  002710            632#   1070    1081
BEGINN  005540            589    1145#
BEGINS  005610            592    1155#
BEGINW  005250            595    1095#
BEGIN2  001722            186     475#
BEGL    005114            571     586   1067#
BEG2    002404            185     448    567#   3364
BITPNT  001412            388#   1314*  1319    1346   1347*
BIT0  = 000001            135#    706     725     734
BIT00 = 000001            125#    135
BIT01 = 000002            124#    134
BIT02 = 000004            123#    133
BIT03 = 000010            122#    132
BIT04 = 000020            121#    131
BIT05 = 000040            120#    130
BIT06 = 000100            119#    129
BIT07 = 000200            118#    128
BIT08 = 000400            117#    127    2347
BIT09 = 001000            116#    126    2355    2421
BIT1  = 000002            134#   2730
BIT10 = 002000            115#   2399
BIT11 = 004000            114#   1308    2362    2670
BIT12 = 010000            113#   1612
BIT13 = 020000            112#   1020    1049    2406
BIT14 = 040000            111#    642    2333    2662
BIT15 = 100000            110#    673     693     733    1863    2670
BIT2  = 000004            133#
BIT3  = 000010            132#
BIT4  = 000020            131#    666     788     792    2931
BIT5  = 000040            130#    658    2674    2887    2904
BIT6  = 000100            129#    650
BIT7  = 000200            128#    687     733     792    1734
BIT8  = 000400            127#    638
BIT9  = 001000            126#    680     737     791
BK3     005642           1162#   1173
BPTVEC= 000014             142#
BUFFER  022354           1355    1400*  1402*   1417    1445    1514    1526    1528    1556    1567    1568    1569    1591
                         1594    3374#
BUFF1   013753           1497    2103#
BUFF2   014001           1590    2111#
BUMPAD  005322           1071    1089    1101    1106#
BYPASS  005474           1114    1128#
```

```
B10     011434          1788#   1789
B11     011470          1798#   1799
CH      012317          1022    1949#
CHAN    012401          1645    1959#
CHANL   001362           376#    947*  1163*  1248   1326   1618*  1619   1646   1676*  1677*  1678   1712*  2195
CHANNL  007042          1371#
CH1     001350           371#   1148*  1149*  1152   1158*  1184   1190   1216   1222*  1224*
CH2     001352           372#   1161*  1163   1176   1212   1223*  1225*
CLEAR1  006770          1357#   1359
CLEAR2  007034          1368#   1370
CLKINT  020534          3110#
CMSG    012244           445    1937#
CNNO    006746          1307*   1311*  1325   1351#
COMPAR  011314           773     816    827    839    852    864    880    904    916    927    938    964    1758#
CONV    006300          1258#   1275
CONVR   007150          1387    1390#
CONVRT  011072           771     814    825    837    850    862    877    890    914    925    936    1195   1710#
CR      =  000015         50#   2549   2559
CRLF    =  000200         51#   2520   2559
C0      013671           543    2091#
C1      013674          1292    2092#
C2      013676          1506    1596   2093#
C3      013701          1593    2094#
DAC     001404           385#    952    955   1215   1219   1315*  1319*  1322   1346*  1664   1667   1670   1673
DASH    012273          1452    1945#
DAWAIT  004646           876     889    913   1007#
DDISP   =  177570         57#    248    504
DECONT  014572          1457*   1486*  2188#
DECTYP  011504           962    1174   1453   1459   1467   1471   1474   1485   1535   1639   1642   1806#
DELAY   001406           386#   1322*  1323*
DELAY1  007230          1396    1404#
DELAY2  007236          1398    1405   1406#
DELAY3  007144          1388#   1389
DELCLR  010342          1495    1507   1597   1606#
DF1     014636           335     342    348    354   2201#
DH1     014413           333    2167#
DH2     014451           352    2173#
DH3     014534           340     346   2182#
DIFLIN  006750          1004    1353#
DISPLA  001142           248#    504*   512*  1051*  2377*  2398*
DISPRE  000174           181#    512
DIST    021534          1360    1428*  1498   3373#
DONE    012416          1962#
DRLPX2=  ******  G        14#   2660
DSWR    =  177570         56#    247    503
DT1     014576           334    2193#
DT2     014610           353    2195#
DT3     014626           341     347   2198#
DUMMY   001360           375#    948*  1212*  1216*  1328   1619*  1678*
EDGE    001410           387#    949*  1230*  1233*  1256*  1273*  1276*  1277*  1278*  1279*  1283   1339   1684*
                        1687*
EDGFLG  006450          1171*   1238*  1289   1299#  1622*  1692*
EDINT   001430           395#   1068*  1078*  1096*  1100*
EMTVEC=  000030           145#    487*   488*
```

```
EM1     014253              332    2148#
EM2     014301              339    2152#
EM3     014331              345    2157#
EM4     014362              351    2162#
ER      010564             1652    1654    1657#
ERMSG   012505              969    1206    1463    1478    1490    1551    1657    1973#
ERR     006052             1203    1206#
ERRVEC= 000004              138#    501     502*    513*    2338    2339*   2341*   2344*
FIRST   001342              368#   1366*   1440    1442*
FIXADR  005372             1107    1115#
FIXONE  005376              559    1116#
FLAG    001400              383#    538*    555*    1493    1554
GETDAT  010120             1558#   1566
GETEDG  006226             1164    1226    1245#   1620    1679
GMSG    012251              459    1939#
GNS   = ****** U            180    2653    2707    2711    2715    2943    3129    3153    3318    3319    3320    3321    3324
                           3325    3326
HAFMSG  012675             1487    1996#
HALF    007600             1479    1481#
HEAD1   014034              557    2121#
HEADS   013613             1016    2082#
HT    = 000011               48#   2518    2559
HUNS    014571             1815*   1827*   1829*   1832    2187#
INRNGE  007324             1424    1427#
IOTVEC= 000020              143#    485*    486*
ISERV   001550              440#    561
KBVECT  001334              365#    560    1130
KMAD0   001436              407#    528    2662*   2670*   2674    2730    2790*   2792*   2794*   2795*   2802*   2807*   2887
                           2904    2931
KMAD1   001440              410#    529
KMAD2   001442              412#   1332*   1333    1336*   1337    1392*   1393    2679*   2692    2732*   2746    2760*   2773
                           2841*   2846*   2850*   2872*   2889*   2906*   2925
KMAD3   001444              414#   3203*   3206*
KMAD4   001446              416#   1331*   1335*   1339    1391*   1395    2694    2699    2723*   2725*   2727*   2748    2791*
                           2796*   2799*   2808*   2840*   2844*   2849*   2871*   2891    2908
KMAD5   001450              418#
KMAD6   001452              420#   2793*   2803    3196*   3204*
KMAD7   001454              422#    533    3197*   3205*
LAST    007442             1438    1455#
LEND    004644             1003    1005#
LESS    011422             1783    1785#
LF    = 000012               49#   2553    2559
LINEA   013572             1536    2079#
LOAD    010320             1592    1595    1599#
LOADY   011402             1501    1503    1602    1779#
LOADO   010324             1600#   1604
LO2     010316             1555    1598#
LPADH   001450              417#
LPADL   001446              415#
LPCI    001436              406#
LPCO    001442              411#
LPMR    001440              409#
LPMS1   001452              419#
LPMS2   001454              421#
```

