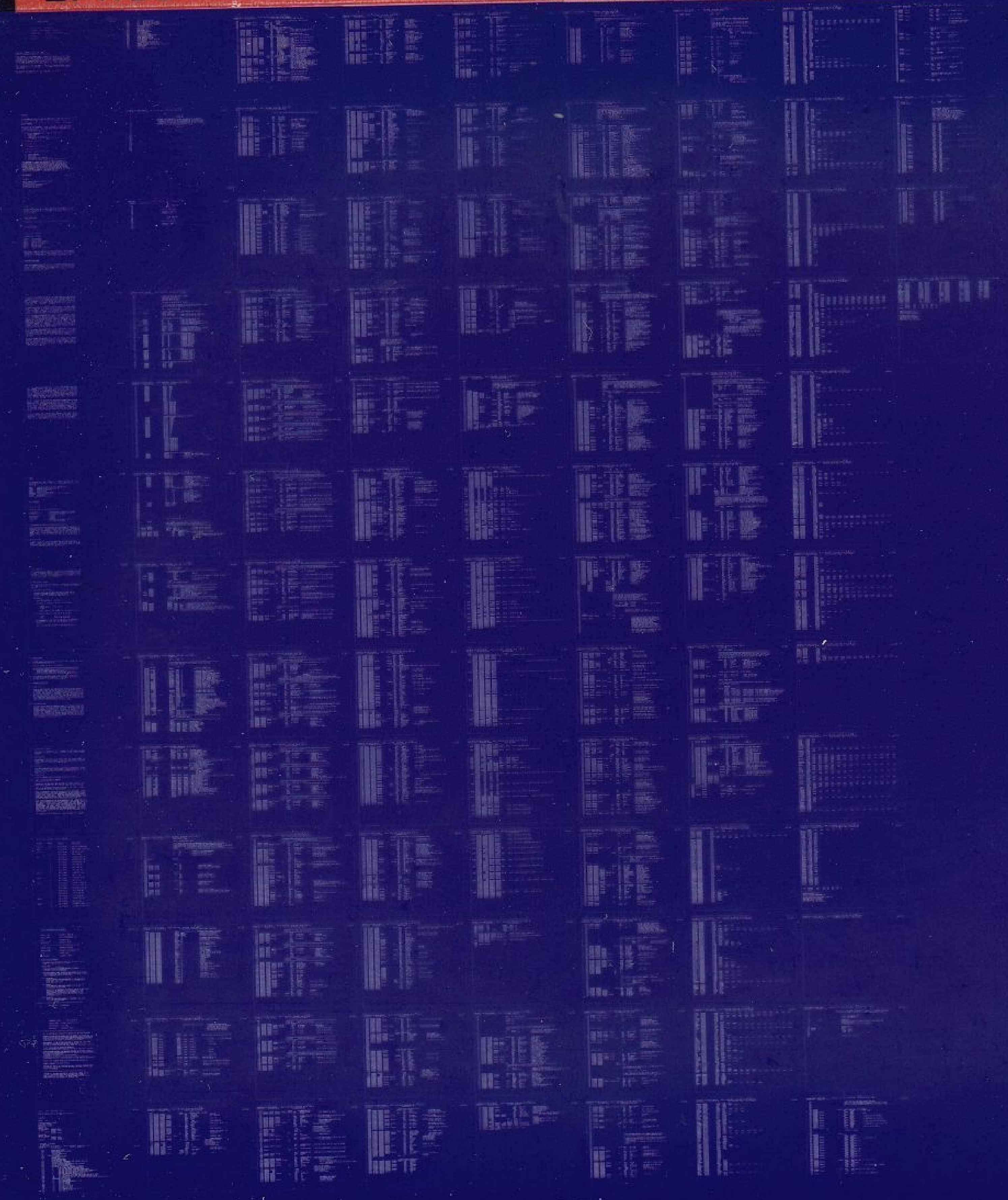
LPA11

LPA/AD11-K TEST
CRLPKCO

AH-B050C-MC
FICHE 1 OF 1

FEB 1981
COPYRIGHT© 76-80
MADE IN USA



IDENTIFICATION

Product Code:	AC-B049C-MC
Diagnostic code:	MAINDEC-11-CRLPK-C
Product Name:	CPLPKCO LPA/AD11-K Test
Revised:	DEC. 1980
Maintainer:	Diagnostic Group

Copyright (C) 1976, 1977, 1979, 1980
Digital Equipment corporation, Maynard, Mass.

This software is furnished under a license for use only on a single computer system and may be copied only with the inclusion of the above copyright notice. This software, or any other copies thereof, may not be provided or otherwise made available to any other person except for use on such system and to one who agrees to these license terms. Title and ownership of the software shall at all times remain in dec.

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

DEC assumes no responsibility for the use or reliability of its software in equipment which is not supplied by DEC.

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-CRLPA" BEFORE
RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 10.

2.0 REQUIREMENTS

2.1 Equipment

PDP-11 family computer with 8K of memory
Console terminal
AD11K Module installed in an LPA-11
Bit-map terminal <OPTIONAL>
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 Bit map 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. Starting address of the USER LINK loop is at 214.

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) (1326) and VVCT (vector address) (1330). 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 ^G 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 BIT-MAP Graphic Output

The screen display may be halted for examination by setting 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 214

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
************************************************************
```

8.2 The program is chainable under XXDP. However, you may only execute it one pass (R RLPKC0/1). The reason is because of the ''BUS RESET'' in the end of pass handler.

9.0 PROGRAM DESCRIPTION

9.1 Logic Tests

These 8 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-CRLPA. WHEN THE USER RUNS "CRLPA" HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY "LOOK" ALIKE AND "CRLPA" 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 "CRLPA" 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-CRLPA	LPA11-K SYSTEM EXER.
M8254	"B"	MD-11-CRLPN	M8254 (IPBM) FIELD DIAG.
AA11-K	A	MD-11-CRLPB	LPA/AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11	A	MD-11-CRLPC	LPA/AR11 DIAG. #1
	A	MD-11-CRLPD	LPA/AR11 DIAG. #2
	A	MD-11-CRLPE	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-CRLPF	LPA/DR11-K DIAG.
	B	MD-11-DZDRG	DR11-K DIAG.
KW11-K	A	MD-11-CRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS11	A	MD-11-CRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-CRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-CRLPJ	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-CRLPK	LPA/AD11-K DIAG.
	B	MD-11-DZADL	AD11-K DIAG.
M8200-YC	B	MD-11-CRLPL	LPA/DMC-11 DIAG. TST I
	B	MD-11-CRLPM	LPA/DMC-11 DIAG. TST II

```
THIS IS A HISTORY FILE OF CRLPK-C
---------------------------------

PRODUCT CODE:           MAINDEC-11-DZADL-B

PRODUCT NAME:           AD11-K PERFORMANCE TEST

DATE:                   DECEMBER 1976

MAINTAINER:             DIANOSTIC GROUP

****************************************************

PRODUCT CODF:           MAINDEC-11-DRLPK-A

PRODUCT NAME:           LPA/AD11-K PERFORMANCE TEST

DATE:                   JANUARY 1978

MAINTAINER:             DIAGNOSTIC GROUP


REASON FOR DEVELOPMENT:

1) TO ENABLE THE OPERATOR TO CHECK OUT THE AD11-K OPTION
   WHEN IT IS ON THE LPA11-KX I/O BUS.

CHANGES MADE:

1) TOOK OUT CERTAIN TESTS FROM ORIGINAL DIAGNOSTIC (I.E.
   INTERRUPTS,TIME DEPENDENT CODE).

2) REPLACED DIRECT LINKS TO DEVICE WITH MACRO CALLS TO THE
   KMC-11 MICRO CODE. KMC-11 MICRO CODE (FILE:DRLPX2) HANDLES
   DIRECT COMMUNICATIONS WITH THE DEVICE.

FILE: DRLPA.MAC
      CONTAINS MACRO LINKS BETWEEN PDP-11 CODE AND KMC-11
      MICRO CODE. FILE: DRLPX2 NEEDS TO BE ASSEMBLED WITH
      DRLPK (SEE .CTL FILE).

FILE: DRLPX2
      MICRO CODE FILE THAT GETS LOADED INTO THE KMC-11
      VIA ROUTINES IN DRLPA.MAC.

      DRLPX2.P11 IS ASSEMBLED WITH MACY11 (ONLY) AS ANY OTHER
      .P11 FILE. THE RESULTS OF ITS ASSEMBLY IS A .OBJ
      MODULE AS WAS THE RESULT OF THE ASSEMBLY OF THE
      DIAGNOSTIC .P11 FILE. BOTH .OBJ FILES GET LINKED
      WITH LNKX11 (ONLY).

FILE: DRLPK.CTL
      THIS FILE EXPLAINS SEQUENCE OF ASSEMBLES AND LINKS.
      IT IS IN TOPS-20 FORMAT.
************************************************************

           PRODUCT CODE:   AC-B049B-MC
```

```
DIAGNOSTIC CODE:       MD-11-CRLPK-B

PRODUCE NAME:   CRLPKB  LPA/AD11-K TEST

DATE REVISED:   JULY 1979

MAINTAINER:     DIAGNOSTIC GROUP

**************************************************************

        THE "B" VERSION WAS GENERATED TO REPAIR THE FOLLOWING PROBLEMS:

1.      PROGRAM LISTING DID NOT AGREE WITH THE BINARY FILE AFTER LOC.
        12064. THIS WAS DUE TO THE RELEASE ENGINEERING GROUP REASSEMBLING
        TO GET THE LISTING AND USING THE BINARY FILE SUPPLIED BY AUTHOR.
        (DEVELOPED WITH C2 SYSMAC - RELEASED WITH C3 SYSMAC)

2.      WHEN SUBTEST "A" OR "W" WAS SELECTED, A "MICRO-CODE LOAD ERROR"
        OCCURRED AT LOCATION 17612 ON THE "THIRD PASS".
        (DUE TO THE AUTHOR FORGETTING ABOUT WHERE THE MICRO-CODE "HIDES" AT.

3.      "TST11" COULD NOT BE LOOPED ON CORRECTLY.
        (ORIGIONAL PROGRAM USED A ABSOLUTE TAG FOR AT THAT TEST <<TST17>>)

4.      AFTER A POWER FAILURE, THE PROGRAM APPEARED TO RECOVERY PROPERLY.
        BUT AFTER THE OPERATOR ENTERED THE TEST NUMBER THE PROGRAM REPORTED
        "LPA FAULT" AND THEN HALTS.
        (PROGRAM DID A RESTART - IT MUST BE STARTED)
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
1.      REASSELBLED THE FILE - <EASY AND FREE FIX WHEN WORKING ON PROBLEM 2-4

2.      PROTECT THE "HIDDEN" SPACE THAT THE MICRO-CODE RESIDES AT.

3.      REMOVE INCORRECT TAG FROM "TST11"

4.      BEACUSE THE KMC-11 IS A VOLIATLE DEVICE A COMPLETE PROGRAM START
        WAS NEEDED. JUST A ONE LOCATION PATCH IN THE POWER FAIL ROUTINE
        FIXES  THE PROBLEM.

        *******************************************************

        VERSION "C" WAS MADE BECAUSE OF A MICRO-CODE CHANGE IN THE
        LPA-11.  EFFECTIVELY ONLY THE TITLE AND TITLE TYPEOUT
        WERE CHANGED IN THE SOURCE. THE MAJOR CHANGE OCCURRED IN THE
        MACRO FILE THAT THE PROGRAM IS ASSEMBLED WITH.
```

```
LNKX11   V023   24-OCT-80   9:51                                                SEQ 0013
#CRLPKC.BIN/B:42000,CRLPKC.MAP=CRLPKC,CRLPX2/E


LOAD MAP

IDENT:  4.01

TRANSFER ADDRESS: 000001
LOW LIMIT: 042000
HIGH LIMIT: 046000
**********
MODULE  LPA
SECTION ENTRY   ADDRESS SIZE
<. ABS.>        000000  000000
        DRLPX2  042000
<      >        042000  000000
**********
MODULE  DRLPX2
SECTION ENTRY   ADDRESS SIZE
<      >        042000  000000
<ABCODE>        042000  004000


 RUN-TIME:  0 SECONDS
 2K CORE USED
LPA-AD11K TEST MD-11-CRLPKC     MACY11 30G(1063)  24-OCT-80  09:48
CRLPKC.P11    14-AUG-80 13:59           TABLE OF CONTENTS                       SEQ 0014
```

```
                    .REM    [

                                        CRLPAB.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 TESTING THE LPA SYSTEM.

                                        GOOD LUCK !

                    [
                    .GLOBL  DRLPX2
```

```
                .TITLE  MMAST.MAC
                .IDENT  /4.01/
        ;
        ; LPA11-K MICRO CODE
        ;
        ; CHARLES A. SAMUELSON
        ; NOVEMBER, 1977
        ;

                .TITLE  DMAST.MAC
                .IDENT  /4.01/
        ;
        ; LPA11-K MICRO CODE
        ;
        ; CHARLES A. SAMUELSON
        ; NOVEMBER, 1977
        ;

```

```
  1162                              .TITLE  LPA-AD11K TEST MD-11-CRLPKC
   (1)                              ;*COPYRIGHT (C) 1980
   (1)                              ;*DIGITAL EQUIPMENT CORP.
   (1)                              ;*MAYNARD, MASS. 01754
   (1)                              ;*
   (1)                              ;*PROGRAM BY MODIFIED BY R. SHOOP
   (1)                              ;*
   (1)                              ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
   (1)                              ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
   (1)                              ;*
  1163                              .SBTTL  BASIC DEFINITIONS
   (1)
   (1)                              ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
   (1)          001100              STACK=  1100
   (1)                              .EQUIV  EMT,ERROR        ;;BASIC DEFINITION OF ERROR CALL
   (1)                              .EQUIV  IOT,SCOPE        ;;BASIC DEFINITION OF SCOPE CALL
   (1)
   (1)                              ;*MISCELLANEOUS DEFINITIONS
   (1)          000011              HT=     11               ;;CODE FOR HORIZONTAL TAB
   (1)          000012              LF=     12               ;;CODE FOR LINE FEED
   (1)          000015              CR=     15               ;;CODE FOR CARRIAGE RETURN
   (1)          000200              CRLF=   200              ;;CODE FOR CARRIAGE RETURN-LINE FEED
   (1)          177776              PS=     177776           ;;PROCESSOR STATUS WORD
   (1)                              .EQUIV  PS,PSW
   (1)          177774              STKLMT= 177774           ;;STACK LIMIT REGISTER
   (1)          177772              PIRQ=   177772           ;;PROGRAM INTERRUPT REQUEST REGISTER
   (1)          177570              DSWR=   177570           ;;HARDWARE SWITCH REGISTER
   (1)          177570              DDISP=  177570           ;;HARDWARE DISPLAY REGISTER
   (1)
   (1)                              ;*GENERAL PURPOSE REGISTER DEFINITIONS
   (1)          000000              R0=     %0               ;;GENERAL REGISTER
   (1)          000001              R1=     %1               ;;GENERAL REGISTER
   (1)          000002              R2=     %2               ;;GENERAL REGISTER
   (1)          000003              R3=     %3               ;;GENERAL REGISTER
   (1)          000004              R4=     %4               ;;GENERAL REGISTER
   (1)          000005              R5=     %5               ;;GENERAL REGISTER
   (1)          000006              R6=     %6               ;;GENERAL REGISTER
   (1)          000007              R7=     %7               ;;GENERAL REGISTER
   (1)          000006              SP=     %6               ;;STACK POINTER
   (1)          000007              PC=     %7               ;;PROGRAM COUNTER
   (1)
   (1)                              ;*PRIORITY LEVEL DEFINITIONS
   (1)          000000              PR0=    0                ;;PRIORITY LEVEL 0
   (1)          000040              PR1=    40               ;;PRIORITY LEVEL 1
   (1)          000100              PR2=    100              ;;PRIORITY LEVEL 2
   (1)          000140              PR3=    140              ;;PRIORITY LEVEL 3
   (1)          000200              PR4=    200              ;;PRIORITY LEVEL 4
   (1)          000240              PR5=    240              ;;PRIORITY LEVEL 5
   (1)          000300              PR6=    300              ;;PRIORITY LEVEL 6
   (1)          000340              PR7=    340              ;;PRIORITY LEVEL 7
   (1)
   (1)                              ;*"SWITCH REGISTER" SWITCH DEFINITIONS
   (1)          100000              SW15=   100000
   (1)          040000              SW14=   40000
   (1)          020000              SW13=   20000
   (1)          010000              SW12=   10000
  (1)        004000            SW11=   4000
  (1)        002000            SW10=   2000
  (1)        001000            SW09=   1000
  (1)        000400            SW08=   400
  (1)        000200            SW07=   200
  (1)        000100            SW06=   100
  (1)        000040            SW05=   40
  (1)        000020            SW04=   20
  (1)        000010            SW03=   10
  (1)        000004            SW02=   4
  (1)        000002            SW01=   2
  (1)        000001            SW00=   1
  (1)                          .EQUIV  SW09,SW9
  (1)                          .EQUIV  SW08,SW8
  (1)                          .EQUIV  SW07,SW7
  (1)                          .EQUIV  SW06,SW6
  (1)                          .EQUIV  SW05,SW5
  (1)                          .EQUIV  SW04,SW4
  (1)                          .EQUIV  SW03,SW3
  (1)                          .EQUIV  SW02,SW2
  (1)                          .EQUIV  SW01,SW1
  (1)                          .EQUIV  SW00,SW0
  (1)
  (1)                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
  (1)        100000            BIT15=  100000
  (1)        040000            BIT14=  40000
  (1)        020000            BIT13=  20000
  (1)        010000            BIT12=  10000
  (1)        004000            BIT11=  4000
  (1)        002000            BIT10=  2000
  (1)        001000            BIT09=  1000
  (1)        000400            BIT08=  400
  (1)        000200            BIT07=  200
  (1)        000100            BIT06=  100
  (1)        000040            BIT05=  40
  (1)        000020            BIT04=  20
  (1)        000010            BIT03=  10
  (1)        000004            BIT02=  4
  (1)        000002            BIT01=  2
  (1)        000001            BIT00=  1
  (1)                          .EQUIV  BIT09,BIT9
  (1)                          .EQUIV  BIT08,BIT8
  (1)                          .EQUIV  BIT07,BIT7
  (1)                          .EQUIV  BIT06,BIT6
  (1)                          .EQUIV  BIT05,BIT5
  (1)                          .EQUIV  BIT04,BIT4
  (1)                          .EQUIV  BIT03,BIT3
  (1)                          .EQUIV  BIT02,BIT2
  (1)                          .EQUIV  BIT01,BIT1
  (1)                          .EQUIV  BIT00,BIT0
  (1)
  (1)                          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
  (1)        000004            ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
  (1)        000010            RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
  (1)        000014            TBITVEC=14              ;;"T" BIT
  (1)        000014            TRTVEC= 14              ;;TRACE TRAP
```