```
LPSO    001444              413#
LSB     012327             1175    1951#
LSBMSG  012264             1454    1943#
MAT     013724             1181    2098#
MAX     001420              391#    884#    892     896*   1574*   1579    1581*   1585
MAXTST  010216             1577    1579#
MEND    012775             1082    2008#
MESP    012757             1643    2005#
MESR    012744             1640    2003#
METST   013731             2099#
MIN     001414              389#    897#    957    1573*   1576    1578*   1584
MINUS   012233             1808    1933#
MLSB    013716              963    2097#
MOFSET  013703              960    2095#
MSG16   013152             1496    2031#
MSG18   013060             1589    2018#
MSG20   013120             1353    2026#
MSG21   013545             1513    2075#
MSG50   013037              617    2015#
MSG71   013477              572    2068#
MYTEMP  001426              394#    684#    687     693*    696     700     706*    709     713     717     725*    728     731
                            734*    737     745     788*    791     793*    796     803     804*    807     873*    876     886*
                            889    1013*   1016    1030*   1033    1038*   1041    1044    1048   1192*   1195   1250*   1253
                           1262*   1265    1268    1273    1376*   1379    1717*   1720   1728*   1731    1735    1742    1746
NAR     007510             1464    1466#
NARMSG  012544             1468    1979#
NARROW  001340              367#   1365*   1432*   1466    1481
NBEXT   001354              373#    601*    608*    612     618*    619    1106    1113*   1126*
NEXT    007126             1384#   1410
NMBEXT  001356              374#    619*   1126
NOI     012371             1631    1957#
NOIA    010572             1621    1660#
NOIMSG  012126             1146    1921#
NOITST  010400             1151    1618#
NOI1    010602             1662#   1683
NOI8    010664              978     980     982    1676#
NONE    001706              456     458     470#
NOTNAR  007360             1431    1437#
NOTNEW  010010             1523    1528#
NOTOK   007254             1408    1412#
NOVT55  002332              546     549     554     557#    628
NXTCMP  010202             1575#   1583
NXTCVT  006634             1330#   1343
NXTSTA  007752             1515#   1529
NXTY1   007706             1500#   1505
NXT8    010164             1571#   1588
OFFERR  004512              967     969#
OFFOK   004520              968     971#
OKAY    007246             1401    1403    1409#   1413
OKAYD   011530             1811    1813#
OKMSG   012474              971    1204    1465    1480    1492    1553    1655    1971#
ONAD    013021             2012#
ONES    014574             1813*   1819*   1820    1822*   1831*   2190#
OUT     001424              393#   1354*   1363*   1425*   1470    1473    1476    1483
```

```
OUTMSG  012642          1475    1991#
PC     =%000007           69#     447*    453*    544*    547*    550*    559*    566*    620*    631*    639*    651*    659*
                         667*    674*    696*    718*    740*    747*    760*    799*    876*    889*    911*    913*    962*
                        1004*   1005*   1010*   1069*   1070*   1071*   1080*   1081*   1088*   1089*   1098*   1099*   1101*
                        1138*   1164*   1166*   1167*   1172*   1174*   1182*   1205*   1207*   1220*   1226*   1229*   1239*
                        1280*   1298*   1384*   1453*   1459*   1467*   1471*   1474*   1485*   1495*   1501*   1503*   1507*
                        1535*   1592*   1595*   1597*   1598*   1602*   1605*   1616*   1620*   1621*   1623*   1639*   1642*
                        1644*   1656*   1658*   1674*   1679*   1683*   1693*   1708*   1771*   1774*   1801*   1833*   1848*
                        1899*   1902*   1909*   1914    2408*   2414*   2470*   2508*   2527*   2534*   2541*   2555*   2557*
                        2590*   2607*   2724*   2726*   2728*   2775*   2838*   2842*   2847*   2851*   2870*   2873*   2890*
                        2907*   2938*   3076*   3103*   3106*   3363
PEAK    001376           382#    1635    1637*   1641    1653    1661*   1670*   1673*   1681*   1690*   1691*
PERCNT  001422           392#    1304*   1305*   1306*   1312*   1313*   1344
PIRQ   = 177772           55#
PIRQVE= 000240           149#
PLUS    007772          1520    1522#
PLUSR2  011410          1780    1782#
POPRO   001702           468#    471
POS     011516          1807    1810#
POSPEA  010466          1636    1638#
POSRMS  010454          1633    1635#
POSR2   005700          1168    1170#
PR0    = 000000            72#
PR1    = 000040            73#
PR2    = 000100            74#
PR3    = 000140            75#
PR4    = 000200            76#
PR5    = 000240            77#
PR6    = 000300            78#
PR7    = 000340            79#
PS     = 177776            52#      53    3098*
PSW    = 177776            53#     575*   1017*   1773*
PWRVEC= 000024           144#     491*    492*   3331*   3332*   3341*   3347*   3359*   3360*
QUEST   012235           470      596    1934#
RANDY   011010          1384    1699#
RBEG    001730           474     476#
RDCHR  = 104405          2254    3324#
RDLIN  = 104406           573    2295    3325#
RDOCT  = 104407           466    1157    1160    3136    3155    3326#
RD1     020070          2874*   2911#
READ    007276          1418#   1456
RELACC  007736          1494    1511#
REST1   002336           540     558#
RESVEC= 000010           139#
RET     006446          1290    1298#
RETERR  003434           758     760#
RETURN  001704           469#    1382    1723
RMS     001374           381#    1632    1634*   1638    1651    1660*   1664*   1667*   1680*   1688*   1689*
RNA     001366           378#     563*   1385    1699*   1700*   1701*   1702    1705
RNB     001370           379#     564*   1699    1702*   1703*   1704*   1706
RNC     001372           380#     565*   1700    1703    1705*   1706*   1707*
RST     011362           447      453    1771#
RTCCSR  020542          3094    3097*   3101*   3112#
R0     =%000000           60#     440     441*    442*    443     449     455     468*    526     528*    531*    532     537*
```