```
   (1)          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
   (1)          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
   (1)          000024                  PWRVEC= 24              ;;POWER FAIL
   (1)          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
   (1)          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   (1)          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   (1)          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
   (1)          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
  1164                                  .SBTTL  OPERATIONAL SWITCH SETTINGS
   (1)                                  ;*
   (1)                                  ;*      SWITCH                  USE
   (1)                                  ;*      ------          --------------------
   (1)                                  ;*        15            HALT ON ERROR
   (1)                                  ;*        14            LOOP ON TEST
   (1)                                  ;*        13            INHIBIT ERROR TYPEOUTS
   (1)                                  ;*        12            HALT FOR BIT-MAP DISPLAY
   (1)                                  ;*        11            INHIBIT ITERATIONS
   (1)                                  ;*        10            BELL ON ERROR
   (1)                                  ;*         9            LOOP ON ERROR
   (1)                                  ;*         8            LOOP ON TEST IN SWR<7:0>
  1165          170400                  ABASE=  170400
  1166          140340                  AVECT1= 140340
  1167          000300                  APRIOR= 300
  1168
  1173
  1180
  1185
  1192
  1197
  1203
  1209
  1214
  1215                                  .SBTTL  TRAP CATCHER
   (1)
   (1)          000000                          .=0
   (1)                                  ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
   (1)                                  ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
   (1)                                  ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
   (1)          000174                          .=174
   (1)  000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
   (1)  000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
   (1)                                  .SBTTL  STARTING ADDRESS(ES)
   (1)  000200  000137  001712                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
  1216  000204  000137  002402                  JMP     @#BEG2          ;RESTART ADDRESS
  1217  000210  000137  001720                  JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
  1218  000214  000137  020346                  JMP     @#$UTK                  ;STARTING ADDRESS FOR USER LINK
```

```
 1220                              .SBTTL  ACT11 HOOKS
  (1)
  (2)                              ;;*****************************************************************
  (1)                              ;HOOKS REQUIRED BY ACT11
  (1)          000220                      $SVPC=.                  ;SAVE PC
  (1)          000046                      .=46
  (1)  000046  012074                      $ENDAD                   ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
  (1)          000052                      .=52
  (1)  000052  000000                      .WORD   0                ;;2)SET LOC.52 TO ZERO
  (1)          000220                      .=$SVPC                  ;; RESTORE PC
 1221          001000              .=1000
 1222                              .SBTTL  APT PARAMETER BLOCK
  (1)
  (2)                              ;;*****************************************************************
  (1)                              ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  (2)                              ;;*****************************************************************
  (1)          001000                      .$X=.   ;;SAVE CURRENT LOCATION
  (1)          000024                      .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
  (1)  000024  000200                      200     ;;FOR APT START UP
  (1)          000044                      .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
  (1)  000044  001000                      $APTHDR ;;POINT TO APT HEADER BLOCK
  (1)          001000                      .=.$X   ;;RESET LOCATION COUNTER
  (2)                              ;;*****************************************************************
  (1)                              ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
  (1)                              ;INTERFACE SPEC.
  (1)
  (1)  001000                      $APTHD:
  (1)  001000  000000              $HIBTS: .WORD   0        ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
  (1)  001002  001174              $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
  (1)  001004  002260              $TSTM:  .WORD   1200.    ;;RUN TIM OF LONGEST TEST
  (1)  001006  000764              $PASTM: .WORD   500.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  (1)  001010  003244              $UNITM: .WORD   1700.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  (1)  001012  000031                      .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  1223                              .SBTTL  COMMON TAGS
   (1)
   (2)                              ;;*******************************************************************
   (1)                              ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
   (1)                              ;*USED IN THE PROGRAM.
   (1)
   (1)          001100                      .=1100
   (1)  001100                      $CMTAG:                         ;;START OF COMMON TAGS
   (1)  001100  000000                      .WORD   0
   (1)  001102     000              $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
   (1)  001103     000              $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
   (1)  001104  000000              $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
   (1)  001106  000000              $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
   (1)  001110  000000              $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
   (1)  001112  000000              $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
   (1)  001114     000              $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
   (1)  001115     001              $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
   (1)  001116  000000              $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
   (1)  001120  000000              $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
   (1)  001122  000000              $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
   (1)  001124  000000              $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
   (1)  001126  000000              $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
   (1)  001130  000000                      .WORD   0               ;;RESERVED--NOT TO BE USED
   (1)  001132  000000                      .WORD   0
   (1)  001134     000              $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
   (1)  001135     000              $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
   (1)  001136  000000                      .WORD   0
   (1)  001140  177570              SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
   (1)  001142  177570              DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
   (1)  001144  177560              $TKS:   177560                  ;;TTY KBD STATUS
   (1)  001146  177562              $TKB:   177562                  ;;TTY KBD BUFFER
   (1)  001150  177564              $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
   (1)  001152  177566              $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
   (1)  001154     000              $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
   (1)  001155     002              $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
   (1)  001156     012              $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
   (1)  001157     000              $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
   (1)  001160  000000              $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
   (1)  001162  000000              $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
   (1)  001164  177607  000377      $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
   (1)  001170     077              $QUES:  .ASCII  /?/             ;;QUESTION MARK
   (1)  001171     015              $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
   (1)  001172  000012              $LF:    .ASCIZ  <12>            ;;LINE FEED
   (2)                              ;;*******************************************************************
   (2)                              .SBTTL  APT MAILBOX-ETABLE
   (2)
   (3)                              ;;*******************************************************************
   (2)                              .EVEN
   (2)  001174                      $MAIL:                  ;;APT MAILBOX
   (2)  001174  000000              $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
   (2)  001176  000000              $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
   (2)  001200  000000              $TESTN: .WORD   ATESTN  ;;TEST NUMBER
   (2)  001202  000000              $PASS:  .WORD   APASS   ;;PASS COUNT
   (2)  001204  000000              $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
   (2)  001206  000000              $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
   (2)  001210  000000              $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
   (2)  001212  000000            $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
   (2)  001214                    $ETABLE:                ;;APT ENVIRONMENT TABLE
   (2)  001214     000            $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
   (2)  001215     000            $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
   (2)  001216  000000            $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
   (2)  001220  000000            $USWR:  .WORD   AUSWR   ;;USER SWITCHES
   (2)  001222  000000            $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
   (2)                            ;*                      BITS 15-11=CPU TYPE
   (2)                            ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
   (2)                            ;*                              11/70=06,PDQ=07,Q=10
   (2)                            ;*                      BIT 10=REAL TIME CLOCK
   (2)                            ;*                      BIT  9=FLOATING POINT PROCESSOR
   (2)                            ;*                      BIT  8=MEMORY MANAGEMENT
   (2)  001224     000            $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
   (2)  001225     000            $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
   (2)                            ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
   (2)                            ;*                              900 NSEC CORE=001
   (2)                            ;*                              300 NSEC BIPOLAR=002
   (2)                            ;*                              500 NSEC MOS=003
   (2)  001226  000000            $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
   (2)                            ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
   (2)  001230     000            $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
   (2)  001231     000            $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
   (2)  001232  000000            $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
   (2)  001234     000            $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
   (2)  001235     000            $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
   (2)  001236  000000            $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
   (2)  001240     000            $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
   (2)  001241     000            $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE BLK#4
   (2)  001242  000000            $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
   (2)  001244  140340            $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
   (2)  001246  000000            $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
   (2)  001250  170400            $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
   (2)  001252  000000            $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
   (2)  001254  000000            $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
   (2)  001256                    $ETEND:
   (2)                            .MEXIT
```

```
    (1)                                  .SBTTL  ERROR POINTER TABLE
    (1)
    (1)                                  ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
    (1)                                  ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
    (1)                                  ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
    (1)                                  ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
    (1)                                  ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
    (1)
    (1)                                  ;*      EM              ;;POINTS TO THE ERROR MESSAGE
    (1)                                  ;*      DH              ;;POINTS TO THE DATA HEADER
    (1)                                  ;*      DT              ;;POINTS TO THE DATA
    (1)                                  ;*      DF              ;;POINTS TO THE DATA FORMAT
    (1)
    (1)
    (1)  001256                          $ERRTB:
   1225
   1226
   1227
   1236                                  ;ITEM   1
   1237  001256  014245                          EM1                     ;STATUS REG. ERROR
   1238  001260  014405                          DH1                     ;ERRPC STREG EXPECTED ACTUAL
   1239  001262  014570                          DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
   1240  001264  014630                          DF1
   1241
   1242
   1243                                  ;ITEM   2
   1244  001266  014273                          EM2                     ;FAILED TO INTERRUPT
   1245  001270  014526                          DH3                     ;ERRPC STREG ACTUAL
   1246  001272  014620                          DT3                     ;$ERRPC, STREG,  $BDDAT
   1247  001274  014630                          DF1
   1248
   1249                                  ;ITEM   3
   1250  001276  014323                          EM3                     ;UNEXPECTED INTERRUPT
   1251  001300  014526                          DH3                     ;ERRPC STREG
   1252  001302  014620                          DT3                     ;$ERRPC, STREG
   1253  001304  014630                          DF1
   1254
   1255                                  ;ITEM   4
   1256  001306  014354                          EM4                     ;ERROR ON A/D CHANNEL
   1257  001310  014443                          DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
   1258  001312  014602                          DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT
   1259  001314  014630                          DF1
   1260
   1261
```

```
 1263                                 .SBTTL        MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
 1264  001316  170400                 STREG:  ABASE                  ;ADDRESS OF STATUS REGISTER
 1265  001320  170402                 ADBUFF: ABASE+2                ;ADDRESS OF A/D BUFFER
 1266  001322  000300                 BASEBR: APRIOR                 ;INTERRUPT PRIORITY LEVEL
 1267  001324  140342                 VECTR1: AVECT1+2
 1268  001326  000040                 VADR:   40                     ;INCREMENT FOR BUS ADDRESS
 1269  001330  000040                 VVCT:   40                     ;INCREMENT FOR VECTOR ADDRESS
 1270  001332  000000                 BASECH: 0                      ;BASE CHANNEL
 1271  001334  000060                 KBVECT: 60
 1272  001336  000000                 WIDE:   0                      ;NO. OF WIDE STATES
 1273  001340  000000                 NARROW: 0                      ;NO. OF NARROW STATES
 1274  001342  000000                 FIRST:  0
 1275  001344  000000                 SKIPST: 0                      ;NO. OF SKIPPED STATES
 1276  001346  000000                 TEMP:   0                      ;WORK AREA
 1277  001350  000000                 CH1:    0                      ;FIRST CHANNEL
 1278  001352  000000                 CH2:    0                      ;SECOND CHANNEL
 1279  001354  000000                 NBEXT:  0                      ;NO. OF AD11K'S TO BE TESTED
 1280  001356  000000                 NMBEXT: 0                      ;NO. OF AD11K'S TO BE TESTED
 1281  001360  000000                 DUMMY:  0                      ;DUMMY CHANNEL
 1282  001362  000000                 CHANL:  0                      ;CHANNEL VALUE
 1283  001364  000000                 TADDR:  0                      ;TEST ADDRESS
 1284  001366  000000                 RNA:    0                      ;RANDOM
 1285  001370  000000                 RNB:    0                      ;NUMBER
 1286  001372  000000                 RNC:    0                      ;VALUES
 1287  001374  000000                 RMS:    0                      ;RMS NOISE VALUE
 1288  001376  000000                 PEAK:   0                      ;PEAK NOISE VALUE
 1289  001400  000000                 FLAG:   0                      ;VT55 FLAG
 1290  001402  000000                 SPREAD: 0                      ;DEVIATION FROM THE NOMINAL
 1291  001404  000000                 DAC:    0                      ;SAR VALUE
 1292  001406  000000                 DELAY:  0                      ;TIME DELAY COUNTER
 1293  001410  000000                 EDGE:   0                      ;EDGE VALUE
 1294  001412  000000                 BITPNT: 0
 1295  001414  000000                 MIN:    0                      ;MIN VALUE
 1296  001416  000000                 WFTEST: 0                      ;OPTION TEST AREA FLAG
 1297  001420  000000                 MAX:    0                      ;MAX VALUE
 1298  001422  000000                 PERCNT: 0                      ;PERCENT FOR SAR ROUTINE
 1299  001424  000000                 OUT:    0
 1300  001426  000000                 MYTEMP: 0
 1301  001430  000000                 $TEMP1: 0
 1302  001432  000000                 $TEMP2: 0
 1304
 1305                                                  ;
  (1)                                                  ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO MAY BE
  (1)                                                  ;                               CHANGED BY THE USER TO REFLECT
  (1)                                                  ;                               A DIFFERENT KMC-11 ADDR. THE
  (1)                                                  ;                               REST OF THE ADDRESSES WILL
  (1)                                                  ;                               BE CHANGED BY THE PROGRAM.
  (1)                                                  ;
  (1)
  (1)   001434                                 LPCI:
  (1)   001434  170460                         KMADO:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.
  (1)
  (1)   001436                                 LPMR:
  (1)   001436  170461                         KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
  (1)   001440                                 LPCO:
  (1)   001440  170462                         KMAD2:  .WORD   170460+2                ;>PATCH         <;
  (1)   001442                                 LPSO:
  (1)   001442  170463                         KMAD3:  .WORD   170460+3                ;>THIS AREA     <
  (1)   001444                                 LPADL:
  (1)   001444  170464                         KMAD4:  .WORD   170460+4                ;
  (1)   001446                                 LPADH:
  (1)   001446  170465                         KMAD5:  .WORD   170460+5                ;>DO NOT        <
  (1)   001450                                 LPMS1:
  (1)   001450  170466                         KMAD6:  .WORD   170460+6                ;>PATCH         <
  (1)   001452                                 IPMS2:
  (1)   001452  170467                         KMAD7:  .WORD   170460+7                ;>THIS AREA     <
  (1)
  (1)   001454  000340                         VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
  (1)   001456  000344                         VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2
  (1)
  (1)   001460  000005                         VERSN:  .WORD   5               ;CURRENT VERSION NUMBER OF MICROCODE.
  (1)
  (1)   001462  000000                         .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
  (1)   001464  000020                                 .BLKW   16.             ;/BY INIT.
  (1)
 1306
 1307   001524                                 UNEXP:
  (1)   001524  012737  001540  001162                 MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
 1308   001532  005237  001103                         INC     $ERFLG
 1309   001536  104003                                 ERROR   3
 1310   001540  005037  001162                 1$:     CLR     $ESCAPE         ;RETURN ESCAPE TO NORMAL
 1311   001544  000002                                 RTI                     ;UNEXPECTED INTERRUPT
```

```
                                              .SBTTL       CONTROL A AND C DECODERS
      001546  010046                          ISERV:  MOV       R0,-(SP)                      ;SAVE R0
      001550  017700  177372                          MOV       @$TKB,R0                      ;GET CHARACTER
      001554  042700  177600                          BIC       #177600,R0
      001560  120027  000003                          CMPB      R0,#3                         ;IS IT ^C?
      001564  001010                                  BNE       1$
      001566  104401  012246                          TYPE      ,CMSG                         ;ECHO CHARACTER
      001572  012706  001100                          MOV       #STACK,SP
      001576  004737  011350                          JSR       PC,RST                        ;RESET & SET INTRPT. EN.
      001602  000137  002402                          JMP       BEG2
      001606  120027  000001                  1$:     CMPB      R0,#1                         ;IS IT ^A?
      001612  001010                                  BNE       2$
      001614  104401  012241                          TYPE      ,AMSG                         ;ECHO CHARACTER
      001620  012706  001100                          MOV       #STACK,SP
      001624  004737  011350                          JSR       PC,RST                        ;RESET & SET INTRPT. EN.
      001630  000177  177530                          JMP       @TADDR                        ;RETURN TO TEST
      001634  120027  000007                  2$:     CMPB      R0,#7                         ;IS IT ^G?
      001640  001021                                  BNE       NONE
      001642  023727  001140  177570                  CMP       SWR,#177570                   ;HARDWARE SWREG?
      001650  001415                                  BEQ       NONE
      001652  104401  012253                          TYPE      ,GMSG                         ;ECHO CHARACTER
      001656  017746  177256                          MOV       @SWR,-(SP)            ;;SAVE @SWR FOR TYPEOUT
 (1)                                                                                ;;TYPE SWREG
 (1)  001662  104403                                  TYPOS                         ;;GO TYPE--OCTAL ASCII
 (1)  001664     006                                  .BYTE     6                   ;;TYPE 6 DIGITS
 (1)  001665     001                                  .BYTE     1                   ;;TYPE LEADING ZEROS
      001666  104401  012433                          TYPE      ,SLASH
      001672  104410                                  RDOCT                                   ;READ NEW VALUE
      001674  012677  177240                          MOV       (SP)+,@SWR                    ;LOAD NEW SWREG VALUE
      001700  012600                          POPR0:  MOV       (SP)+,R0
      001702  000002                          RETURN: RTI
      001704  104401  012237                  NONE:   TYPE      ,QUEST                        ;TYPE "?"
      001710  000773                                  BR        POPR0
```