```
                    545     548     551     553     560*    561*    562*    576*    577*    578     581     584     587
                    590     593     622*    625*    629*    630*    680*    688*    700*    717*    737*    738*    745*
                    791*    797*    803*    957*    958*    959     1007*   1008*   1018*   1019*   1023    1029*   1030
                   1048*   1051    1054    1128*   1132    1134*   1135    1136    1139    1140    1248*   1249*   1250
                   1253*   1254*   1257*   1274*   1318*   1341*   1344    1355*   1357*   1360*   1368*   1371*   1372*
                   1373*   1376    1379    1381*   1414*   1416*   1455*   1484*   1488    1498*   1500    1514*   1515
                   1518*   1525    1528    1556*   1558    1564*   1567*   1572    1575    1591*   1594*   1600    1601
                   1606*   1608*   1710*   1711*   1712    1713*   1717    1720*   1721*   1724*   1749*   1762*   1763*
                   1765*   1766    1906*   1909    2292    2296*   2299    2315*   2442    2443*   2444*   2451*   2452*
                   2453*   2454*   2455*   2456    2461    2466*   2468*   2472    2474    2501    2502*   2507    2512
                   2515*   2567    2575*   2579    2580    2582*   2583*   2584    2606*   2788    2789*   2800*   2803
                   2811*   2835    2836*   2837*   2839    2840    2852*   2867    2868*   2869*   2871    2874    2909*
                   2962    2965*   2981*   3004    3007*   3025*   3295    3296*   3297    3298*   3299*   3300*   3301*
                   3333    3358*
R1      =%000001     61#     527     529*    533     536*    1059*   1060*   1129*   1130    1132*   1133*   1134    1139
                   1140    1325*   1342*   1356*   1358*   1361*   1369*   1383*   1406*   1409*   1417*   1418    1444
                   1499*   1504*   1511*   1517*   1518    1519    1557*   1565*   1570*   1587*   1599*   1603*   1607*
                   1610*   1731*   1732*   1761*   1763    1835*   1845*   1846    2293    2297*   2301*   2303*   2305*
                   2308*   2311    2314*   2568    2605*   2666    2667*   2668*   2671*   2719*   2720    2725    2727
                   2754*   2923    2924*   2927*   2933*   2937*   2963    2966*   2971*   2974    2980*   3005    3008*
                   3014*   3017    3024*   3334    3357*
R2      =%000002     62#     885*    886     897     898     908*    909     961*    1033*   1062*   1165*   1169*   1170
                   1215*   1219*   1227*   1234    1236*   1237    1385*   1386*   1388*   1395*   1397    1399*   1400*
                   1402*   1404    1418*   1419*   1420*   1421*   1422*   1423    1427*   1428*   1429*   1430    1433
                   1437    1458*   1466*   1469*   1470*   1473*   1481*   1482*   1483*   1484    1500*   1502*   1515*
                   1516*   1517    1534*   1558*   1559*   1560*   1561*   1562*   1563*   1564    1571*   1582*   1600*
                   1638*   1641*   1779    1781*   1782    1784*   1785    1786*   1787*   1790    1806    1809*   1810
                   1812*   1816    1818*   1838*   1842*   1844*   1845    2294    2298*   2302*   2304*   2306*   2312
                   2313*   3335    3356*
R2POS    006210    1235    1237#
R3      =%000003     63#     901*    903     1228*   1231*   1283*   1284    1291*   1293    1362*   1407*   1444*   1445*
                   1446*   1447    1512*   1522    1524*   1530*   1531*   1532*   1533*   1534    1549    1572*   1573
                   1574    1575*   1576    1578    1579    1581    1785*   1791*   1792*   1793*   1794*   1795*   1796*
                   1797*   1800    2250    2251*   2252    2255*   2256    2260    2262    2264*   2266*   3243    3252*
                   3258*   3259*   3262*   3267*   3268*   3269    3278*   3336    3355*
R4      =%000004     64#     879*    902     903*    1170*   1202    1237*   1519*   1521*   1522    1524    1568*   1584*
                   1586    2787    2812*   2817*   3244    3246*   3247*   3248*   3249    3250*   3264    3266*   3274*
                   3277*   3337    3354*
R5      =%000005     65#     605*    684*    687*    693*    696*    700*    709*    713*    717*    728*    731*    737*
                    745*    754*    757*    771*    773*    791*    796*    803*    807*    814*    816*    825*    827*
                    837*    839*    850*    852*    862*    864*    876*    877*    880*    889*    890*    904*    914*
                    916*    925*    927*    936*    938*    950*    953*    964*    978*    980*    982*    990*    994*
                   1016*   1033*   1038*   1041*   1044*   1048*   1151*   1195*   1213*   1217*   1222    1223    1240*
                   1248*   1253*   1262*   1265*   1268*   1273*   1304    1316*   1322*   1349*   1374*   1379*   1525*
                   1526*   1527*   1537    1543*   1544    1569*   1585*   1586    1618    1624*   1662*   1665*   1668*
                   1671*   1676    1694*   1710    1717*   1720*   1728*   1731*   1742*   1746*   1755*   1758    1759
                   1768    1769*   2659*   2681*   2735*   2755*   2762*   2813*   2853*   2876*   2893*   2910*   2977*
                   2982*   2985*   3019*   3026*   3030*   3042    3044    3049*   3050*   3067*   3141*   3158*   3193*
                   3200*   3207*   3245    3251*   3253*   3255*   3256*   3257*   3258    3276*   3338    3353*
R6      =%000006     66#     479*    480*    481
R7      =%000007     67#
SARSUB   006452     950     953     1213    1217    1304#   1662    1665    1668    1671
SAR1     006524    1309    1310    1314#
SDELAY   020460    3093#
```