```
                                              .SBTTL      INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
      001712  005037  001416                  BEGIN:  CLR     WFTEST
      001716  000403                                  BR      RBEG
      001720  012737  000001  001416          BEGIN2: MOV     #1,WFTEST
      001726                                  RBEG:   ;RESET
                                              .SBTTL  INITIALIZE THE COMMON TAGS
 (1)                                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 (1)  001726  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 (1)  001732  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 (1)  001734  022706  001140                          CMP     #SWR,R6 ;;DONE?
 (1)  001740  001374                                  BNE     .-6             ;;LOOP BACK IF NO
 (1)  001742  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 (1)                                          ;;INITIALIZE A FEW VECTORS
 (1)  001746  012737  015226  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 (1)  001754  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 (1)  001762  012737  015504  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 (1)  001770  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 (1)  001776  012737  021324  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 (1)  002004  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
 (1)  002012  012737  021402  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 (1)  002020  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 (1)  002026  013737  012042  012034                  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 (1)  002034  005037  001160                          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 (1)  002040  005037  001162                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 (1)  002044  112737  000001  001115                  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 (1)  002052  012737  002052  001106                  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 (1)  002060  012737  002060  001110                  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 (2)                                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 (2)                                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 (2)  002066  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 (2)  002072  012737  002126  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 (2)  002100  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 (2)  002106  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 (2)  002114  022777  177777  177016                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 (2)  002122  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 (2)                                                                          ;;AND  THE HARDWARE SWR IS NOT = -1
 (2)  002124  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
 (2)  002126  012716  002134                  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 (2)  002132  000002                                  RTI
 (2)  002134  012737  000176  001140          65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 (2)  002142  012737  000174  001142                  MOV     #DISPREG,DISPLAY
 (2)  002150  012637  000004                  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 (1)
 (2)  002154  005037  001202                          CLR     $PASS           ;;CLEAR PASS COUNT
 (2)  002160  132737  000200  001215                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
 (2)  002166  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
 (2)  002170  012737  001216  001140                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
 (2)  002176                                  67$:
```

```
 1350                                                ;
  (1)                                                ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
  (1)                                                ;
  (1)
  (1)   002176  010046                               MOV     R0,-(SP)
  (1)   002200  010146                               MOV     R1,-(SP)
  (1)   002202  013700  001434                       MOV     KMAD0,R0        ;GET KMC-11 ADDRESS.
  (1)   002206  012701  001436                       MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.
  (1)
  (1)   002212  005200                       68$:    INC     R0              ;UPDATE ADDR.
  (1)   002214  010021                               MOV     R0,(1)+         ;WRITE ADDR.
  (1)   002216  020127  001454                       CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
  (1)   002222  001373                               BNE     68$             ;NO - DO NEXT ADDR.
  (1)   002224  005037  001462                       CLR     .DVLS           ;CLR ADDR. LIST.
  (1)   002230  012601                               MOV     (SP)+,R1
  (1)   002232  012600                               MOV     (SP)+,R0
 1351   002234  005037  001400                       CLR     FLAG            ;CLEAR VT55 FLAG
 1352   002240  005737  000042                       TST     @#42                    ;IS IT CHAINED?
 1353   002244  001033                               BNE     REST1
 1354                                        .SBTTL        DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
 1355   002246  042777  000100  176670               BIC     #100,@$TKS
 1356   002254  104401  013673                       TYPE    ,C0             ;TYPE ASCIZ STRING
 1357   002260  004737  002664                       JSR     PC,VTFLG        ;GET A CHARACTER
 1358   002264  020027  000033                       CMP     R0,#33
 1359   002270  001017                               BNE     NOVT55          ;NO VT55 PRESENT
 1360   002272  004737  002664                       JSR     PC,VTFLG        ;GET A CHARACTER
 1361   002276  020027  000057                       CMP     R0,#57
 1362   002302  001012                               BNE     NOVT55          ;NO VT55 PRESENT
 1363   002304  004737  002664                       JSR     PC,VTFLG        ;GET A CHARACTER
 1364   002310  020027  000103                       CMP     R0,#103
 1365   002314  001403                               BEQ     VT55            ;VT55 IS PRESENT
 1366   002316  020027  000105                       CMP     R0,#105
 1367   002322  001002                               BNE     NOVT55
 1368   002324  005237  001400               VT55:   INC     FLAG
```

```
                                              ;       DIALOGUE TO DETERMINE WHICH TEST TO RUN
002330  104401  014036                NOVT55: TYPE    ,HEAD1
002334  004737  005364                REST1:  JSR     PC,FIXONE               ;INITIALIZE ADDRESSES
002340  013700  001334                        MOV     KBVECT,R0
002344  012720  001546                        MOV     #ISERV,(R0)+
002350  012710  000340                        MOV     #340,(R0)
002354  012737  062341  001366                MOV     #62341,RNA              ;RANDOM NO. VARIABLES
002362  012737  142315  001370                MOV     #142315,RNB
002370  012737  127623  001372                MOV     #127623,RNC
002376  004737  011636                        JSR     PC,WFADJ                ;STANDARD OR OPTION TEST TOLERANCES?
002402  012706  001100                BEG2:   MOV     #STACK,SP               ;RESET STACK IN CASE RESTARTED
002406  005737  000042                        TST     @#42                    ;IS IT CHAINED?
002412  001402                                BEQ     1$
002414  000137  005122                        JMP     BEGL                    ;GO TO LOGIC TESTS
002420  104401  013501                1$:     TYPE    ,MSG71
002424  104407                        TRYAG:  RDLIN
002426  052777  000100  176510                BIS     #100,@$TKS
002434  005037  177776                        CLR     PSW
002440  012600                                MOV     (SP)+,R0                ;READ ANSWER
002442  142710  000040                        BICB    #40,(R0)
002446  121027  000101                        CMPB    (R0),#'A                ;IS IT A?
002452  001002                                BNE     1$              ;;NO, TRY C
002454  000137  005160                        JMP     BEGINA                  ;GO TO AUTO TEST
002460  121027  000103                1$:     CMPB    (R0),#'C                ;IS IT C?
002464  001002                                BNE     2$              ;;NO, TRY L
002466  000137  004664                        JMP     BEGINC                  ;GO TO CALIBRATION TEST
002472  121027  000114                2$:     CMPB    (R0),#'L                ;IS IT L?
002476  001002                                BNE     3$              ;;NO, TRY N
002500  000137  005122                        JMP     BEGL                    ;GO TO LOGIC TESTS
002504  121027  000116                3$:     CMPB    (R0),#'N                ;IS IT N?
002510  001002                                BNE     4$              ;;NO, TRY S
002512  000137  005526                        JMP     BEGINN                  ;GO TO NOISE TEST
002516  121027  000123                4$:     CMPB    (R0),#'S                ;IS IT S?
002522  001002                                BNE     5$              ;;NO, TRY W
002524  000137  005576                        JMP     BEGINS                  ;GO TO SETTLE TEST
002530  121027  000127                5$:     CMPB    (R0),#'W                ;IS IT W?
002534  001002                                BNE     6$              ;;NO,TRY AGAIN
002536  000137  005246                        JMP     BEGINW                  ;GO TO WRAPAROUND TEST
002542  104401  012237                6$:     TYPE    ,QUEST
002546  000726                                BR      TRYAG                   ;WAIT FOR CHARACTER
```

```


                                        ;SIZE AND REPORT THE NUMBER OF AD11K DETECTED

     002550  013737  001250  001126  TESTAD: MOV     $BASE,$BDDAT     ;SETUP TO TEST FOR AD11K'S
     002556  005037  001462                  CLR     .DVLS
     002562  005037  001464                  CLR     .DVLS+2
     002566  005037  001354                  CLR     NBEXT            ;CLEAR AD11K COUNTER
     002572                          1$:                              ;ADDRESS AD11K

 (1)                                 ;*      MOV     $GDDAT,@$BDDAT   ;/ PUT DATA FROM $GDDAT TO DEVICE REG $BDDAT
     002602  005737  017252                  TST     $AERR            ;DEVICE EXSIST? =0,YES
     002606  001006                          BNE     2$               ;=1,NO.

     002610  005237  001354                  INC     NBEXT            ;INCREMENT AD11K COUNTER
     002614  063737  001326  001126          ADD     VADR,$BDDAT      ;GET NEXT AD11K
     002622  000763                          BR      1$               ;;TRY NEXT AD11K
     002624  122737  000001  001134  2$:     CMPB    #1,$AUTOB        ;TEST IF AUTO MODE
     002632  001406                          BEQ     3$               ;BR IF YES
     002634  013746  001354                  MOV     NBEXT,-(SP)      ;;SAVE NBEXT FOR TYPEOUT
 (1)                                                                  ;;TYPE NUMBER OF AD11K'S
 (1) 002640  104403                          TYPOS                    ;;GO TYPE--OCTAL ASCII
 (1) 002642     002                          .BYTE   2                ;;TYPE 2 DIGIT(S)
 (1) 002643     000                          .BYTE   0                ;;SUPPRESS LEADING ZEROS
     002644  104401  013041                  TYPE    ,MSG50
     002650  005337  001354          3$:     DEC     NBEXT            ;ADJUST AD11K COUNT
     002654  013737  001354  001356          MOV     NBEXT,NMBEXT     ;KEEP COUNT OF NUMBER
     002662  000207                          RTS     PC

     002664  005000                  VTFLG:  CLR     R0               ;TEST FOR PRESENCE
     002666  105777  176252          1$:     TSTB    @$TKS            ;OF VT55
     002672  100404                          BMI     2$               ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
     002674  005300                          DEC     R0
     002676  001373                          BNE     1$               ;;
     002700  005726                          TST     (SP)+            ;POP A WORD OFF STACK
     002702  000612                          BR      NOVT55           ;;NO VT55 PRESENT
     002704  017700  176236          2$:     MOV     @$TKB,R0
     002710  042700  177600                  BIC     #177600,R0               ;TEST VT55 CODE
     002714  000207                          RTS     PC
 1445  002716                             BEGINL:
 1446                                     ;;*************************************************************
  (3)                                     ;*TEST 1        FLOAT A ONE THRU MULTIPLEXER BITS
  (3)                                     ;;*************************************************************
  (2)  002716  012737  002716  001106     TST1:   MOV     #TST1,$LPADR
 1447  002724  012737  002716  001110             MOV     #TST1,$LPERR
 1448  002732  012737  000400  001124             MOV     #BIT8,$GDDAT              ;LOAD FIRST BIT
 1449  002740  004737  003406             2$:     JSR     PC,TESTIT
 1450  002744  104001                             ERROR   1                         ;FAILED TO LOAD + READ BIT
 1451  002746  006137  001124             1$:     ROL     $GDDAT                    ;GET NEXT BIT
 1452  002752  023727  001124  040000             CMP     $GDDAT,#BIT14             ;FINISHED?
 1453  002760  001367                             BNE     2$              ;;NO,GO TO NEXT TEST
 1454
 1455                                     ;;*************************************************************
  (3)                                     ;*TEST 2        LOAD AND READ BACK INTERRUPT ENABLE BIT6
  (3)                                     ;;*************************************************************
  (2)  002762  000004                     TST2:   SCOPE
 1456  002764  012777  001524  176462             MOV     #UNEXP,@VECTOR            ;SETUP FOR UNEXPECTED INTERUPT
 1457  002772  012737  000100  001124             MOV     #BIT6,$GDDAT              ;LOAD EXPECTED DATA
 1458  003000  004737  003406                     JSR     PC,TESTIT
 1459  003004  104001                             ERROR   1                         ;FAILED TO LOAD + READ INTERRUPT ENABLE
 1460
 1461                                     ;;*************************************************************
  (3)                                     ;*TEST 3        LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
  (3)                                     ;;*************************************************************
  (2)  003006  000004                     TST3:   SCOPE
 1462  003010  012737  000040  001124             MOV     #BIT5,$GDDAT              ;LOAD EXPECTED DATA
 1463  003016  004737  003406                     JSR     PC,TESTIT
 1464  003022  104001                             ERROR   1                         ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB
 1465
 1466                                     ;;*************************************************************
  (3)                                     ;*TEST 4        LOAD AND READ BACK EXTERNAL START ENABLE BIT4
  (3)                                     ;;*************************************************************
  (2)  003024  000004                     TST4:   SCOPE
 1467  003026  012737  000020  001124             MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
 1468  003034  004737  003406                     JSR     PC,TESTIT
 1469  003040  104001                             ERROR   1               ;FAILED TO LOAD + READ EXT. START ENABLE
 1470                                     ;;*************************************************************
  (3)                                     ;*TEST 5        LOAD AND READ BACK ERROR FLAG BIT15
  (3)                                     ;;*************************************************************
  (2)  003042  000004                     TST5:   SCOPE
 1471  003044  012737  100000  001124             MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
 1472  003052  004737  003406                     JSR     PC,TESTIT
 1473  003056  104001                             ERROR   1               ;FAILED TO LOAD + READ ERROR FLAG
```

```
1475                                    ;;*****************************************************************
 (3)                                    ;*TEST 6          TEST  DONE FLAG  SETS AND BIT0 CLEARS ON END OF CONV.
 (3)                                    ;;*****************************************************************
 (2)  003060  000004                    TST6:   SCOPE
1476  003062  012700  001000                    MOV     #BIT9,R0         ;STALL TIME COUNTER
1477
 (2)
 (2)                                    ;*      MOV     @STREG,MYTEMP    ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1)  003076  005237  001426                    INC     MYTEMP
 (2)
 (2)                                    ;*      MOV     MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1478  003112  012737  000200  001124            MOV     #BIT7,$GDDAT     ;LOAD EXPECTED
1479  003120  005300                    1$:     DEC     R0               ;STALL
1480  003122  001376                            BNE     1$               ;TIME
1481
 (2)
 (2)                                    ;*      MOV     @STREG,MYTEMP    ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1)  003134  042737  100000  001426            BIC     #BIT15,MYTEMP
 (2)
 (2)                                    ;*      MOV     MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1482  003152  004737  003416                    JSR     PC,TEST
1483  003156  104001                            ERROR   1                ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
1484
 (2)                                    ;*      MOV     @ADBUFF,MYTEMP   ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
 (1)  003170  013700  001426                    MOV     MYTEMP,R0        ;/PUT CONVERTED VALUE IN R0.
1485
1486                                    ;;*****************************************************************
 (3)                                    ;*TEST 7          TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
 (3)                                    ;;*****************************************************************
 (2)  003174  000004                    TST7:   SCOPE
1487  003176  012737  000001  001426            MOV     #BIT0,MYTEMP
1488
 (1)                                    ;*      MOV     MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
1489  003214  005037  001124                    CLR     $GDDAT
1490  003220                            1$:
 (2)
 (2)                                    ;*      MOV     @STREG,MYTEMP    ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1)  003230  105737  001426                    TSTB    MYTEMP
1491  003234  100371                            BPL     1$
1492
 (2)                                    ;*      MOV     @ADBUFF,MYTEMP   ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
 (1)  003246  013700  001426                    MOV     MYTEMP,R0        ;/PUT CONVERTED VALUE IN R0.
1493  003252  004737  003416                    JSR     PC,TEST
1494  003256  104001                            ERROR   1                ;DONE FLAG FAILED TO CLEAR
```

```
 1496                                  ;;*************************************************************
  (3)                                  ;*TEST 10       TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
  (3)                                  ;;*************************************************************
  (2)  003260  000004                  TST10:  SCOPE
  (1)  003262  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1497  003270  012737  000001  001426          MOV     #BIT0,MYTEMP
 1498
  (1)                                  ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
 1499  003306                          1$:
  (2)
  (2)                                  ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
  (1)  003316  105737  001426                  TSTB    MYTEMP
 1500  003322  100371                          BPL     1$
 1501  003324  012737  100200  001124  2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
 1502  003332  012737  000001  001426          MOV     #BIT0,MYTEMP
 1503
  (1)                                  ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
 1504  003350  012700  001000                  MOV     #BIT9,R0        ;WAIT FOR 2ND
 1505  003354  005300                  3$:     DEC     R0              ;CONVERSION TO END
 1506  003356  001376                          BNE     3$
 1507  003360  004737  003416          4$:     JSR     PC,TEST
 1508  003364  104001                          ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
 1509                                                                  ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
 1510
  (2)                                  ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
  (1)  003376  013700  001426                  MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
 1511
 1512  003402  000004                          SCOPE
 1513  003404  000207                          RTS     PC              ;RETURN TO TEST SECTION
 1514
 1515
 1516                                  ;;SUBROUTINE FOR LOGIC TESTS;;
 1517  003406                          TESTIT:
  (1)
  (1)                                  ;*      MOV     $GDDAT,@STREG   ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG
 1518  003416                          TEST:
  (1)
  (1)                                  ;*      MOV     @STREG,$BDDAT   ;/READ DEVICE REG STREG,PUT DATA IN $BDDAT.
 1519  003426  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE RESULTS
 1520  003434  001002                          BNE     RETERR          ;;ERROR RETURN
 1521  003436  062716  000002                  ADD     #2,(SP)         ;BUMP RETURN ADDRESS TO GET AROUND ERROR
 1522  003442  000207                  RETERR: RTS     PC
```