```
SETAA   006160          1229*  1232
SETCH   012342          1183   1953*
SETMSG  012146          1156   1924*
SET1A   006060          1166   1167   1212*  1229
SET8    006122           990    994   1222*
SHIFT   006736          1345   1347*
SKIPST  001344           369*  1367*  1435*  1458   1461
SKPMSG  012522          1460   1976*
SLASH   012431           465   1542   1964*
SP     =%000006           68*   440*   446*   452*   460*   467    468    483*   501*   509*   513    526*   527*
                         536    537    568*   576    612*   627    759*   1023*  1054*  1083*  1115*  1158   1161
                        1176*  1184*  1197*  1284*  1293*  1447*  1537*  1544*  1646*  2222*  2223*  2226*  2227*
                        2228   2232*  2233*  2234   2237   2239   2241*  2250*  2255   2266   2267*  2268*  2269*
                        2290*  2291*  2292*  2293*  2294*  2296   2299*  2307*  2308   2310   2311*  2313   2314
                        2315   2338*  2341   2343   2344   2373   2374   2378*  2403   2423*  2426*  2442*  2447*
                        2468   2472*  2501*  2502   2512*  2514   2515   2516*  2518   2520   2522   2528   2530*
                        2532*  2540*  2544   2548   2549   2553   2567*  2568*  2575   2576*  2587   2588*  2589*
                        2599   2600*  2605   2606   2634*  2656   2657   2666*  2754   2787*  2788*  2811   2812
                        2835*  2852   2867*  2909   2923*  2937   2962*  2963*  2980   2981   3004*  3005*  3024
                        3025   3137   3145*  3156   3235*  3236   3237   3238*  3243*  3244*  3245*  3251   3276
                        3277   3278   3279*  3280*  3295*  3296   3306*  3307*  3333*  3334*  3335*  3336*  3337*
                        3338*  3339*  3340   3348*  3352   3353   3354   3355   3356   3357   3358   3363*
SPACE   012322          1053   1950*
SPREAD  001402           384*  1759*  1766   2195
STACK = 001100            43*   446    452    483    568
STATE   012277          1443   1946*
STKLMT= 177774            54*
STREG   001316           358*   684    687    693    696    709    713    728    731    737    754    757    791
                         807   1016   1033   1038   1041   1044   1083   1108*  1116*  1253   1262   1265   1268
                        1317   1375   1379   1720   1728   1731   1742   2193   2195   2198
SWDIST  007654          1491   1493*
SWR     001140           247*   457    460    467*   481    503*   505    511*   518*   1018   1020   1049   1148
                        1308   1612   2333   2347   2349   2355   2362   2399   2406   2418   2421   3339   3352*
SWREG   000176           182*   511
SW0    = 000001          107*
SW00   = 000001           97*   107
SW01   = 000002           96*   106
SW02   = 000004           95*   105
SW03   = 000010           94*   104
SW04   = 000020           93*   103
SW05   = 000040           92*   102
SW06   = 000100           91*   101
SW07   = 000200           90*   100
SW08   = 000400           89*    99
SW09   = 001000           88*    98
SW1    = 000002          106*
SW10   = 002000           87*
SW11   = 004000           86*
SW12   = 010000           85*
SW13   = 020000           84*
SW14   = 040000           83*
SW15   = 100000           82*
SW2    = 000004          105*
SW3    = 000010          104*
```

```
SW4    = 000020            103#
SW5    = 000040            102#
SW6    = 000100            101#
SW7    = 000200            100#
SW8    = 000400             99#
SW9    = 001000             98#
TADDR    001364            377#    454    1012*  1067*  1077*  1095*  1145*  1155*
TBITVE= 000014            140#
TEMP     001346            370#    879     894    900    901    902*   952*   955*   956*   959*   961    977*   993*
                          1150*   1162*  1197   1248   1407*  1412*  1714*  1746*  1751*  1752*  1753*  1754*  1760
TENS     014573           1814*   1823*  1824   1826*  1830*  2189#
TEST     003410            696     718    740    754#   799
TESTAD   002552            598#   1069   1080   1098
TESTIT   003400            639     651    659    667    674    751#
TESTR2   011544           1816#   1821   1825   1828
TIME     020532           3096*   3099   3104*  3108#  3110*
TKVEC  = 000060            147#
TOMSG    012433           1159    1965#
TPVEC  = 000064            148#
TRAPVE= 000034            146#    489*   490*
TRTVEC= 000014            141#
TRY      006544           1318#   1348
TRYAG    002426            573#    597
TST1     002710            636#    637
TST10    003252            723#
TST11    003436            766#
TST12    003510            784#
TST13    003636            787     812#
TST14    003666            823#
TST15    003716            835#
TST16    003746            848#
TST17    003776            769     770    860#
TST2     002754            648#
TST20    004026            871#
TST21    004274            923#
TST22    004324            934#
TST23    004354            945#
TST24    004524            970     975#
TST25    004564            988#
TST26    004622           1000#
TST3     003000            657#
TST4     003016            665#
TST5     003034            672#
TST6     003052            679#
TST7     003166            705#
TST8     010230           1580    1582#
TYPBAD   007372           1426    1436   1440#
TYPE   = 104401            445     451    459    465    470    543    557    572    596    617    960    963    969
                           971    1016   1022   1053   1064   1082   1087   1146   1147   1156   1159   1175   1181
                          1183    1189   1204   1206   1292   1353   1443   1452   1454   1460   1463   1465   1468
                          1472    1475   1478   1480   1487   1490   1492   1496   1497   1506   1513   1536   1542
                          1551    1553   1589   1590   1593   1596   1615   1631   1640   1643   1645   1655   1657
                          1808    1832   1905   2258   2261   2265   2401   2409   2441   2458   2460   2463   2465
                          2469    2476   2523   2651   2705   2709   2713   2941   3127   3134   3151   3270   3318#
```