```
                                                  .SBTTL         WRAPAROUND TEST SECTION
       003444                                     WRAP:
                                                  ;;*****************************************************************
 (3)                                              ;*TEST 11        TEST CH14 GROUND
 (3)                                              ;;*****************************************************************
 (2)   003444  000240                             TST11:  NOP
 (1)   003446  012737  000010  001160                     MOV     #10,$TIMES       ;;DO 10 ITERATIONS
       003454  012737  000011  001102                     MOV     #$TN-1,$TSTNM
       003462  012737  003476  001110                     MOV     #1$,$LPERR
       003470  012737  003476  001106                     MOV     #1$,$LPADR
       003476  004537  011060             1$:     JSR     R5,CONVRT                ;DO 8 CONVERSIONS
       003502  000014                                     14
       003504  004537  011302                     JSR     R5,COMPAR                ;COMPARE RESULTS
       003510  004000                                     4000                ;NOMINAL
       003512  011714                                     V50                 ;TOLERANCE
       003514  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
                                                  ;;*****************************************************************
 (3)                                              ;*TEST 12        TEST CONVERSION FROM EXT. START
 (3)                                              ;;*****************************************************************
 (2)   003516  000004                             TST12:  SCOPE
 (1)   003520  012737  000010  001160                     MOV     #10,$TIMES       ;;DO 10 ITERATIONS
       003526  005737  001332                             TST     BASECH                   ;TESTING AN AM?
       003532  001044                                     BNE     TST13            ;;YES, GOTO NEXT TEST
       003534  012737  000020  001426                     MOV     #BIT4,MYTEMP

 (1)                                              ;*      MOV     MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
       003552  012700  001000                             MOV     #BIT9,R0                 ;TIME DELAY COUNTER
       003556  012737  000220  001124                     MOV     #BIT7!BIT4,$GDDAT        ;LOAD EXPECTED
       003564  012737  000200  001426                     MOV     #200,MYTEMP

 (1)                                              ;*      MOV     MYTEMP,@ADBUFF   ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
                                                                                           ;WRAPAROUND MODULE PRESENT
       003602  005300                             1$:     DEC     R0
       003604  001376                                     BNE     1$
       003606  004737  003416                             JSR     PC,TEST
       003612  104001                                     ERROR   1                        ;FAILED TO DO CONVERSION FROM EXT. START

 (2)                                              ;*      MOV     @ADBUFF,MYTEMP   ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
 (1)   003624  013700  001426                             MOV     MYTEMP,R0        ;/PUT CONVERTED VALUE IN R0.
       003630  005037  001426                             CLR     MYTEMP
 (2)
 (2)                                              ;*      MOV     MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
                                                  ;;*****************************************************************
 (3)                                              ;*TEST 13        TEST CH0 GROUND
 (3)                                              ;;*****************************************************************
 (2)   003644  000004                             TST13:  SCOPE
 (1)   003646  012737  000010  001160                     MOV     #10,$TIMES       ;;DO 10 ITERATIONS
       003654  004537  011060                             JSR     R5,CONVRT                ;CONVERT 8 TIMES
       003660  000000                                     0
       003662  004537  011302                             JSR     R5,COMPAR                ;COMPARE RESULTS
       003666  004000                                     4000                             ;NOMINAL
       003670  011706                                     V1                               ;TOLERANCE
       003672  104004                                     ERROR   4                        ;ERROR ON A/D CHANNEL
 1561                                          ;;*****************************************************************
  (3)                                          ;*TEST 14        TEST CH1 GROUND
  (3)                                          ;;*****************************************************************
  (2)  003674  000004                          TST14:  SCOPE
  (1)  003676  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1562  003704  004537  011060                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
 1563  003710  000001                                  1                                 ;CHANNEL 1
 1564  003712  004537  011302                          JSR     R5,COMPAR                 ;COMPARE RESULTS
 1565  003716  004000                                  4000                              ;NOMINAL
 1566  003720  011712                                  V10                               ;TOLERANCE
 1567  003722  104004                                  ERROR   4                         ;ERROR ON A/D CHANNEL
 1568
 1569                                          ;;*****************************************************************
  (3)                                          ;*TEST 15        TEST CH2 +1 VOLT
  (3)                                          ;;*****************************************************************
  (2)  003724  000004                          TST15:  SCOPE
  (1)  003726  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1570  003734  004537  011060                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
 1571  003740  000002                                  2                                 ;CHANNEL 2
 1572  003742  004537  011302                          JSR     R5,COMPAR                 ;COMPARE RESULTS
 1573  003746  004632                                  4632                              ;NOMINAL
 1574  003750  011714                                  V50                               ;TOLERANCE
 1575  003752  104004                                  ERROR   4                         ;ERROR ON A/D CHANNEL
 1576                                                                                    ;AD11K MUST BE SET UP FOR +OR- 5V OR +OR- 5.12V
 1577
 1578                                          ;;*****************************************************************
  (3)                                          ;*TEST 16        TEST CH3 +2.5 VOLTS
  (3)                                          ;;*****************************************************************
  (2)  003754  000004                          TST16:  SCOPE
  (1)  003756  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1579  003764  004537  011060                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
 1580  003770  000003                                  3                                 ;CHANNEL 3
 1581  003772  004537  011302                          JSR     R5,COMPAR                 ;COMPARE RESULTS
 1582  003776  006000                                  6000                              ;NOMINAL
 1583  004000  011722                                  V240                              ;TOLERANCE
 1584  004002  104004                                  ERROR   4                         ;ERROR ON A/D CHANNEL
 1585
 1586                                          ;;*****************************************************************
  (3)                                          ;*TEST 17        TEST CH4 -2.5 VOLTS
  (3)                                          ;;*****************************************************************
  (2)  004004  000004                          TST17:  SCOPE
  (1)  004006  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1587  004014  004537  011060                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
 1588  004020  000004                                  4                                 ;CHANNEL 4
 1589  004022  004537  011302                          JSR     R5,COMPAR                 ;COMPARE RESULTS
 1590  004026  002000                                  2000                              ;NOMINAL
 1591  004030  011722                                  V240                              ;TOLERANCE
 1592  004032  104004                                  ERROR   4
 1594                                           ;;*****************************************************************
  (3)                                           ;*TEST 20       TEST VERNIER OFFSET DAC ON CH12
  (3)                                           ;;*****************************************************************
  (2)  004034  000004                           TST20:  SCOPE
  (1)  004036  012737  000001  001160                   MOV     #1,$TIMES       ;;DO 1 ITERATION
 1595  004044  005037  001426                           CLR     MYTEMP
 1596
  (1)                                           ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
 1597  004060  004737  004654                           JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
 1598  004064  004537  011060                           JSR     R5,CONVRT               ;CONV. CH12, DIRECT VERNIER DAC
 1599  004070  000012                                   12
 1600  004072  013704  001346                           MOV     TEMP,R4                 ;SAVE VALUE IN R4
 1601  004076  004537  011302                           JSR     R5,COMPAR               ;COMPARE RESULTS
 1602  004102  002376                                   2376                            ;WITH -1.875 VOLTS
 1603  004104  011720                                   V115                            ;TOLERANCE OF 10%
 1604  004106  104004                                   ERROR   4
 1605  004110  005037  001420                           CLR     MAX
 1606  004114  012702  000001                           MOV     #1,R2
 1607  004120  010237  001426                   1$:     MOV     R2,MYTEMP               ;SET UP NEXT VERNIER DAC VALUE`
 1608
  (1)                                           ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
 1609  004134  004737  004654                           JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
 1610  004140  004537  011060                           JSR     R5,CONVRT               ;CONVERT IT
 1611  004144  000012                                   12
 1612  004146  005737  001420                           TST     MAX
 1613  004152  001010                                   BNE     2$
 1614  004154  023727  001346  004000                   CMP     TEMP,#4000
 1615  004162  002404                                   BLT     2$
 1616  004164  005237  001420                           INC     MAX
 1617  004170  010237  001414                           MOV     R2,MIN
 1618  004174  020227  000200                   2$:     CMP     R2,#200
 1619  004200  001003                                   BNE     3$
 1620  004202  013737  001346  004274                   MOV     TEMP,4$
 1621  004210  013703  001346                   3$:     MOV     TEMP,R3                 ;SAVE VALUE
 1622  004214  160437  001346                           SUB     R4,TEMP                 ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
 1623  004220  010304                                   MOV     R3,R4                   ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
 1624  004222  004537  011302                           JSR     R5,COMPAR               ;COMPARE RESULTS
 1625  004226  000006                                   6                               ;WITH 15 MILLIVOLTS(1 DAC LSB)
 1626  004230  011724                                   V5
 1627  004232  104004                                   ERROR   4
 1628  004234  005202                                   INC     R2
 1629  004236  020227  000400                           CMP     R2,#400                 ;DONE?
 1630  004242  001326                                   BNE     1$                      ;NO-DO NEXT VERNIER DAC VALUE
 1631  004244  004737  020224                           JSR     PC,$RESET
 1632  004250  052777  000100  174666                   BIS     #100,@$TKS
 1633  004256  004737  004654                           JSR     PC,DAWAIT               ;LET DAC SETTLE
 1634  004262  004537  011060                           JSR     R5,CONVRT               ;CONVERT IT
 1635  004266  000012                                   12
 1636  004270  004537  011302                           JSR     R5,COMPAR               ;COMPARE RESULTS
 1637  004274  000000                           4$:     0
 1638  004276  011710                                   V2
 1639  004300  104004                                   ERROR   4
```

```
                                        ;;*************************************************************
                                        ;*TEST 21         TEST CH13 +2.5 VOLTS
                                        ;;*************************************************************
004302  000004                          TST21:  SCOPE
004304  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004312  004537  011060                          JSR     R5,CONVRT                  ;CONVERT 8 TIMES
004316  000013                                  13
004320  004537  011302                          JSR     R5,COMPAR                  ;COMPARE RESULTS
004324  006000                                  6000                               ;NOMINAL
004326  011716                                  V144                               ;TOLERANCE
004330  104004                                  ERROR   4
                                        ;;*************************************************************
                                        ;*TEST 22         TEST CH17 +4V
                                        ;;*************************************************************
004332  000004                          TST22:  SCOPE
004334  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004342  004537  011060                          JSR     R5,CONVRT                  ;CONVERT 8 TIMES
004346  000017                                  17                                 ;CHANNEL 17
004350  004537  011302                          JSR     R5,COMPAR                  ;COMPARE RESULTS
004354  007146                                  7146                               ;NOMINAL
004356  011722                                  V240                               ;TOLERANCE
004360  104004                                  ERROR   4                          ;ERROR ON A/D CHANNEL
                                        ;;*************************************************************
                                        ;*TEST 23         OFFSET ON CH0
                                        ;;*************************************************************
004362  000004                          TST23:  SCOPE
004364  012737  000001  001160                  MOV     #1,$TIMES       ;;DO 1 ITERATION
004372  013737  001332  001362                  MOV     BASECH,CHANL               ;LOAD CHANNEL
004400  013737  001332  001360                  MOV     BASECH,DUMMY               ;LOAD DUMMY
004406  012737  004001  001410                  MOV     #4001,EDGE
004414  004537  006440                          JSR     R5,SARSUB
004420  000062                                  50.
004422  013737  001404  001346                  MOV     DAC,TEMP
004430  004537  006440                          JSR     R5,SARSUB
004434  000062                                  50.
004436  063737  001404  001346                  ADD     DAC,TEMP
004444  162737  000062  001346                  SUB     #62,TEMP
004452  013700  001414                          MOV     MIN,R0
004456  006300                                  ASL     R0
004460  160037  001346                          SUB     R0,TEMP
004464  104401  013705                          TYPE    ,MOFSET         ;TYPE ASCIZ STRING
004470  013702  001346                          MOV     TEMP,R2
004474  004737  011472                          JSR     PC,DECTYP
004500  104401  013720                          TYPE    ,MLSB           ;TYPE ASCIZ STRING
004504  004537  011302                          JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
004510  000000                                  0
004512  011726                                  V50D
004514  000401                                  BR      OFFERR          ;NO-ERROR
004516  000403                                  BR      OFFOK           ;YES-OK
004520  104401  012507                  OFFERR: TYPE    ,ERMSG
004524  000402                                  BR      TST24           ;;GO TO NEXT TEST
004526  104401  012476                  OFFOK:  TYPE    ,OKMSG
```

```
 1682                                    ;;*******************************************************************
  (3)                                    ;*TEST 24        NOISE TEST ON 8 EDGES
  (3)                                    ;;*******************************************************************
  (2)  004532  000004                    TST24:  SCOPE
  (1)  004534  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
 1683  004542  012737  000116  001346            MOV     #116,TEMP               ;DAC VALUE
 1684  004550  004537  010652                    JSR     R5,NOI8                 ;NOISE AT -FULL SCALE
 1685  004554  000015                            15
 1686  004556  004537  010652                    JSR     R5,NOI8                 ;NOISE AT MID-RANGE
 1687  004562  000007                            7
 1688  004564  004537  010652                    JSR     R5,NOI8                 ;NOISE AT +FULL SCALE
 1689  004570  000016                            16
 1690
 1691                                    ;;*******************************************************************
  (3)                                    ;*TEST 25        SETTLE TEST ON 8 EDGES
  (3)                                    ;;*******************************************************************
  (2)  004572  000004                    TST25:  SCOPE
  (1)  004574  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
 1692  004602  004537  006110                    JSR     R5,SET8                 ;SETTLE-POSITIVE DIRECTION
 1693  004606  000015                            15
 1694  004610  000016                            16
 1695  004612  012737  000116  001346            MOV     #116,TEMP
 1696  004620  004537  006110                    JSR     R5,SET8                 ;SETTLE-NEGATIVE DIRECTION
 1697  004624  000016                            16
 1698  004626  000015                            15
 1699                                    ;;*******************************************************************
  (3)                                    ;*TEST 26        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
  (3)                                    ;;*******************************************************************
  (2)  004630  000004                    TST26:  SCOPE
  (1)  004632  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
 1700  004640  005737  001202                    TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
 1701  004644  001402                            BEQ     LEND
 1702  004646  004737  006736                    JSR     PC,DIFLIN
 1703  004652  000207                    LEND:   RTS     PC                      ;RETURN TO TEST SECTION
 1704
 1705  004654  005000                    DAWAIT: CLR     R0
 1706  004656  105300                    1$:     DECB    R0
 1707  004660  001376                            BNE     1$
 1708  004662  000207                            RTS     PC
```

```
                                        .SBTTL        CALIBRATION TEST
      004664  012737  004664  001364   BEGINC: MOV     #BEGINC,TADDR           ;TEST ADDRESS IN TADDR
      004672  005037  001426                   CLR     MYTEMP
 (2)
 (2)                                   ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
      004706  104401  013615                   TYPE    ,HEAD5                  ;TYPE OUT HEADING
      004712  005037  177776                   CLR     PSW
      004716  017700  174216           1$:     MOV     @SWR,R0                 ;READ CHANNEL FROM SWITCH REG.
      004722  042700  177700                   BIC     #177700,R0              ;ISOLATE MUX BITS
      004726  032777  020000  174204           BIT     #BIT13,@SWR             ;IS BIT 13 SET?
      004734  001005                           BNE     2$              ;;YES,SKIP TYPEOUT
      004736  104401  012321                   TYPE    ,CH
      004742  010046                           MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
 (1)                                                                   ;;TYPE CHANNEL
 (1)  004744  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  004746     002                           .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  004747     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
      004750                           2$:
      004750  000300                           SWAB    R0                      ;SWITCH BYTES
      004752  010037  001426                   MOV     R0,MYTEMP
 (2)
 (2)                                   ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
      004766  012702  000010                   MOV     #10,R2                  ;TYPEOUT COUNTER
      004772                           3$:
 (1)
 (2)
 (2)                                   ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1)  005002  005237  001426                   INC     MYTEMP
 (2)
 (2)                                   ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
      005016                           30$:

 (2)                                   ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1)  005026  105737  001426                   TSTB    MYTEMP
      005032  100371                           BPL     30$

 (2)                                   ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
 (1)  005044  013700  001426                   MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
      005050  032777  020000  174062           BIT     #BIT13,@SWR             ;IS BIT 13 SET?
      005056  001403                           BEQ     4$                      ;NOT SET, TYPE OUT LIST
      005060  010077  174056                   MOV     R0,@DISPLAY                     ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
      005064  000714                           BR      1$                      ;REPEAT CONVERSION
      005066  104401  012324           4$:     TYPE    ,SPACE
      005072  010046                           MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
 (1)                                                                   ;;PRINT OCTAL CONVERTED VALUE
 (1)  005074  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  005076     004                           .BYTE   4               ;;TYPE 4 DIGIT(S)
 (1)  005077     001                           .BYTE   1               ;;TYPE LEADING ZEROS
      005100  012701  010000                   MOV     #10000,R1
      005104  005301                   5$:     DEC     R1
      005106  001376                           BNE     5$
      005110  005302                           DEC     R2                      ;DECREMENT THE COUNTER
      005112  001327                           BNE     3$                      ;NO CARRIAGE RETURN
      005114  104401  001171                   TYPE    ,$CRLF                  ;CARRIAGE RETURN
      005120  000676                           BR      1$                      ;REPEAT CONVERSION
```

```
                                        .SBTTL          LOGIC TEST SECTION
005122  012737  005122  001364  BEGL:   MOV     #BEGL,TADDR             ;TEST ADDRESS
005130  004737  002550                  JSR     PC,TESTAD               ;NO OF ADDITIONAL AD'S
005134  004737  002716          1$:     JSR     PC,BEGINL               ;LOGIC TESTS
005140  004737  005310                  JSR     PC,BUMPAD               ;MORE TO TEST?
005144  000773                          BR      1$                      ;TEST NEXT A/D
005146  012737  005134  012004          MOV     #1$,AGTST               ;ADDRESS FOR EOP
005154  000137  012006                  JMP     $EOP                    ;TYPE END OF PASS

                                        .SBTTL          AUTO TEST
005160  012737  005160  001364  BEGINA: MOV     #BEGINA,TADDR           ;TEST ADDRESS
005166  005037  001202                  CLR     $PASS                   ;CLEAR PASS COUNTER
005172  004737  002550                  JSR     PC,TESTAD               ;NO. OF AD'S TO BE TESTED
005176  004737  002716          1$:     JSR     PC,BEGINL               ;LOGIC TESTS
005202  104401  012777                  TYPE    ,MEND                   ;TYPE END OF LOGIC TEST
005206  013746  001316                  MOV     STREG,-(SP)             ;SAVE STREG FOR TYPEOUT
005212  104403                          TYPOS                           ;TYPE OCTAL NUMBER
005214     006                          .BYTE   6                       ;TYPE 6 DIGITS
005215     001                          .BYTE   1                       ;TYPE LEADING ZEROS
005216  104401  001171                  TYPE    ,$CRLF                  ;TYPE A CR,LF
005222  004737  003444                  JSR     PC,WRAP
005226  004737  005310                  JSR     PC,BUMPAD               ;TEST NEXT A/D
005232  000761                          BR      1$                      ;TEST NEXT AD
005234  012737  005176  012004          MOV     #1$,AGTST               ;ADDRESS FOR EOP
005242  000137  012006                  JMP     $EOP                    ;TYPE END OF PASS

                                        .SBTTL          WRAPAROUND TEST
005246  012737  005246  001364  BEGINW: MOV     #BEGINW,TADDR           ;TEST ADDRESS
005254  005037  001202                  CLR     $PASS                   ;CLEAR PASS COUNT
005260  004737  002550                  JSR     PC,TESTAD               ;NO. OF AD'S TO BE TESTED
005264  004737  003444          1$:     JSR     PC,WRAP                 ;WRAPAROUND TESTS
005270  004737  005310                  JSR     PC,BUMPAD               ;MORE A/D'S TO BE TESTED?
005274  000773                          BR      1$                      ;YES-GO TEST NEXT AD11K
005276  012737  005264  012004          MOV     #1$,AGTST
005304  000137  012006                  JMP     $EOP                    ;INCREMENTS $PASS
```

```
                                      ;          DETERMINE IF MORE AD11K'S TO BE TESTED
005310  005737  001354               BUMPAD: TST      NBEXT                    ;ADDITIONAL AD'S?
005314  001421                               BEQ      FIXADR                   ;NO-INITIALIZE ADDRESSES
005316  063737  001326  001316               ADD      VADR,STREG               ;SET UP NEW ST. REG.
005324  063737  001326  001320               ADD      VADR,ADBUFF              ;SET UP NEW BUFFER ADDRESS
005332  063737  001330  001454               ADD      VVCT,VECTOR              ;SET UP NEW VECTOR
005340  063737  001330  001324               ADD      VVCT,VECTR1
005346  005077  173752                       CLR      @VECTR1
005352  005337  001354                       DEC      NBEXT                    ;ONE LESS AD11K
005356  000441                               BR       BYPASS
005360  062716  000002               FIXADR: ADD      #2,(SP)
005364  013737  001250  001316       FIXONE: MOV      $BASE,STREG              ;RELOAD INITIAL ADDRESSES
005372  013737  001250  001320               MOV      $BASE,ADBUFF
005400  062737  000002  001320               ADD      #2,ADBUFF
005406  013737  001244  001454               MOV      $VECT1,VECTOR
005414  042737  170000  001454               BIC      #170000,VECTOR
005422  113737  001245  001322               MOVB     $VECT1+1,BASEBR
005430  105037  001323                       CLRB     BASEBR+1                 ;CLEAR HIGH BYTE
005434  013737  001454  001324               MOV      VECTOR,VECTR1
005442  062737  000002  001324               ADD      #2,VECTR1
005450  005077  173650                       CLR      @VECTR1
005454  013737  001356  001354               MOV      NMBEXT,NBEXT             ;RESET COUNTER
                                     ;;LOAD .+2 AND HALT TRAP CATCH;;
005462  012700  000216               BYPASS: MOV      #216,R0                  ;FILL .+2
005466  012701  000214                       MOV      #214,R1                  ;LOAD HALT
005472  020137  001334               1$:     CMP      R1,KBVECT
005476  001410                               BEQ      2$
005500  010021                               MOV      R0,(R1)+
005502  005021                               CLR      (R1)+
005504  010100                               MOV      R1,R0
005506  005720                               TST      (R0)+
005510  020027  001002                       CMP      R0,#1002
005514  001366                               BNE      1$
005516  000207                               RTS      PC                       ;TEST NEXT A/D
005520  022021                       2$:     CMP      (R0)+,(R1)+
005522  022021                               CMP      (R0)+,(R1)+
005524  000762                               BR       1$


                                     ;           NOISE TEST, 1 EDGE
005526  012737  005526  001364       BEGINN: MOV      #BEGINN,TADDR            ;TEST ADDRESS IN TADDR
005534  104401  012130                       TYPE     ,NOIMSG                  ;ASK FOR CHANNEL
005540  104401  013634                       TYPE     ,ASKCH
005544  017737  173370  001350       1$:     MOV      @SWR,CH1                 ;LOAD CHANNEL
005552  042737  177700  001350               BIC      #177700,CH1
005560  012737  000200  001346               MOV      #200,TEMP                ;LOAD DAC VALUE
005566  004537  010366                       JSR      R5,NOITST                ;GO TO NOISE SUBROUTINE
005572  001350                               CH1
005574  000763                               BR       1$
```

```
                                       ;          INTERCHANNEL SETTLING TEST, 1 EDGE
      005576  012737  005576  001364  BEGINS: MOV     #BEGINS,TADDR           ;TEST ADDPESS IN TADDR
      005604  104401  012150                  TYPE    ,SETMSG                 ;ASK FOR CHANNELS
      005610  104410                          RDOCT
      005612  012637  001350                  MOV     (SP)+,CH1
      005616  104401  012435                  TYPE    ,TOMSG
      005622  104410                          RDOCT
      005624  012637  001352                  MOV     (SP)+,CH2
      005630  012737  000200  001346  BK3:    MOV     #200,TEMP               ;LOAD DAC
      005636  013737  001352  001362          MOV     CH2,CHANL
      005644  004737  006214                  JSR     PC,GETEDG               ;GET EDGE VALUES
      005650  005002                          CLR     R2
      005652  004737  006046                  JSR     PC,SET1A                ;SCALING = .02 LSB
      005656  004737  006046                  JSR     PC,SET1A                ;MAKE IT .01 LSB
      005662  100001                          BPL     POSR2
      005664  005402                          NEG     R2
      005666  010204                  POSR2:  MOV     R2,R4
      005670  012737  000001  006436          MOV     #1,EDGFLG
      005676  004737  005704                  JSR     PC,TYPSET
      005702  000752                          BR      BK3
      005704  004737  011472          TYPSET: JSR     PC,DECTYP
      005710  104401  012331                  TYPE    ,LSB
      005714  013746  001352                  MOV     CH2,-(SP)       ;;SAVE CH2 FOR TYPEOUT
 (1)                                                                  ;;TYPE CH
 (1)  005720  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  005722     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  005723     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
      005724  104401  013726                  TYPE    ,MAT            ;TYPE ASCIZ STRING
      005730  004737  006374                  JSR     PC,TYPEDG
      005734  104401  012344                  TYPE    ,SETCH
      005740  013746  001350                  MOV     CH1,-(SP)       ;;SAVE CH1 FOR TYPEOUT
 (1)                                                                  ;;TYPE CH
 (1)  005744  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  005746     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 (1)  005747     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
      005750  104401  012366                  TYPE    ,ATMSG
      005754  013737  001350  006012          MOV     CH1,1$
      005762  163737  001332  006012          SUB     BASECH,1$
      005770  012737  000200  001426          MOV     #200,MYTEMP

 (1)                                  ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
      006006  004537  011060                  JSR     R5,CONVRT
      006012  000000                  1$:     0
      006014  013746  001346                  MOV     TEMP,-(SP)      ;;SAVE TEMP FOR TYPEOUT
 (1)                                                                  ;;TYPE VALUE
 (1)  006020  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  006022     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
 (1)  006023     001                          .BYTE   1               ;;TYPE LEADING ZEROS
      006024  020437  011734                  CMP     R4,VSET
      006030  003003                          BGT     ERR
      006032  104401  012476                  TYPE    ,OKMSG
      006036  000207                          RTS     PC
      006040  104401  012507          ERR:    TYPE    ,ERMSG
      006044  000207                          RTS     PC



                                      ;;SUBROUTINE FOR SETTLING TESTS;;
      006046  013737  001352  001360  SET1A:  MOV     CH2,DUMMY               ;LOAD DUMMY
      006054  004537  006440                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
      006060  000062                          50.
      006062  063702  001404                  ADD     DAC,R2          ;ADD RESULT TO R2
      006066  013737  001350  001360          MOV     CH1,DUMMY               ;CHANGE DUMMY VALUE
      006074  004537  006440                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
      006100  000062                          50.
      006102  163702  001404                  SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
      006106  000207                          RTS     PC              ;RETURN

      006110  012537  001350          SET8:   MOV     (R5)+,CH1               ;GET FIRST CHANNEL
      006114  012537  001352                  MOV     (R5)+,CH2               ;GET SECOND CHANNEL
      006120  063737  001332  001350          ADD     BASECH,CH1
      006126  063737  001332  001352          ADD     BASECH,CH2
      006134  004737  006214                  JSR     PC,GETEDG               ;GET EDGE VALUES
      006140  005002                          CLR     R2
      006142  012703  000010                  MOV     #10,R3          ;SET UP COUNTER
      006146  004737  006046          SETAA:  JSR     PC,SET1A                ;GET SETTLE VALUES
      006152  005237  001410                  INC     EDGE
      006156  005303                          DEC     R3
      006160  001372                          BNE     SETAA           ;REPEAT 8 TIMES
      006162  162737  000010  001410          SUB     #10,EDGE
      006170  005702                          TST     R2
      006172  100001                          BPL     R2POS
      006174  005402                          NEG     R2
      006176  010204                  R2POS:  MOV     R2,R4
      006200  012737  000010  006436          MOV     #8.,EDGFLG
      006206  004737  005704                  JSR     PC,TYPSET               ;TYPE OUT RESULTS
      006212  000205                          RTS     R5                      ;RETURN


                                      ;SUBROUTINE TO GET EDGE VALUE
                                      ;CALL=JSR   PC,GETEDG
                                      ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                      ;RESULT IN EDGE, USES R0
      006214                          GETEDG:
 (1)
 (1)                                  ;*      MOV     TEMP,@ADBUFF    ;/ PUT DATA FROM TEMP TO DEVICE REG ADBUFF
      006224  113700  001362                  MOVB    CHANL,R0                ;GET CHANNEL
      006230  000300                          SWAB    R0              ;SET UP A.D STATUS REG.
      006232  010037  001426                  MOV     R0,MYTEMP
 (2)
 (2)                                  ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
      006246  012700  000100                  MOV     #100,R0         ;DAC SETTLING DELAY
      006252  005300                  1$:     DEC     R0
      006254  001376                          BNE     1$
      006256  005037  001410                  CLR     EDGE
      006262  012700  000010                  MOV     #10,R0
      006266                          CONV:
 (1)
```

```
(2)
(2)                                     ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
(1)   006276  005237  001426                    INC     MYTEMP
(2)
(2)                                     ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
      006312                            30$:
(2)
(2)                                     ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
(1)   006322  105737  001426                    TSTB    MYTEMP
      006326  100371                            BPL     30$

(2)
(2)                                     ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
(1)   006340  063737  001426  001410            ADD     MYTEMP,EDGE
      006346  005300                            DEC     R0
      006350  001346                            BNE     CONV
      006352  006237  001410                    ASR     EDGE
      006356  006237  001410                    ASR     EDGE
      006362  006237  001410                    ASR     EDGE
      006366  005537  001410                    ADC     EDGE
      006372  000207                            RTS     PC
                                        ;;SUBROUTINE TO TYPE EDGE VALUES;;
      006374  013703  001410            TYPEDG: MOV     EDGE,R3
      006400  010346                            MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
(1)                                                                     ;;TYPE OCTAL VALUE OF EDGE
(1)   006402  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
(1)   006404     004                            .BYTE   4               ;;TYPE 4 DIGIT(S)
(1)   006405     001                            .BYTE   1               ;;TYPE LEADING ZEROS
      006406  023727  006436  000001            CMP     EDGFLG,#1
      006414  001407                            BEQ     RET
      006416  062703  000007                    ADD     #7,R3
      006422  104401  013676                    TYPE    ,C1             ;TYPE ASCIZ STRING
      006426  010346                            MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
(1)                                                                     ;;TYPE EDGE VALUE
(1)   006430  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
(1)   006432     004                            .BYTE   4               ;;TYPE 4 DIGIT(S)
(1)   006433     001                            .BYTE   1               ;;TYPE LEADING ZEROS
      006434  000207                    RET:    RTS     PC
      006436  000000                    EDGFLG: 0
```

```
                                   ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
                                   ;CALL=JSR    R5,SARSUB
                                     ;   XXX;XXX=PERCENT
                                   ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
006440  012537  001422             SARSUB: MOV     (R5)+,PERCNT            ;GET PERCENT
006444  006337  001422                     ASL     PERCNT
006450  006337  001422                     ASL     PERCNT
006454  012737  000620  006734             MOV     #400.,CNNO              ;NO OF SAMPLES FOR SHORT PASS.
006462  032777  004000  172450             BIT     #BIT11,@SWR             ;USER WANT SHORT PASS?
006470  001010                             BNE     SAR1
006472  000407                             BR      SAR1                    ;ALWAYS USE SHORT SAMPLE COUNT.
006474  012737  003100  006734             MOV     #1600.,CNNO
006502  006337  001422                     ASL     PERCNT                  ;RESCALE PERCENT FOR 1600.
006506  006337  001422                     ASL     PERCNT                  ;POINTS PER BURST
006512  012737  000200  001412     SAR1:   MOV     #200,BITPNT             ;INITIALIZE BIT POINTER AT MSB
006520  005037  001404                     CLR     DAC                     ;INITIALIZE DAC VALUE
006524  004537  020542                     JSR     R5,$PUTS
006530  001316                             .WORD   STREG
006532  005000                     TRY:    CLR     R0
006534  063737  001412  001404             ADD     BITPNT,DAC              ;TRY BIT

 (1)                               ;*      MOV     DAC,@ADBUFF      ;/ PUT DATA FROM DAC TO DEVICE REG ADBUFF
006552  012737  000100  001406             MOV     #100,DELAY
006560  005337  001406             1$:     DEC     DELAY                   ;STALL TIME
006564  001375                             BNE     1$
006566  013701  006734                     MOV     CNNO,R1          ;SET UP FOR 1600. OR 400. CONVERSIONS
006572  113737  001362  001433             MOVB    CHANL,$TEMP2+1
006600  052737  000001  001432             BIS     #1,$TEMP2
006606  113737  001360  001431             MOVB    DUMMY,$TEMP1+1
006614  052737  000001  001430             BIS     #1,$TEMP1
006622                             NXTCVT:
006622  013777  001430  172614     $T6MP:  MOV     $TEMP1,@KMAD4
006630  112777  000006  172602             MOVB    #6,@KMAD2
006636  122777  000377  172574     10$:    CMPB    #377,@KMAD2
006644  001374                             BNE     10$
006646  013777  001432  172570             MOV     $TEMP2,@KMAD4
006654  112777  000006  172556             MOVB    #6,@KMAD2
006662  122777  000377  172550     20$:    CMPB    #377,@KMAD2
006670  001374                             BNE     20$
006672  027737  172546  001410             CMP     @KMAD4,EDGE
006700  002001                             BGE     2$
006702  005200                             INC     R0                      ;COUNT RESULTS .LT. EDGE
006704  005301                     2$:     DEC     R1
006706  001345                             BNE     NXTCVT
006710  020037  001422                     CMP     R0,PERCNT
006714  003003                             BGT     SHIFT
006716  163737  001412  001404             SUB     BITPNT,DAC              ;TAKE THE BIT OUT
006724  006237  001412             SHIFT:  ASR     BITPNT
006730  001300                             BNE     TRY
006732  000205                             RTS     R5

006734  000000                     CNNO:   .WORD   0
```

```
                                        ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
006736  104401  013122                  DIFLIN: TYPE    ,MSG20
006742  005037  001424                          CLR     OUT
006746  012700  042300                          MOV     #BUFFER,R0
006752  012701  010000                          MOV     #4096.,R1       ;4096 WORDS FOR HISTOGRAM
006756  005020                          CLEAR1: CLR     (R0)+           ;CLEAR BUFFER AREA
006760  005301                                  DEC     R1
006762  001375                                  BNE     CLEAR1
006764  012700  021622                          MOV     #DIST,R0        ;DISTRIBUTION BUFFER POINTER
006770  012701  000310                          MOV     #200.,R1        ;200. WORDS FOR DISTRIBUTION
006774  005003                                  CLR     R3
006776  005037  001424                          CLR     OUT
007002  005037  001336                          CLR     WIDE
007006  005037  001340                          CLR     NARROW
007012  005037  001342                          CLR     FIRST
007016  005037  001344                          CLR     SKIPST
007022  005020                          CLEAR2: CLR     (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
007024  005301                                  DEC     R1
007026  001375                                  BNE     CLEAR2
007030  012700  000011                  CHANNL: MOV     #11,R0          ;CHANNEL 11
007034  063700  001332                          ADD     BASECH,R0
007040  000300                                  SWAB    R0                              ;LOAD MUX BITS
007042  004537  020542                          JSR     R5,$PUTS
007046  001316                                  .WORD   STREG
007050  010037  001426                          MOV     R0,MYTEMP