```
                    3361
TYPEDG  006406      1182    1283#   1644
TYPOC = 104402      2449    2473    3146    3319#
TYPON = 104404      3321#
TYPOS = 104403       462     614    1025    1056    1084    1178    1186    1199    1286    1295    1449    1539    1546
                    1648    3320#
TYPOUT  011620      1817    1829#
TYPRP   010436      1623    1631#   1693
TYPSET  005716      1172    1174#   1239
UNEXP   001526       433#    649
VADR    001326       362#    609    1108    1109
VARLT1  011754      1842    1865#
VARLT2  011764      1844    1870#
VARLT3  011774      1838    1875#
VECTOR  001456       424#    649*   1110*   1119*   1120*   1123    1382*   1723*
VECTPS  001460       425#
VECTR1  001324       361#   1111*   1112*   1123*   1124*   1125*
VERSN   001462       427#
VLIN    011750      1549    1862#
VNP     011744      1653    1860#
VNR     011742      1651    1835    1859#
VSET    011746      1202    1861#
VTFLG   002656       544     547     550     622#
VTINIT  014021      1615    2117#
VTSS    002326       552     555#
VVCT    001330       363#   1110    1111
V1      011720       818    1849#
V10     011724       829    1851#
V115    011732       882    1854#
V144    011730       929    1853#
V2      011722       918    1850#
V240    011734       854     866     940    1855#
V5      011736       906    1856#
V50     011726       775     841    1852#
V50D    011740       966    1857#
WFADJ   011650       566    1835#
WFTEST  001416       390#    473#    475*   1840
WIDE    001336       366#   1364*   1439*   1469    1482
WIDMSG  012603      1472    1985#
WRAP    003436       762#   1088    1099
W1      017720      2839*   2854#
W2      017722      2843*   2855#
W3      017724      2848*   2856#
$AERR   017450       605    2658*   2751*   2777#   3067
$APTHD  001000       207     213#
$ASTAT= ****** U    2593    2608
$ATYC   016350      2564    2566#
$ATY1   016324      2562#
$ATY3   016332      2508    2563#
$ATY4   016342      2414    2565#
$AUTOB  001134       244#
$BASE   001250       308#    598    1116    1117
$BDADR  001122       239#
$BDDAT  001126       241#    598*    605     609*    757    1735*   1760*   1761    2193    2195    2198
```

```
$BELL   001164          259#  2401  2433
$CDWI   001254          310#
$CHARC  016320         2525*  2535*  2542  2551*  2556#
$CKSWR= ****** U        3324
$CMTAG  001100          257#   478   479   487   493   494   495
$CM3    = 000000        257#
$CNT    020420         3045*  3047*  3053#
$CNTLG  015103         2275#
$CNTLU  015076         2274#
$CPUOP  001222          282#
$CRLF   001171          261#  1064  1087  2274  2409  2433  2441  2460  2465  2469  2524  2559
$DAtR   020072         2891*  2908*  2912#  3023
$DEVCT  001204          273#
$DEVm   001252          309#
$DOAGN  012104         1901   1907  1913#
$ENDAD  012074          193   1909#  2428
$ENDCT  012054          493   1903#
$ENDMG  012113         1905   1917#
$ENuLL  012110         1916#
$ENV    001214          278#  2411  2503  2571  2595
$ENVM   001215          279#   516  2505  2510  2573
$EOP    012020         1074   1092  1104  1893#
$EOPCT  012046          493*  1900#  1904
$ERFLG  001103          230#   435*  2323  2351  2353  2359*  2381  2396*  2433
$ERMAX  001115          236#   496*  2353  2376*  2381
$ERrOR  015512          487   2395#
$ERRPC  001116          237#  2193  2195  2198  2403*  2404*  2405  2433  2447
$ERrTB  001256          327#  2455
$ERRTY  015706         2408   2440#
$ERTTL  001112          234#  2402*  2433
$ESCAP  001162          258#   434*   437*   495*  2375*  2424  2426  2433
$ETABL  001214          277#
$ETEND  001256          219    311#
$FATAL  001176          270#  2599*
$FFLG   016570         2562*  2565*  2593  2602*  2610#
$FILLC  001156          255#  2528  2559
$FILLS  001155          254#  2559
$GDADR  001120          238#
$GDDAT  001124          240#   605   638*  641*  642  650*  658*  666*  673*  687*  709*  733*  754
                        757    792*  1734*  1758*  1762  2193  2195
$GET42  012064         1906#
$GTSWR= ****** U        3323
$HD     = 000000         40
$HIBTS  001000          214#
$HIOCT  015232         2312*  2317#
$ICNT   001104          231#  2366*  2367  2369*  2380
$ILLUP  021516         3331   3347  3366#
$INLP   020266          684    693   700   713   717  731  745  757  803  1038  1044  1048  1262
                       1268   1273  1717  1728  1742  1746  3004#  3067  3141  3193  3200
$INTAG  001135          245#
$ITEMB  001114          235#  2405*  2413  2433  2444
$LF     001172          262#  2265  2274  2433  2559
$LFLG   016567         2603*  2609#
$LOAD   017452         2659   2787#
```