                                        ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
007064  010037  001430                          MOV     R0,$TEMP1
007070  052737  000001  001430                  BIS     #1,$TEMP1
007076  012700  001440                          MOV     #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
007102  012777  001702  172344                  MOV     #RETURN,@VECTOR
007110  012701  007776                  AGAIN:  MOV     #4094.,R1
007114  004737  010776                  NEXT:   JSR     PC,RANDY        ;GET RANDOM NUMBER
007120  013702  001366                          MOV     RNA,R2
007124  042702  177760                          BIC     #177760,R2      ;MASK IT TO 4 BITS ONLY
007130  001402                                  BEQ     CONVR
007132  005302                          DELAY3: DEC     R2              ;STALL
007134  001376                                  BNE     DELAY3          ;TIME
007136                                  CONVR:
007136  013777  001430  172300          $TBF4:  MOV     $TEMP1,@KMAD4
007144  112777  000006  172266                  MOVB    #6,@KMAD2
007152  122777  000377  172260          31$:    CMPB    #377,@KMAD2
007160  001374                                  BNE     31$
007162  017702  172256                          MOV     @KMAD4,R2
007166  001413                                  BEQ     DELAY1          ;IGNORE IF =0
007170  020227  007777                          CMP     R2,#7777        ;IGNORE IF =7777
007174  001413                                  BEQ     DELAY2
007176  006302                                  ASL     R2
007200  005262  042300                          INC     BUFFER(R2)      ;MAKE HISTOGRAM
007204  100013                                  BPL     OKAY
007206  012762  077777  042300                  MOV     #077777,BUFFER(R2)      ;PREVENT OVERFLOW
007214  000407                                  BR      OKAY
007216  020227  007777                  DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
007222  001400                                  BEQ     DELAY2          ;WITH DUMMY INSTR.
007224  005201                          DELAY2: INC     R1
007226  005263  001346                          INC     TEMP(R3)
```

```
      007232  100403                           BMI     NOTOK
      007234  005301                  OKAY:    DEC     R1
      007236  001326                           BNE     NEXT
      007240  000403                           BR      AROUND
      007242  005037  001346          NOTOK:   CLR     TEMP
      007246  000772                           BR      OKAY
      007250  005300                  AROUND:  DEC     R0
      007252  001316                           BNE     AGAIN
                                      ;DATA COLLECTION HAS NOW BEEN COMPLETED - WORK ON THE DATA COLLECTED
      007254  012700  007776                   MOV     #4094.,R0
      007260  012701  042302                   MOV     #BUFFER+2,R1
      007264  012102                  READ:    MOV     (R1)+,R2        ;GET STATE WIDTH
      007266  006202                           ASR     R2              ;1 LSB = 800.
      007270  006202                           ASR     R2
      007272  006202                           ASR     R2
      007274  005502                           ADC     R2              ;1 LSB = 100.
      007276  020227  000310                   CMP     R2,#200.        ;OUT OF RANGE?
      007302  002403                           BLT     INRNGE
      007304  005237  001424                   INC     OUT             ;YES - INCREMENT COUNTER
      007310  000423                           BR      TYPBAD
      007312  006302                  INRNGE:  ASL     R2
      007314  005262  021622                   INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
      007320  006202                           ASR     R2
      007322  020227  000062                   CMP     R2,#50.         ;IS IT 1/2 LSB?
      007326  002007                           BGE     NOTNAR
      007330  005237  001340                   INC     NARROW
      007334  005702                           TST     R2              ;IS IT A SKIPPED STATE?
      007336  001002                           BNE     31$
      007340  005237  001344                   INC     SKIPST
      007344  000405                  31$:     BR      TYPBAD
      007346  020227  000226          NOTNAR:  CMP     R2,#150.        ;IS IT 1.5 LSB?
      007352  003426                           BLE     LAST
      007354  005237  001336                   INC     WIDE
      007360  005737  001342          TYPBAD:  TST     FIRST
      007364  001004                           BNE     60$
      007366  005237  001342                   INC     FIRST
      007372  104401  012301                   TYPE    ,STATE
      007376  010103                  60$:     MOV     R1,R3
      007400  162703  042302                   SUB     #BUFFER+2,R3
      007404  006203                           ASR     R3
      007406  010346                           MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
 (1)                                                                   ;;TYPE STATE
 (1)  007410  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  007412     004                           .BYTE   4               ;;TYPE 4 DIGIT(S)
 (1)  007413     001                           .BYTE   1               ;;TYPE LEADING ZEROS
      007414  104401  012275                   TYPE    ,DASH
      007420  004737  011472                   JSR     PC,DECTYP
      007424  104401  012266                   TYPE    ,LSBMSG
      007430  005300                  LAST:    DEC     R0
      007432  001314                           BNE     READ
      007434  112737  000177  014564           MOVB    #177,DECPNT
      007442  013702  001344                   MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
      007446  004737  011472                   JSR     PC,DECTYP               ;TYPE IT
      007452  104401  012524                   TYPE    ,SKPMSG         ;TYPE MESSAGE
      007456  005737  001344                   TST     SKIPST
      007462  001403                           BEQ     1$
```

```
007464  104401  012507                   TYPE     ,ERMSG          ;TYPE "ERROR"
007470  000402                           BR       NAR
007472  104401  012476           1$:     TYPE     ,OKMSG          ;TYPE #OK#
007476  013702  001340           NAR:    MOV      NARROW,R2               ;GET NO. OF NARROW STATES
007502  004737  011472                   JSR      PC,DECTYP               ;TYPE IT
007506  104401  012546                   TYPE     ,NARMSG         ;TYPE MESSAGE
007512  013702  001336                   MOV      WIDE,R2
007516  063702  001424                   ADD      OUT,R2
007522  004737  011472                   JSR      PC,DECTYP               ;TYPE NO. OF WIDE STATES
007526  104401  012605                   TYPE     ,WIDMSG         ;TYPE MESSAGE
007532  013702  001424                   MOV      OUT,R2
007536  004737  011472                   JSR      PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE 2 LSB
007542  104401  012644                   TYPE     ,OUTMSG         ;TYPE MESSAGE
007546  005737  001424                   TST      OUT
007552  001403                           BEQ      11$
007554  104401  012507                   TYPE     ,ERMSG          ,TYPE "ERROR"
007560  000402                           BR       HALF
007562  104401  012476           11$:    TYPE     ,OKMSG          ;TYPE "OK"
007566  013702  001340           HALF:   MOV      NARROW,R2
007572  063702  001336                   ADD      WIDE,R2
007576  063702  001424                   ADD      OUT,R2
007602  010200                           MOV      R2,R0
007604  004737  011472                   JSR      PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE LIMITS
007610  112737  000056  014564           MOVB     #56,DECPNT
007616  104401  012677                   TYPE     ,HAFMSG
007622  020027  000051                   CMP      R0,#41.         ;COMPARE IT TO NOMINAL
007626  003403                           BLE      21$
007630  104401  012507                   TYPE     ,ERMSG          ;TYPE "ERROR"
007634  000402                           BR       SWDIST
007636  104401  012476           21$:    TYPE     ,OKMSG          ;TYPE "OK"
007642  005737  001400           SWDIST: TST      FLAG            ;VT55?
007646  001426                           BEQ      RELACC
007650  004737  010330                   JSR      PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
007654  104401  013154                   TYPE     ,MSG16
007660  104401  013755                   TYPE     ,BUFF1          ;TYPE BUFF1-PRINT GRID
007664  012700  021622                   MOV      #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
007670  012701  000310                   MOV      #200.,R1        ;GO 200. TIMES UP TO 2 LSB
007674  012002                   NXTY1:  MOV      (R0)+,R2
007676  004737  011370                   JSR      PC,LOADY
007702  005002                           CLR      R2
007704  004737  011370                   JSR      PC,LOADY
007710  005301                           DEC      R1
007712  001370                           BNE      NXTY1
007714  104401  013700                   TYPE     ,C2             ;TYPE ASCIZ STRING
007720  004737  010330                   JSR      PC,DELCLR
```

```
                                  ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

     007724  005001               RELACC: CLR     R1                   ;RUNNING ERROR = 0
     007726  005003                       CLR     R3                   ;MAXIMUM ERROR = 0
     007730  104401  013547               TYPE    ,MSG21
     007734  012700  042302               MOV     #BUFFER+2,R0
     007740  011002               NXTSTA: MOV     (R0),R2              ;STATE WIDTH = R2
     007742  162702  001440               SUB     #800.,R2             ;STATE WIDTH ERROR IN R2
     007746  060201                       ADD     R2,R1                ;UPDATE RUNNING ERROR
     007750  010120                       MOV     R1,(R0)+             ;SAVE IN BUFFER
     007752  010104                       MOV     R1,R4                ;SAVE IN R4 ALSO
     007754  100001                       BPL     PLUS                 ;IS IT POSITIVE?
     007756  005404                       NEG     R4                   ;NO - MAKE IT POSITIVE
     007760  020403               PLUS:   CMP     R4,R3                ;CHECK AGAINST PREVIOUS MAX. ERROR
     007762  003405                       BLE     NOTNEW               ;NOT A NEW MAXIMUM
     007764  010403                       MOV     R4,R3                ;UPDATE MAXIMUM IN R3
     007766  010005                       MOV     R0,R5
     007770  162705  042302               SUB     #BUFFER+2,R5
     007774  006205                       ASR     R5                   ;R5=EDGE VALUE AT MAX. RELACC
     007776  020027  062276       NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
     010002  001356                       BNE     NXTSTA               ;NO - REPEAT
     010004  006203                       ASR     R3                   ;RESCALE FROM 1 LSB = 800. SCALING
     010006  006203                       ASR     R3                   ;TO 1 LSB = 100. SCALING
     010010  006203                       ASR     R3
     010012  005503                       ADC     R3
     010014  010302                       MOV     R3,R2
     010016  004737  011472               JSR     PC,DECTYP
     010022  104401  013574               TYPE    ,LINEA
     010026  010546                       MOV     R5,-(SP)             ;;SAVE R5 FOR TYPEOUT
 (1)                                                                   ;;TYPE VALUE
 (1) 010030  104403                       TYPOS                        ;;GO TYPE--OCTAL ASCII
 (1) 010032     004                       .BYTE   4                    ;;TYPE 4 DIGIT(S)
 (1) 010033     001                       .BYTE   1                    ;;TYPE LEADING ZEROS
     010034  104401  012433               TYPE    ,SLASH                       ;PRINT '/'
     010040  005205                       INC     R5
     010042  010546                       MOV     R5,-(SP)             ;;SAVE R5 FOR TYPEOUT
 (1)                                                                   ;;TYPE VALUE
 (1) 010044  104403                       TYPOS                        ;;GO TYPE--OCTAL ASCII
 (1) 010046     004                       .BYTE   4                    ;;TYPE 4 DIGIT(S)
 (1) 010047     001                       .BYTE   1                    ;;TYPE LEADING ZEROS
     010050  020337  011736               CMP     R3,VLIN
     010054  003403                       BLE     41$
     010056  104401  012507               TYPE    ,ERMSG
     010062  000402                       BR      42$
     010064  104401  012476       41$:    TYPE    ,OKMSG
     010070  005737  001400       42$:    TST     FLAG                 ;VT55?
     010074  001503                       BEQ     L02
     010076  012700  042300               MOV     #BUFFER,R0
     010102  012701  010000               MOV     #4096.,R1
```

```
010106  011002                  GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
010110  006202                          ASR     R2              ;RESCALE IT TO 1 LSB = 100.
010112  006202                          ASR     R2
010114  006202                          ASR     R2
010116  005502                          ADC     R2
010120  062702  000166                  ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
010124  010220                          MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
010126  005301                          DEC     R1
010130  001366                          BNE     GETDAT
010132  012700  042300                  MOV     #BUFFER,R0
010136  012704  042300                  MOV     #BUFFER,R4
010142  012705  042302                  MOV     #BUFFER+2,R5
010146  012701  001000                  MOV     #512.,R1
010152  012702  000007          NXT8:   MOV     #7.,R2
010156  012003                          MOV     (R0)+,R3
010160  010337  001414                  MOV     R3,MIN          ;MINIMUM
010164  010337  001420                  MOV     R3,MAX          ;MAXIMUM
010170  012003                  NXTCMP: MOV     (R0)+,R3
010172  020337  001414                  CMP     R3,MIN
010176  002002                          BGE     MAXTST
010200  010337  001414                  MOV     R3,MIN          ;NEW MINIMUM
010204  020337  001420          MAXTST: CMP     R3,MAX
010210  003402                          BLE     TST8
010212  010337  001420                  MOV     R3,MAX          ;NEW MAXIMUM
010216  005302                  TST8:   DEC     R2
010220  001363                          BNE     NXTCMP
010222  013724  001414                  MOV     MIN,(R4)+
010226  013725  001420                  MOV     MAX,(R5)+
010232  022425                          CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
010234  005301                          DEC     R1
010236  001345                          BNE     NXT8
010240  104401  013062                  TYPE    ,MSG18
010244  104401  014003                  TYPE    ,BUFF2          ;TYPE BUFF2
010250  012700  042300                  MOV     #BUFFER,R0
010254  004737  010306                  JSR     PC,LOAD
010260  104401  013703                  TYPE    ,C3             ;TYPE ASCIZ STRING
010264  012700  042302                  MOV     #BUFFER+2,R0
010270  004737  010306                  JSR     PC,LOAD
010274  104401  013700                  TYPE    ,C2             ;TYPE ASCIZ STRING
010300  004737  010330                  JSR     PC,DELCLR
010304  000207                  L02:    RTS     PC
010306  012701  001000          LOAD:   MOV     #512.,R1
010312  012002                  LOAD0:  MOV     (R0)+,R2
010314  005720                          TST     (R0)+
010316  004737  011370                  JSR     PC,LOADY
010322  005301                          DEC     R1
010324  001372                          BNE     LOAD0
010326  000207                          RTS     PC
```

```
010330  005000                          DELCLR: CLR     R0
010332  012701  000020                          MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
010336  005300                          1$:     DEC     R0
010340  001376                                  BNE     1$
010342  005301                                  DEC     R1
010344  001374                                  BNE     1$
010346  032777  010000  170564                  BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
010354  001401                                  BEQ     2$              ;;DON'T HALT FOR DISPLAY
010356  000000                                  HALT
010360  104401  014023                  2$:     TYPE    ,VTINIT
010364  000207                                  RTS     PC
                                        ;;NOISE SUBROUTINE;;
010366  013537  001362                  NOITST: MOV     @(R5)+,CHANL            ;LOAD CHANNEL
010372  013737  001362  001360                  MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
010400  004737  006214                          JSR     PC,GETEDG               ;GET EDGE VALUE
010404  004737  010560                          JSR     PC,NOIA                 ;GET RMS AND PEAK VALUES
010410  012737  000001  006436                  MOV     #1,EDGFLG
010416  004737  010424                          JSR     PC,TYPRP                ;TYPE RMS AND PEAK VALUES
010422  000205                                  RTS     R5





                                        ;;TYPE RMS AND PEAK VALUES;;
010424  104401  012373                  TYPRP:  TYPE    ,NOI
010430  005737  001374                          TST     RMS
010434  100002                                  BPL     POSRMS
010436  005037  001374                          CLR     RMS                     ;RMS<0,SET RMS=0
010442  005737  001376                  POSRMS: TST     PEAK
010446  100002                                  BPL     POSPEA
010450  005037  001376                          CLR     PEAK                    ;PEAK<0,SET PEAK=0
010454  013702  001374                  POSPEA: MOV     RMS,R2
010460  004737  011472                          JSR     PC,DECTYP
010464  104401  012746                          TYPE    ,MESR
010470  013702  001376                          MOV     PEAK,R2
010474  004737  011472                          JSR     PC,DECTYP
010500  104401  012761                          TYPE    ,MESP
010504  004737  006374                          JSR     PC,TYPEDG
010510  104401  012403                          TYPE    ,CHAN
010514  013746  001362                          MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
                                                                        ;;TYPE CHANL
010520  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
010522     002                                  .BYTE   2               ;;TYPE 2 DIGIT(S)
010523     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
010524  023737  001374  011730                  CMP     RMS,VNR         ;WITHIN LIMITS?
010532  003007                                  BGT     ER
010534  023737  001376  011732                  CMP     PEAK,VNP                ;WITHIN LIMITS?
010542  003003                                  BGT     ER
010544  104401  012476                          TYPE    ,OKMSG
010550  000207                                  RTS     PC
010552  104401  012507                   ER:    TYPE    ,ERMSG
010556  000207                                  RTS     PC
```

```
                                ;;SUBROUTINES FOR NOISE TEST;;
010560  005037  001374          NOIA:   CLR     RMS                     ;CLEAR RMS VLAUE
010564  005037  001376                  CLR     PEAK                    ;CLEAR PEAK VALUE
010570  004537  006440          NOI1:   JSR     R5,SARSUB               ;DO SAR ROUTINE AT 16%
010574  000020                          16.
010576  063737  001404  001374          ADD     DAC,RMS         ;ADD RESULT TO RMS
010604  004537  006440                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 84%
010610  000124                          84.
010612  163737  001404  001374          SUB     DAC,RMS         ;SUBTRACT RESULT FROM RMS
010620  004537  006440                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 1%
010624  000001                          1
010626  063737  001404  001376          ADD     DAC,PEAK                ;ADD RESULT TO PEAK
010634  004537  006440                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 99%
010640  000143                          99.
010642  163737  001404  001376          SUB     DAC,PEAK                ;SUBTRACT RESULT FROM PEAK
010650  000207                          RTS     PC                      ;RETURN