```
$LPADR  001106          232#   497*   636*   770*  2357*  2373*  2378   2380
$LPAI   016572         2633#  2985   3030
$LPERR  001110          233#   498*   637*   769*  2357   2374*  2380   2423
$LPW    020074         2838   2842   2847   2851   2870   2873   2890   2907   2923#
$MADR1  001226          295#
$MADR2  001232          299#
$MADR3  001236          302#
$MADR4  001242          305#
$MAIL   001174          215    219    268#   515   2372   2411   2503
$MAMS1  001224          289#
$MAMS2  001230          297#
$MAMS3  001234          300#
$MAMS4  001240          303#
$MBADR  001002          215#
$MFLG   016566         2563*  2569   2604*  2608#
$MNEW   015121         2278#
$MSGAD  001210          275#  2579*  2582
$MSGLG  001212          276#  2584*
$MSGTY  001174          269#  2577   2585*  2597   2601*
$MSWR   015110         2276#
$MTYP1  001225          290#
$MTYP2  001231          298#
$MTYP3  001235          301#
$MTYP4  001241          304#
$MXCNT  015510         2370   2380#
$NULL   001154          253#  2530   2559
$NWTST= 000001          633#   645#   654#   662#   669#   676#   702#   720#   763#   781#   809#   820#   832#
                        845#   857#   868#   920#   931#   942#   972#   985#   997#
$OCNT   021272         3242*  3271*  3284#
$OFS  = 020330         3020#  3196   3197   3203
$OMODE  021274         3237*  3241*  3246   3249*  3260*  3286#
$OUTLP  020170          605    687    696    709    728    737    754    791    796    807    876    889   1016
                       1033   1041   1195   1248   1253   1265   1322   1379   1720   1731   2962#  3158
$OVER   015474         2334   2350   2358   2368   2377#
$PASS   001202          272#   515*  1002   1079*  1097*  1897*  1898*  1916   2364   2381
$PASTM  001006          217#
$POWER  021524         3362   3369#
$PUTS   020740         1316   1374   3192#
$PWRAD  021512         3364#
$PWRDN  021352          491   3331#  3359
$PWRMG  021506         3362#
$PWRUP  021424         3341   3347#
$QUES   001170          260#  2258   2274   2433   2559
$RDCHR  014640         2222#  3324
$RDDEC= ****** U       3327
$RDLIN  014760         2250#  3325
$RDOCT  015132         2290#  3326
$RDSZ = 000010         2243#
$RESET  020422          911   1771   3064#  3074
$RTNAD  012106         1915#
$R2A  = ****** U       3327
$SAD    020416         3042   3044*  3052#
$SAVRE= ****** U       3327
$SAVR6  021522         3340*  3348   3349*  3350*  3368#
```

```
$SCOPE  015234          485    2332#
$SETUP= 000037          328#    484     485     487     489     491     493     494     495     497    1895    2211    2280
                       2333    2396    2421    2428
$STUP = 177777          328#
$SVLAD  015440         2342    2371#
$SVPC = 000214          191#    196
$SWR  = 167400           30#     40     154     155     156     157     158     159     160     161     257     258     259
                        494     495     497     498     637     649     658     666     673     680     706     724     767
                        785     813     824     836     849     861     872     924     935     946     976     989    1001
                       1888    1896    1908    1914    1916    2324    2325    2326    2327    2328    2333    2345    2347
                       2348    2351    2352    2353    2360    2361    2362    2374    2377    2380    2387    2388    2389
                       2390    2391    2399    2406    2418    2421    2433    3365
$SWREG  001216          280#    518
$SWRMK= 000000          161     162    2328    2329    2349
$TBF4   007150         1391#
$TEMP1  001432          396#   1328#   1329#   1331    1379#   1380#   1391
$TEMP2  001434          397#   1326#   1327#   1335
$TESTN  001200          271#   2372#
$TIMES  001160          257#    494#    724#    767#    785#    813#    824#    836#    849#    861#    872#    924#    935#
                        946#    976#    989#   1001#   1896#   2360#   2367    2370#   2380
$TKB    001146          250#    441     629    2209    2226    2232    3133
$TKs    001144          249#    542#    574#    623     912#   1772#   1880#   2209    2224    2230    3131
$TLKR   017726         2867#   3019
$TLKW   017612         2835#   2977
$TN   = 000027           30#     40     633    637#     645    649#     654    658#     662    666#     669    673#     676
                        680#    702    706#     720    724#     763    767#     768     781    785#     787     809    813#
                        820     824#    832    836#     845    849#     857    861#     868    872#     920    924#     931
                        935#    942     946#    970     972     976#     985    989#     997   1001#
$TOUT   020362         2681    2735    2762    2876    2893   3042#
$TPB    001152          252#   1790*   1800*   2548*   2559
$TPFLG  001157          256#   2497    2559
$TPS    001150          251#   1788    1798    2546    2559
$TRAP   021276          489    3295#
$TRAP2  021320         3306#   3317
$TRP  = 000010         3310#   3319#   3320#   3321#   3322#   3324    3325#   3326#   3327#
$TRPAD  021332         3300    3317#
$TSTM   001004          216#
$TSTNM  001102          229#    768*   1895*   2323    2349    2371*   2372    2377    2381    2398    2433
$TTYIN  015066         2251    2252    2269    2273#
$TYPBN= ******  U      3322
$TYPDS= ******  U      3322
$TYPE   016042         2497#   2590    3310    3318
$TYPEC  016254         2527    2534    2541    2546#   2547
$TYPEX  016322         2552    2554    2557#
$TYPOC  021074         3240#   3319
$TYPON  021110         3239    3242#   3321
$TYpOS  021050         3235#   3320
$T6MP   006634         1331#
$UNIT   001206          274#
$UNITM  001010          218#
$USWR   001220          281#
$UTK    020544         3124#
$VECT1  001244          306#   1119    1121
$VECT2  001246          307#
```

```
$XTSTR  015244            2336#
$$GET4= 000000            1908#
$OFILL  021273            3236*   3240*   3250    3285#
$40CAT= ****** U          2333    2408
.       = 042354           176#    180#    191     192#    194#    196#    197#    203     204#    206#    208#    226#    263
                           430#    482     497     498    1916    1919    2209   2273#   2274    2280    2380    2381    2433
                          2479#   2559    2611#   3020    3130#   3343    3367   3373#   3374#
.DVLS   001464             429#    535*    599*    600*   2965    2986    3007    3031   3125*
.$ASTA= ****** U          2563    2566
.$X     = 001000           203#    208
```

```
ADDM      171#  1270
BICM      169#   690
CLRM      168#   804  1013
CMPM      172#
COMMEN    150#
DUMWRN    331#
ENDCOM    150#
ERROR      44#   436   640   652   660   668   675   697   719   741   776   800   819   830   842
           855   867   883   907   919   930   941  1737  2677  2683  2696  2737  2764  2878  2895
ESCAPE    150#   433
GETPRI    150#
GETSWR    150#
INCRM     167#   681  1034  1258
MOVEI      20#   682   691   698   711   715   729   743   754   801  1036  1042  1046  1260  1266
          1271  1715  1725  1740  1744  3065
MOVEM      19#   603   685   694   707   726   735   751   789   794   805   874   887  1014  1031
          1039  1193  1245  1251  1263  1320  1377  1718  1729
MOVEMR    173#  1030  1250  1376  1717
MOVERO    166#   698   715   743   801  1046
MULT      150#
NEWTST    150#   633   645   654   662   669   676   702   720   763   781   809   820   832   845
           857   868   920   931   942   972   985   997
POP       150#  2313  2605  2606  3352  3353
PUSH      150#  2292  2566  2568  2589  3333  3339
REPORT    150#
SCOPE      45#   648   657   665   672   679   705   723   746   784   812   823   835   848   860
           871   923   934   945   975   988  1000
SETPRI    150#
SETTRA   3310#  3319  3320  3321  3324  3325  3326
SETUP     150#   477
SKIP      150#   579   582   585   588   591   594   610   624   626   628   643   758   787   970
          1021  1613  1837  1839
SLASH     150#
SPACE     150#
STARS     150#   189   200   202   209   222   263   266   633   635   645   647   654   656   662
           664   669   671   676   678   702   704   720   722   763   765   781   783   809   811
           820   822   832   834   845   847   857   859   868   870   920   922   931   933   942
           944   972   974   985   987   997   999  1885  2208  2214  2243  2282  2320  2383  2435
          2482  2561  3212  3289  3329  3345
SWRSU     150#   499#
TOUT      432#  2680  2734  2761  2875  2892
TRMTRP   3310#
TSTBM     170#   710   728  1042  1266  1740
TYPBIN    150#
TYPDEC    150#
TYPNAM    150#
TYPNUM    150#
TYPOCS    150#   460   612  1023  1054  1176  1184  1197  1284  1293  1447  1537  1544  1646
TYPOCT    150#  2447  2471  3145
TYPTXT    150#  2651  2705  2709  2713  2940  3126  3151
$CAL.      30#  2616
$DMAST     30#
$DMDT      30#
$MMAST     30#
```

```
$$CMRE     220#
$$CMTM     220#
$$ESCA     150#
$$NEWT     150#     633     645     654     662     669     676     702     720     763     781     809     820     832     845
           857      868     920     931     942     972     985     997
$$SET     3310#    3319    3320    3321    3324    3325    3326
$$SETM     515#
$$SKIP     150#     787     970
.EQUAT      30#      40
.HEADE      30#
.KMADR      17#     398
.KSIS       22#     522
.LOADL      25#    2779
.LPAIN      22#    2617
.PUTCS      24#    3172
.RESET      22#    3054
.SETUP      30#     328
.SWRHI      30#     150
.SWRLO     162#
.UTK        29#    3114
.$ACT1      30#     187
.$APTB      30#     264#
.$APTH      30#     198
.$APTY      30#    2559
.$CATC      30#     174
.$CMTA      30#     220
.$EOP       30#    1883
.$ERRO      30#    2381
.$ERRT      30#    2433
.$INLP      28#    2989
.$MMAC      18#
.$OUTL      27#    2950
.$PARM      30#
.$POWE      30#    3327
.$RAND      30#
.$RDOC      30#    2280
.$READ      30#    2206
.$SAVE      30#
.$SCOP      30#    2318
.$SPAC      30#
.$SWDO      30#
.$TLKW      26#    2828
.$TOUT     432#    3033
.$TRAP      30#    3287
.$TYPD      30#
.$TYPE      30#    2480
.$TYPO      30#    3210
```