010652  012537  001362          NOI8:   MOV     (R5)+,CHANL             ;GET CHANNEL VALUE
010656  063737  001332  001362          ADD     BASECH,CHANL
010664  013737  001362  001360          MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
010672  004737  006214                  JSR     PC,GETEDG               ;GET EDGE VALUES
010676  005037  001374                  CLR     RMS             ;CLEAR RMS VALUE
010702  005037  001376                  CLR     PEAK            ;CLEAR PEAK VALUE
010706  012737  000010  010774          MOV     #10,10$         ;SET UP COUNTER
010714  004737  010570          1$:     JSR     PC,NOI1         ;GET NOISE VALUES
010720  005237  001410                  INC     EDGE
010724  005337  010774                  DEC     10$
010730  001371                          BNE     1$              ;REPEAT 8 TIMES
010732  162737  000010  001410          SUB     #10,EDGE
010740  006237  001374                  ASR     RMS             ;SCALE IT TO 1 LSB=100.
010744  005537  001374                  ADC     RMS
010750  006237  001376                  ASR     PEAK
010754  005537  001376                  ADC     PEAK
010760  012737  000010  006436          MOV     #8.,EDGFLG
010766  004737  010424                  JSR     PC,TYPRP                ;TYPE RESULTS
010772  000205                          RTS     R5              ;RETURN
010774  000000                  10$:    0                       ;COUNTER



                                ;;RANDOM NUMBER GENERATOR;;
010776  063737  001370  001366  RANDY:  ADD     RNB,RNA
011004  063737  001372  001366          ADD     RNC,RNA
011012  005537  001366                  ADC     RNA
011016  063737  001366  001370          ADD     RNA,RNB
011024  063737  001372  001370          ADD     RNC,RNB
011032  005537  001370                  ADC     RNB
011036  063737  001366  001372          ADD     RNA,RNC
011044  063737  001370  001372          ADD     RNB,RNC
011052  005537  001372                  ADC     RNC
011056  000207                          RTS     PC
```

```
                                             ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
     011060  012500                          CONVRT: MOV     (R5)+,R0                  ;GET CHANNEL VALUE
     011062  063700  001332                          ADD     BASECH,R0
     011066  010037  001362                          MOV     R0,CHANL
     011072  000300                                  SWAB    R0
     011074  005037  001346                          CLR     TEMP

 (1)                                         ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
     011110  010037  001426                          MOV     R0,MYTEMP
 (2)
 (2)                                         ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
     011124  012700  010000                          MOV     #10000,R0
     011130  005300                          2$:     DEC     R0
     011132  001376                                  BNE     2$
     011134  012777  001702  170312                  MOV     #RETURN,@VECTOR           ;LOAD VECTOR
     011142  012700  000010                          MOV     #10,R0          ;SET UP COUNTER
     011146                                  1$:
 (1)
 (1)                                         ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
     011156  052737  000001  001426                  BIS     #1,MYTEMP

 (1)                                         ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
     011174  005001                                  CLR     R1
     011176  105201                          10$:    INCB    R1
     011200  001007                                  BNE     11$
     011202  012737  000200  001124                  MOV     #BIT7,$GDDAT              ;EXPECT DONE TO SET BY NOW
     011210  013737  001426  001126                  MOV     MYTEMP,$BDDAT

     011216  104001                                  ERROR   1                         ;DONE FAILED TO SET ON A/D

     011220                                  11$:

 (2)                                         ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1) 011230  105737  001426                          TSTB    MYTEMP
     011234  100360                                  BPL     10$

 (1)                                         ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
     011246  063737  001426  001346                  ADD     MYTEMP,TEMP
                                                                             ;WAIT FOR CONVERSION
                                                                             ;READ BUFFER
     011254  005300                                  DEC     R0
     011256  001333                                  BNE     1$              ;DO 8 TIMES
     011260  006237  001346                          ASR     TEMP            ;AVERAGE VALUE
     011264  006237  001346                          ASR     TEMP
     011270  006237  001346                          ASR     TEMP
     011274  005537  001346                          ADC     TEMP
     011300  000205                                  RTS     R5              ;RETURN
```

```
                                        ;COMPARE $GDDAT AND $BDDAT;;
011302  012537  001124                  COMPAR: MOV     (R5)+,$GDDAT            ;GET GOOD DATA
011306  013537  001402                          MOV     @(R5)+,SPREAD           ;GET SPREAD
011312  013737  001346  001126                  MOV     TEMP,$BDDAT             ;GET BAD(ACTUAL) DATA
011320  013701  001126                          MOV     $BDDAT,R1
011324  013700  001124                          MOV     $GDDAT,R0
011330  160100                                  SUB     R1,R0           ;GET DIFFERENCE
011332  100001                                  BPL     7$
011334  005400                                  NEG     R0
011336  020037  001402                  7$:     CMP     R0,SPREAD               ;COMPARE IT TO SPREAD
011342  003001                                  BGT     10$             ;GO TO ERROR PRINTOUT
011344  005725                                  TST     (R5)+           ;BUMP RETURN POINTER AROUND ERROR CALL
011346  000205                          10$:    RTS     R5

                                        ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011350  004737  020224                  RST:    JSR     PC,$RESET
011354  052777  000100  167562                  BIS     #100,@$TKS
011362  005037  177776                          CLR     PSW
011366  000207                                  RTS     PC



                                        ;SUBROUTINE LOADY;
011370  005702                          LOADY:  TST     R2              ;ROUTINE TO LOAD VLAUE INTO R2
011372  100001                                  BPL     PLUSR2          ;AS A VT55 Y-VALUE
011374  005002                                  CLR     R2
011376  020227  000353                  PLUSR2: CMP     R2,#235.
011402  002402                                  BLT     LESS
011404  012702  000353                          MOV     #235.,R2
011410  010203                          LESS:   MOV     R2,R3
011412  042702  177740                          BIC     #177740,R2
011416  052702  000040                          BIS     #40,R2
011422  105777  167522                  B10:    TSTB    @$TPS           ;PRINT CHARACTER
011426  100375                                  BPL     B10
011430  110277  167516                          MOVB    R2,@$TPB
011434  006203                                  ASR     R3
011436  006203                                  ASR     R3
011440  006203                                  ASR     R3
011442  006203                                  ASR     R3
011444  006203                                  ASR     R3
011446  042703  177770                          BIC     #177770,R3
011452  052703  000040                          BIS     #40,R3
011456  105777  167466                  B11:    TSTB    @$TPS           ;PRINT CHARACTER
011462  100375                                  BPL     B11
011464  110377  167462                          MOVB    R3,@$TPB
011470  000207                                  RTS     PC


```

```
                                        ;;SUBROUTINE TO TYPE DECIMAL VALUE;.
                                        ;;IN R2 AS X.XX;;
011472  005702                          DECTYP: TST     R2              ;TEST VALUE TO BE TYPED
011474  100003                                  BPL     POS
011476  104401  012235                          TYPE    ,MINUS                  ;TYPE MINUS SIGN
011502  005402                                  NEG     R2
011504  020227  001747                  POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
011510  003402                                  BLE     OKAYD
011512  012702  001747                          MOV     #999.,R2
011516  105037  014566                  OKAYD:  CLRB    ONES            ;CLEAR ONES
011522  105037  014565                          CLRB    TENS            ;CLEAR TENS
011526  105037  014563                          CLRB    HUNS            ;CLEAR HUNS
011532  005702                          TESTR2: TST     R2                      ;CONVERT VALUE TO A DECIMAL VALUE
011534  001424                                  BEQ     TYPOUT
011536  005302                                  DEC     R2
011540  105237  014566                          INCB    ONES
011544  123727  014566  000012                  CMPB    ONES,#10.
011552  001367                                  BNE     TESTR2
011554  105037  014566                          CLRB    ONES
011560  105237  014565                          INCB    TENS
011564  123727  014565  000012                  CMPB    TENS,#10.
011572  001357                                  BNE     TESTR2
011574  105037  014565                          CLRB    TENS
011600  105237  014563                          INCB    HUNS
011604  000752                                  BR      TESTR2
011606  152737  000060  014563          TYPOUT: BISB    #60,HUNS                ;PREPARE FOR TYPOUT
011614  152737  000060  014565                  BISB    #60,TENS
011622  152737  000060  014566                  BISB    #60,ONES
011630  104401  014563                          TYPE    ,HUNS           ;TYPE VALUE
011634  000207                                  RTS     PC

011636  012701  011730                  WFADJ:  MOV     #VNR,R1                 ;SUBROUTINE TO SET UP LIMITS
011642  005737  001332                          TST     BASECH                  ;TESTING AN AM11K?
011646  001403                                  BEQ     1$              ;;
011650  012702  011762                          MOV     #VARLT3,R2              ;BASECH NOT ZERO, USE AM11K LIMITS
011654  000410                                  BR      3$              ;;
011656  005737  001416                  1$:     TST     WFTEST
011662  001003                                  BNE     2$
011664  012702  011742                          MOV     #VARLT1,R2              ;WFTEST=0,USE NORMAL LIMITS
011670  000402                                  BR      3$
011672  012702  011752                  2$:     MOV     #VARLT2,R2              ;WFTEST=1,USE OPTION AREA LIMITS
011676  012221                          3$:     MOV     (R2)+,(R1)+
011700  005711                                  TST     (R1)
011702  100375                                  BPL     3$
011704  000207                                  RTS     PC
```

```
2473 011706  000001                         V1:     1                              ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
2474 011710  000002                         V2:     2
2475 011712  000010                         V10:    10
2476 011714  000050                         V50:    50
2477 011716  000144                         V144:   144
2478 011720  000115                         V115:   115
2479 011722  000240                         V240:   240
2480 011724  000005                         V5:     5
2481 011726  000062                         V50D:   50.
2482
2483 011730  000000                         VNR:    0                              ;RMS NOISE LIMIT
2484 011732  000000                         VNP:    0                              ;PEAK NOISE LIMIT
2485 011734  000000                         VSET:   0                              ;INTER-CHANNEL SETTLING LIMIT
2486 011736  000000                         VLIN:   0                              ;RELATIVE ACCURACY ERROR LIMIT
2487 011740  100000                                 BIT15
2488
2489 011742  000036                         VARLT1: 36                   ;.3 LSB,NORMAL LIMITS FOR SYSTEM
2490 011744  000310                                 200.                 ;2. LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
2491 011746  000226                                 226                  ;1.5 LSB
2492 011750  000156                                 156                  ;1.1 LSB
2493
2494 011752  000036                         VARLT2: 36                   ;.25 LSB,  TIGHTER LIMITS FOR OPTION
2495 011754  000226                                 150.                 ;1.5 LSB,  AREA USE ON SPEC TESTS
2496 011756  000206                                 206                  ;1.35 LSB
2497 011760  000151                                 151                  ;1.05 LSB
2498
2499 011762  000062                         VARLT3: 50.                  ;.5 LSB,  LIMITS FOR AM11K TESTING
2500 011764  000310                                 200.                 ;2. LSB
2501 011766  000310                                 200.                 ;2. LSB
2502 011770  000226                                 150.                 ;1.5 LSB
2503
2504 011772  052777  000100  167144         AGATST: BIS     #100,@$TKS
2505 012000  000177  000000                         JMP     @AGTST
2506 012004  001712                         AGTST:  BEGIN
```

```
 2508                                  .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                                  ;;*****************************************************************
  (1)                                  ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                  ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
  (1)                                  ;*IF THERES A MONITOR GO TO IT
  (1)                                  ;*IF THERE ISN'T JUMP TO AGATST
  (1)
  (1)  012006                          $EOP:
  (2)  012006  000240                          NOP
  (1)  012010  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  012014  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  012020  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  012024  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  012032  005327                          DEC     (PC)+           ;;LOOP?
  (1)  012034  000001                  $EOPCT: .WORD   1
  (1)  012036  003022                          BGT     $DOAGN          ;;YES
  (1)  012040  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  012042  000001                  $ENDCT: .WORD   1
  (1)  012044  012034                          $EOPCT
  (1)  012046  104401  012113                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
  (2)  012052  013746  001202                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
  (2)  012056  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
  (1)  012060  104401  012110                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
  (1)  012064  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (1)  012070  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  012072  000005                          RESET                   ;;CLEAR THE WORLD
  (1)  012074  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  012076  000240                          NOP                     ;;SAVE ROOM
  (1)  012100  000240                          NOP                     ;;FOR
  (1)  012102  000240                          NOP                     ;;ACT11
  (1)  012104                          $DOAGN:
  (1)  012104  000137                          JMP     @(PC)+          ;;RETURN
  (1)  012106  011772                  $RTNAD: .WORD   AGATST
  (1)  012110     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)  012113     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
  (1)  012120  050040  051501  020123
  (1)  012126  000043
 2509
```

```
                                         .SBTTL         ASCII MESSAGES
012130  005015  047516  051511   NOIMSG: .ASCIZ  <15><12>/NOISE TEST-- /
012136  020105  042524  052123
012144  026455  000040
012150  005015  042523  052124   SETMSG: .ASCIZ  <15><12>/SETTLING TEST-- TYPE DESIRED 'FROM' CHANNEL & CR: /
012156  044514  043516  052040
012164  051505  026524  020055
012172  054524  042520  042040
012200  051505  051111  042105
012206  023440  051106  046517
012214  020047  044103  047101
012222  042516  020114  020046
012230  051103  020072     000
012235     055     000           MINUS:  .BYTE   55,0
012237     077     000           QUEST:  .BYTE   77,0
012241     136     101     040   AMSG:   .BYTE   136,101,40,40,0
012244     040     000
012246     136     103     040   CMSG:   .BYTE   136,103,40,40,0
012251     040     000
012253     136     107     015   GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012256     012     123     127
012261     122     105     107
012264     072     000
012266  046040  041123  005015   LSBMSG: .ASCIZ  / LSB/<15><12>
012274     000
012275     055  020055     000   DASH:   .ASCIZ  /-- /
012301     123  040524  042524   STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012306  026455  053440  042111
012314  044124  005015     000
012321     103  000110           CH:     .ASCIZ  /CH/
012324  020040  020040     000   SPACE:  .ASCIZ  /    /
012331     040  051514  020102   LSB:    .ASCIZ  / LSB ON CH/
012336  047117  041440  000110
012344  051440  052105  046124   SETCH:  .ASCIZ  / SETTLING FROM CH/
012352  047111  020107  051106
012360  046517  041440  000110
012366  040440  020124     000   ATMSG:  .ASCIZ  / AT /
012373     116  044517  042523   NOI:    .ASCIZ  /NOISE: /
012400  020072     000
012403     040  047117  041440   CHAN:   .ASCIZ  / ON CHANNEL /
012410  040510  047116  046105
012416  000040
012420  020040  020040  047504   DONE:   .ASCIZ  /    DONE/<15><12>
012426  042516  005015     000
012433     057     000           SLASH:  .ASCIZ  #/#
012435     124  050131  020105   TOMSG:  .ASCIZ  /TYPE DESIRED 'TO' CHANNEL & CR: /
012442  042504  044523  042522
012450  020104  052047  023517
012456  041440  040510  047116
012464  046105  023040  041440
012472  035122  000040
012476  020040  020040  045517   OKMSG:  .ASCIZ  /    OK/<15><12>
012504  005015     000
```

```
 2534  012507      040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
       012514   047522  025122  006452
       012522   000012
 2535  012524   051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
       012532   042105  051440  040524
       012540   042524  051450  000051
 2536  012546   047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
       012554   020127  036050  030440
       012562   031057  046040  041123
       012570   020051  052123  052101
       012576   024105  024523  005015
       012604      000
 2537  012605      040  044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
       012612   024040  020076  020061
       012620   027461  020062  051514
       012626   024502  051440  040524
       012634   042524  051450  006451
       012642   000012
 2538  012644   051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
       012652   051450  020051  044527
       012660   042504  020122  044124
       012666   047101  031040  046040
       012674   041123     000
 2539  012677      040  052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
       012704   026505  044527  052104
       012712   024110  024523  047440
       012720   052125  044523  042504
       012726   025440  047440  020122
       012734   020055  027461  020062
       012742   051514  000102
 2540  012746   046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
       012754   051515  020054     000
 2541  012761      040  051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
       012766   042520  045501  040440
       012774   020124     000
 2542  012777      015  042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
       013004   047440  020106  047514
       013012   044507  020103  042524
       013020   052123     123
 2543  013023      040  047117  040440  ONAD:   .ASCIZ  / ON AD11K AT /
       013030   030504  045461  040440
       013036   020124     000
 2544  013041      040  042101  030461  MSG50:  .ASCIZ  / AD11K'S FOUND/<15><12>
       013046   023513  020123  047506
       013054   047125  006504  000012
 2545  013062   005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
       013070   020062  051514  006502
       013076   005012  005012  005012
       013104   005012  005012  005012
 2546  013112   030455  031057  051514          .ASCIZ  \-1/2LSB\
       013120   000102
 2547
 2549                                               .EVEN
 2550  013122  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
       013130  047105  044524  046101
       013136  046040  047111  040505
       013144  044522  054524  006472
       013152  000012
 2551  013154  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
       013162  020040  020040  020040
       013170  020040  020040  020040
       013176  020040  052123  052101
       013204  026505  044527  052104
       013212  020110  044504  052123
       013220  044522  052502  044524
       013226  047117  005015  005012
 2552  013234  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><12><
       013242  051440  040524  042524
       013250  005123  005012  005012
       013256  005012  005012  005012
       013264  005012  005012  005012
       013272  005012
 2553  013274  020040  020040  020040          .ASCII  /                                                  STATE WIDTH (LSB)/<15>
       013302  020040  020040  020040
       013310  020040  020040  020040
       013316  020040  020040  020040
       013324  020040  020040  020040
       013332  020040  020040  020040
       013340  020040  020040  020040
       013346  020040  020040  020040
       013354  051440  040524  042524
       013362  053440  042111  044124
       013370  024040  051514  024502
       013376  005015
 2554  013400  030040  020040  020040          .ASCIZ  # 0                1/2               1               1 1/2               2#
       013406  020040  020040  020040
       013414  020040  020040  027461
       013422  020062  020040  020040
       013430  020040  020040  020040
       013436  020040  020061  020040
       013444  020040  020040  020040
       013452  020040  030440  030440
       013460  031057  020040  020040
       013466  020040  020040  020040
       013474  020040  031040     000
 2555  013501     015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR DESIRED TEST: /
       013506  020105  042514  052124
       013514  051105  023040  041440
       013522  020122  047506  020122
       013530  042504  044523  042522
       013536  020104  042524  052123
       013544  020072     000
 2556  013547     122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
       013554  053111  020105  041501
       013562  052503  040522  054503
       013570  006472  000012
 2557  013574  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
       013602  054101  046511  046525
```