```
          3339  3340  3341  3347  3348  3352  3353  3354  3355  3356  3357  3358  3359  3360  3363
MOVB       496  1121  1248  1326  1328  1332  1336  1392  1457  1486  1790  1800  2226  2232  2255
          2260  2299  2372  2376  2405  2413  2512  2540  2548  2562  2563  2565  2718  2723  2725
          2727  2732  2760  2841  2844  2846  2848  2849  2850  2871  2872  2889  2891  2906  2908
          3133  3196  3197  3203  3236  3237  3240  3241  3242  3246  3249  3250  3269  3298
NEG       1169  1236  1521  1765  1809  3247
NOP        766  1894  1910  1911  1912
RESET     1908  3064
ROL        641  2302  2304  2306  3253  3255  3256  3257  3259
RTI        438   469   510  2242  2270  2316  2379  2432  2517  3111  3281  3308  3365
RTS        620   631   747   760  1005  1010  1138  1205  1207  1220  1240  1280  1298  1349  1598
          1605  1616  1624  1656  1658  1674  1694  1708  1755  1769  1774  1801  1833  1848  2470
          2557  2607  2755  2775  2813  2853  2910  2938  2982  3026  3050  3076  3103  3106  3207
          3301
SUB        902   956   959  1191  1219  1233  1346  1445  1516  1526  1667  1673  1687  1763  2404
          2582
SWAB      1029  1249  1373  1713
TRAP      3310  3319  3320  3321  3324  3325  3326
TST        539   569   605   627   786   892  1002  1106  1135  1234  1433  1440  1461  1476  1493
          1554  1601  1632  1635  1768  1779  1806  1816  1836  1840  1846  2310  2340  2364  2418
          2424  2474  2514  2522  2544  2577  2595  2597  2816  2967  2975  3009  3018  3067  3094
          3099  3264  3297
TSTB       623   713   731  1044  1268  1742  1788  1798  2224  2230  2351  2497  2546  2569  2580
          2593  2748  3131
.ASCII     260   261  2008  2018  2031  2039  2045  2082  2121  2129  2132  2135  2138  2141
.ASCIZ     259   262  1917  1921  1924  1943  1945  1946  1949  1950  1951  1953  1956  1957  1959
          1962  1964  1965  1971  1973  1976  1979  1985  1991  1996  2003  2005  2012  2015  2022
          2026  2057  2068  2075  2079  2085  2091  2092  2093  2094  2095  2097  2098  2099  2144
          2148  2152  2157  2162  2167  2173  2182  2274  2275  2276  2278  2478  2654  2708  2712
          2716  2944  3130  3154  3165  3369
.ASECT      14
.BLKB     2273
.BLKW      430  3373  3374
.BYTE      229   230   235   236   244   245   253   254   255   256   278   279   289   290   297
           298   300   301   303   304   463   464   615   616  1026  1027  1057  1058  1085  1086
          1179  1180  1187  1188  1200  1201  1287  1288  1296  1297  1450  1451  1540  1541  1547
          1548  1649  1650  1916  1933  1934  1935  1937  1939  2103  2111  2117  2187  2188  2189
          2190  2271  2272  2415  2416  2608  2609  2610  3282  3283  3284  3285
.DSABL    2211
.ENABL      30  2209
.END      3376
.ENDC       35    44   136   150   158   160   161   162   185   190   194   196   201   203   210
           223   227   229   257   258   259   260   264   267   289   297   300   303   306   307
           308   309   310   313   328   427   435   464   465   483   484   487   489   491   493
           494   495   497   499   520   538   580   583   586   589   592   595   611   616   617
           625   627   629   634   635   636   637   644   646   647   648   649   655   656   657
           658   663   664   665   666   670   671   672   673   677   678   679   680   703   704
           705   706   721   722   723   724   725   759   764   765   766   767   768   782   783
           784   785   786   788   810   811   812   813   814   821   822   823   824   825   833
           834   835   836   837   846   847   848   849   850   858   859   860   861   862   869
           870   871   872   873   921   922   923   924   925   932   933   934   935   936   943
           944   945   946   947   971   973   974   975   976   977   986   987   988   989   990
           998   999  1000  1001  1002  1022  1027  1028  1058  1059  1180  1181  1188  1189  1201
          1202  1288  1289  1297  1298  1451  1452  1541  1542  1548  1549  1614  1650  1651  1838
```

```
                857   868   920   931   942   972   985   997  1894  2292  2313  2567  2568  2589  2605
               2606  3333  3339  3352  3353
.LIST            14    30   150   161   180   257   264   267   328   499   605   633   637   645   649
              654   658   662   666   669   673   676   680   684   687   693   696   700   702   706
              709   713   717   720   724   728   731   737   745   754   757   763   767   781   785
              791   796   803   807   809   813   820   824   832   836   845   849   857   861   868
              872   876   889   920   924   931   935   942   946   972   976   985   989   997  1001
             1016  1033  1038  1041  1044  1048  1195  1248  1253  1262  1265  1268  1273  1322  1379
             1717  1720  1728  1731  1742  1746  1895  1908  2243  2328  2428  2654  2708  2712  2716
             2944  3067  3130  3154  3310  3318  3319  3320  3321  3322  3324  3325  3326  3327
.MACRO         17    18    19    20    22    24    25    26    27    28    29    30   162   166   167
              168   169   170   171   172   173   220   331   432   515  3310
.MCALL          30   150   264   499
.MEXIT         312
.NLIST          14    30   150   161   180   257   264   267   328   499   605   633   637   645   649
              654   658   662   666   669   673   676   680   684   687   693   696   700   702   706
              709   713   717   720   724   728   731   737   745   754   757   763   767   781   785
              791   796   803   807   809   813   820   824   832   836   845   849   857   861   868
              872   876   889   920   924   931   935   942   946   972   976   985   989   997  1001
             1016  1033  1038  1041  1044  1048  1195  1248  1253  1262  1265  1268  1273  1322  1379
             1717  1720  1728  1731  1742  1746  1895  1908  2243  2328  2428  2654  2708  2712  2716
             2944  3067  3130  3154  3310  3318  3319  3320  3321  3322  3324  3325  3326  3327
.PAGE          220   313
.PSECT          14
.REM             1    14
.REPT          180
.SBTTL          40   150   174   183   187   198   220   264   313   357   439   472   477   541   556
              633   645   654   662   669   676   702   720   761   763   781   809   820   832   845
              857   868   920   931   942   972   985   997  1011  1066  1076  1094  1105  1144  1154
             1883  1920  2206  2280  2318  2381  2433  2480  2559  3210  3287  3310  3327
.TITLE          30
.WORD          180   181   182   195   214   215   216   217   218   219   228   231   232   233   234
              237   238   239   240   241   242   243   246   247   248   269   270   271   272   273
              274   275   276   280   281   282   295   299   302   305   306   307   308   309   310
              407   410   412   414   416   418   420   422   424   425   427   429   605   684   687
              693   696   700   709   713   717   728   731   737   745   754   757   791   796   803
              807   876   889  1016  1033  1038  1041  1044  1048  1195  1248  1253  1262  1265  1268
             1273  1317  1322  1351  1375  1379  1717  1720  1728  1731  1742  1746  1900  1903  1915
             2317  2459  2464  2509  2556  2591  2660  2757  2777  2912  2978  2979  2986  3021  3031
             3052  3053  3067  3077  3078  3108  3112  3142  3143  3148  3159  3160  3163  3164  3194
             3195  3201  3202  3208  3286  3317  3362  3364

        000000

ERRORS DETECTED:  0
```

```
*DRLPK,DRLPK/SOL/CRF=DRLPA.MAC,DRLPK
RUN-TIME:  27 15 2 SECONDS
CORE USED:  41K
```