```
        013610  040440  020124     000
  2558  013615     015  041412  046101  HEAD5:  .ASCII  <15><12>/CALIBRATION--/
        013622  041111  040522  044524
        013630  047117  026455
  2559  013634  051440  052105  041440  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
        013642  040510  047116  046105
        013650  044440  020116  053523
        013656  020122  047514  020127
        013664  054502  042524  005015
        013672     000
  2560  013673     033  000132          C0:     .ASCIZ  <33><132>
  2561  013676  000055                  C1:     .ASCIZ  <55>
  2562  013700  031033     000          C2:     .ASCIZ  <33><62>
  2563  013703     112     000          C3:     .ASCIZ  <112>
  2564  013705     015  047412  043106  MOFSET: .ASCIZ  <15><12>/OFFSET =/
        013712  042523  020124  000075
  2565  013720  046040  041123  000040  MLSB:   .ASCIZ  / LSB /
  2566  013726  040440  020124     000  MAT:    .ASCIZ  / AT /
  2567  013733     015  020012  047105  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
        013740  042524  044522  043516
        013746  052040  051505  020124
        013754     000
  2568  013755     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
        013760     061     111     062
        013763     114     041     060
        013766     045     063     051
        013771     066     055     071
        013774     061     074     110
        013777     041     040     112
        014002     000
  2569  014003     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
        014006     047     111     061
        014011     104     050     065
        014014     044     062     110
        014017     040     040     102
        014022     000
  2570  014023     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0
        014026     112     033     061
        014031     101     040     033
        014034     062     000
  2571  014036  005015  041412  046122  HEAD1:  .ASCII  <15><12><12>#CRLPKC0    LPA/AD11-K DIAGNOSTIC#<15><12>
        014044  045520  030103  020040
        014052  020040  050114  027501
        014060  042101  030461  045455
        014066  042040  040511  047107
        014074  051517  044524  006503
        014102     012
  2572  014103     012  035101  040440          .ASCII  <12>/A: AUTO TEST/
        014110  052125  020117  042524
        014116  052123
  2573  014120  005015  035103  041440          .ASCII  <15><12>/C: CALIBRATION/
        014126  046101  041111  040522
        014134  044524  047117
  2574  014140  005015  035114  046040          .ASCII  <15><12>/L: LOGIC TEST/
        014146  043517  041511  052040
        014154  051505     124
```

```
 2575 014157    015  047012  020072          .ASCII  <15><12>/N: NOISE TEST/
      014164 047516  051511  020105
      014172 042524  052123
 2576 014176 005015  035123  051440          .ASCII  <15><12>/S: SETTLE TEST/
      014204 052105  046124  020105
      014212 042524  052123
 2577 014216 005015  035127  053440          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
      014224 040522  040520  047522
      014232 047125  020104  042524
      014240 052123  005015     000
 2578 014245    015  051412  040524  EM1:    .ASCIZ  <15><12>/STATUS REG. ERROR/<15><12>
      014252 052524  020123  042522
      014260 027107  042440  051122
      014266 051117  005015     000
 2579 014273    015  043012  044501  EM2:    .ASCIZ  <15><12>/FAILED TO INTERRUPT/<15><12>
      014300 042514  020104  047524
      014306 044440  052116  051105
      014314 052522  052120  005015
      014322    000
 2580 014323    015  052412  042516  EM3:    .ASCIZ  <15><12>/UNEXPECTED INTERRUPT/<15><12>
      014330 050130  041505  042524
      014336 020104  047111  042524
      014344 051122  050125  006524
      014352 000012
 2581 014354 005015  051105  047522  FM4:    .ASCIZ  <15><12>#ERROR ON A/D CHANNEL#<15><12>
      014362 020122  047117  040440
      014370 042057  041440  040510
      014376 047116  046105  005015
      014404    000
 2582 014405    105  051122  041520  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/<15><12>
      014412 051440  051124  043505
      014420 042440  050130  041505
      014426 042524  020104  041501
      014434 052524  046101  005015
      014442    000
 2583 014443    105  051122  041520  DH2:    .ASCIZ  /ERR^C  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
      014450 020040  052123  042522
      014456 020107  020040  044103
      014464 047101  042516  020114
      014472 047040  046517  047111
      014500 046101  020040  047524
      014506 042514  040522  041516
      014514 020105  040440  052103
      014522 040525  000114
 2584 014526 051105  050122  020103  DH3:    .ASCIZ  /ERRPC      STREG    ACTUAL/<15><12>
      014534 020040  020040  051440
      014542 051124  043505  020040
      014550 020040  041501  052524
      014556 046101  005015     000
```

```
2586 014563     000                     HUNS:   .BYTE   0
2587 014564     056                     DECPNT: .BYTE   56
2588 014565     000                     TENS:   .BYTE   0
2589 014566     000       000           ONES:   .BYTE   0,0
2590                                    .EVEN
2591
2592 014570  001116  001316  001124  DT1:    $ERRPC, STREG, $GDDAT, $BDDAT,0
     014576  001126  000000
2593 014602  001116  001316  001362  DT2:    $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
     014610  001124  001402  001126
     014616  000000
2594 014620  001116  001316  001126  DT3:    $ERRPC,STREG,$BDDAT,0
     014626  000000
2595
2596 014630  000000                  DF1:    0
2597
2598
```

```
 2600                                         .SBTTL  TTY INPUT ROUTINE
  (1)
  (2)                                         ;;*************************************************************
  (1)                                         .ENABL  LSB
  (1)
  (1)                                         .DSABL  LSB
  (1)
  (1)
  (2)                                         ;;*************************************************************
  (1)                                         ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
  (1)                                         ;*CALL:
  (1)                                         ;*        RDCHR                        ;;INPUT A SINGLE CHARACTER FROM THE TTY
  (1)                                         ;*        RETURN HERE                  ;;CHARACTER IS ON THE STACK
  (1)                                         ;*                                     ;;WITH PARITY BIT STRIPPED OFF
  (1)                                         ;
  (1)
  (1) 014632 011646                           $RDCHR: MOV     (SP),-(SP)           ;;PUSH DOWN THE PC
  (1) 014634 016666 000004 000002                     MOV     4(SP),2(SP)          ;;SAVE THE PS
  (1) 014642 105777 164276                    1$:     TSTB    @$TKS                ;;WAIT FOR
  (1) 014646 100375                                   BPL     1$                   ;;A CHARACTER
  (1) 014650 117766 164272 000004                     MOVB    @$TKB,4(SP)          ;;READ THE TTY
  (1) 014656 042766 177600 000004                     BIC     #^C<177>,4(SP)       ;;GET RID OF JUNK IF ANY
  (1) 014664 026627 000004 000023                     CMP     4(SP),#23            ;;IS IT A CONTROL-S?
  (1) 014672 001013                                   BNE     3$                   ;;BRANCH IF NO
  (1) 014674 105777 164244                    2$:     TSTB    @$TKS                ;;WAIT FOR A CHARACTER
  (1) 014700 100375                                   BPL     2$                   ;;LOOP UNTIL ITS THERE
  (1) 014702 117746 164240                            MOVB    @$TKB,-(SP)          ;;GET CHARACTER
  (1) 014706 042716 177600                            BIC     #^C177,(SP)          ;;MAKE IT 7-BIT ASCII
  (1) 014712 022627 000021                            CMP     (SP)+,#21            ;;IS IT A CONTROL-Q?
  (1) 014716 001366                                   BNE     2$                   ;;IF NOT DISCARD IT
  (1) 014720 000750                                   BR      1$                   ;;YES, RESUME
  (1) 014722 026627 000004 000140             3$:     CMP     4(SP),#140           ;;IS IT UPPER CASE?
  (1) 014730 002407                                   BLT     4$                   ;;BRANCH IF YES
  (1) 014732 026627 000004 000175                     CMP     4(SP),#175           ;;IS IT A SPECIAL CHAR?
  (1) 014740 003003                                   BGT     4$                   ;;BRANCH IF YES
  (1) 014742 042766 000040 000004                     BIC     #40,4(SP)            ;;MAKE IT UPPER CASE
  (1) 014750 000002                           4$:     RTI                          ;;GO BACK TO USER
  (2)                                         ;;*************************************************************
  (1)                                         ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
  (1)                                         ;*CALL:
  (1)                                         ;*        RDLIN                        ;;INPUT A STRING FROM THE TTY
  (1)                                         ;*        RETURN HERE                  ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
  (1)                                         ;*                                     ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
  (1)
  (1) 014752 010346                           $RDLIN: MOV     R3,-(SP)             ;;SAVE R3
  (1) 014754 012703 015060                    1$:     MOV     #$TTYIN,R3           ;;GET ADDRESS
  (1) 014760 022703 015070                    2$:     CMP     #$TTYIN+8.,R3        ;;BUFFER FULL?
  (1) 014764 101405                                   BLOS    4$                   ;;BR IF YES
  (1) 014766 104406                                   RDCHR                        ;;GO READ ONE CHARACTER FROM THE TTY
  (1) 014770 112613                                   MOVB    (SP)+,(R3)           ;;GET CHARACTER
  (1) 014772 122713 000177                    10$:    CMPB    #177,(R3)            ;;IS IT A RUBOUT
  (1) 014776 001003                                   BNE     3$                   ;;SKIP IF NOT
  (1) 015000 104401 001170                    4$:     TYPE    ,$QUES               ;;TYPE A '?'
  (1) 015004 000763                                   BR      1$                   ;;CLEAR THE BUFFER AND LOOP
  (1) 015006 111337 015056                    3$:     MOVB    (R3),9$              ;;ECHO THE CHARACTER
  (1) 015012 104401 015056                            TYPE    ,9$
  (1)  015016  122723  000015                    CMPB    #15,(R3)+         ;;CHECK FOR RETURN
  (1)  015022  001356                            BNE     2$                ;;LOOP IF NOT RETURN
  (1)  015024  105063  177777                    CLRB    -1(R3)            ;;CLEAR RETURN (THE 15)
  (1)  015030  104401  001172                    TYPE    ,$LF              ;;TYPE A LINE FEED
  (1)  015034  012603                            MOV     (SP)+,R3          ;;RESTORE R3
  (1)  015036  011646                            MOV     (SP),-(SP)        ;;ADJUST THE STACK AND PUT ADDRESS OF THE
  (1)  015040  016666  000004  000002            MOV     4(SP),2(SP)       ;;      FIRST ASCII CHARACTER ON IT
  (1)  015046  012766  015060  000004            MOV     #$TTYIN,4(SP)
  (1)  015054  000002                            RTI                       ;;RETURN
  (1)  015056     000                   9$:      .BYTE   0                 ;;STORAGE FOR ASCII CHAR. TO TYPE
  (1)  015057     000                            .BYTE   0                 ;;TERMINATOR
  (1)  015060  000010                   $TTYIN:  .BLKB   8.                ;;RESERVE 8 BYTES FOR TTY INPUT
  (1)  015070  052536  005015     000   $CNTLU:  .ASCIZ  /^U/<15><12>      ;;CONTROL "U"
  (1)  015075     136  006507  000012   $CNTLG:  .ASCIZ  /^G/<15><12>      ;;CONTROL "G"
  (1)  015102  005015  053523  020122   $MSWR:   .ASCIZ  <15><12>/SWR = /
  (1)  015110  020075     000
  (1)  015113     040  047040  053505   $MNEW:   .ASCIZ  /  NEW = /
  (1)  015120  036440  000040
```

```
 2602                                   .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
  (1)
  (2)                                   ;;*************************************************************
  (1)                                   ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
  (1)                                   ;*CHANGE IT TO BINARY.
  (1)                                   ;*CALL:
  (1)                                   ;*      RDOCT                   ;;READ AN OCTAL NUMBER
  (1)                                   ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
  (1)                                   .*                              ;;HIGH ORDER BITS ARE IN $HIOCT
  (1)
  (1)  015124  011646                   $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
  (1)  015126  016666  000004  000002           MOV     4(SP),2(SP)     ;;INPUT NUMBER
  (3)  015134  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  015136  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  015140  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (1)  015142  104407                   1$:     RDLIN                   ;;READ AN ASCIZ LINE
  (1)  015144  012600                           MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
  (1)  015146  005001                           CLR     R1              ;;CLEAR DATA WORD
  (1)  015150  005002                           CLR     R2
  (1)  015152  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
  (1)  015154  001412                           BEQ     3$              ;;IF ZERO GET OUT
  (1)  015156  006301                           ASL     R1              ;;*2
  (1)  015160  006102                           ROL     R2
  (1)  015162  006301                           ASL     R1              ;;*4
  (1)  015164  006102                           ROL     R2
  (1)  015166  006301                           ASL     R1              ;;*8
  (1)  015170  006102                           ROL     R2
  (1)  015172  042716  177770                   BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
  (1)  015176  062601                           ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
  (1)  015200  000764                           BR      2$              ;;LOOP
  (1)  015202  005726                   3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
  (1)  015204  010166  000012                   MOV     R1,12(SP)       ;;SAVE THE RESULT
  (1)  015210  010237  015224                   MOV     R2,$HIOCT
  (3)  015214  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  015216  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  015220  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  015222  000002                           RTI                     ;;RETURN
  (1)  015224  000000                   $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 2604                                   .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                                   ;;***************************************************************
  (1)                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
  (1)                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
  (1)                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                   ;*SW14=1        LOOP ON TEST
  (1)                                   ;*SW11=1        INHIBIT ITERATIONS
  (1)                                   ;*SW09=1        LOOP ON ERROR
  (1)                                   ;*SW08=1        LOOP ON TEST IN SWR<7:0>
  (1)                                   ;*CALL
  (1)                                   ;*      SCOPE           ;;SCOPE=IOT
  (1)
  (1)  015226                           $SCOPE:
  (1)  015226  032777  040000  163704   1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
  (1)  015234  001114                           BNE     $OVER           ;;YES IF SW14=1
  (1)                                   ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  015236  000416                   $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
  (1)                                                                   ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
  (1)  015240  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  015244  012737  015264  000004           MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
  (1)  015252  005737  177060                   TST     @#177060        ;;TIME OUT ON XOR?
  (1)  015256  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  015262  000463                           BR      $SVLAD          ;;GO TO THE NEXT TEST
  (1)  015264  022626                   5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  015266  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  015272  000423                           BR      7$              ;;LOOP ON THE PRESENT TEST
  (1)  015274                           6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  015274  032777  000400  163636           BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
  (1)  015302  001404                           BEQ     2$              ;;BR IF NO
  (1)  015304  127737  163630  001102           CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
  (1)  015312  001465                           BEQ     $OVER           ;;BR IF YES
  (1)  015314  105737  001103           2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
  (1)  015320  001421                           BEQ     3$              ;;BR IF NO
  (1)  015322  123737  001115  001103           CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  015330  101015                           BHI     3$              ;;BR IF NO
  (1)  015332  032777  001000  163600           BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
  (1)  015340  001404                           BEQ     4$              ;;BR IF NO
  (1)  015342  013737  001110  001106   7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
  (1)  015350  000446                           BR      $OVER
  (1)  015352  105037  001103           4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
  (1)  015356  005037  001160                   CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  (1)  015362  000415                           BR      1$              ;;ESCAPE TO THE NEXT TEST
  (1)  015364  032777  004000  163546   3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
  (1)  015372  001011                           BNE     1$              ;;BR IF YES
  (1)  015374  005737  001202                   TST     $PASS           ;;IF FIRST PASS OF PROGRAM
  (1)  015400  001406                           BEQ     1$              ;;      INHIBIT ITERATIONS
  (1)  015402  005237  001104                   INC     $ICNT           ;;INCREMENT ITERATION COUNT
  (1)  015406  023737  001160  001104           CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
  (1)  015414  002024                           BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
  (1)  015416  012737  000001  001104   1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
  (1)  015424  013737  015502  001160           MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
  (1)  015432  105237  001102           $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
  (1)  015436  113737  001102  001200           MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
  (1)  015444  011637  001106                   MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
    (1)  015450  011637  001110                   MOV     (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
    (1)  015454  005037  001162                   CLR     $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
    (1)  015460  112737  000001  001115           MOVB    #1,$ERMAX        ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
    (1)  015466  013777  001102  163446  $OVER:   MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER
    (1)  015474  013716  001106                   MOV     $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
    (1)  015500  000002                           RTI                      ;;FIXES PS
    (1)  015502  003720                  $MXCNT:  2000.                    ;;MAX. NUMBER OF ITERATIONS
  2605                                   .SBTTL  ERROR HANDLER ROUTINE
    (1)
    (2)                                  ;;*************************************************************
    (1)                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
    (1)                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
    (1)                                  ;*AND GO TO $ERRTYP ON ERROR
    (1)                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
    (1)                                  ;*SW15=1         HALT ON ERROR
    (1)                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
    (1)                                  ;*SW10=1         BELL ON ERROR
    (1)                                  ;*SW09=1         LOOP ON ERROR
    (1)                                  ;*CALL
    (1)                                  ;*       ERROR   N        ;;ERROR=EMT AND N=ERROR ITEM NUMBER
    (1)
    (1)  015504                          $ERROR:
    (1)  015504  105237  001103          7$:      INCB    $ERFLG           ;;SET THE ERROR FLAG
    (1)  015510  001775                           BEQ     7$               ;;DON'T LET THE FLAG GO TO ZERO
    (1)  015512  013777  001102  163422           MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
    (1)  015520  032777  002000  163412           BIT     #BIT10,@SWR      ;;BELL ON ERROR?
    (1)  015526  001402                           BEQ     1$               ;;NO - SKIP
    (1)  015530  104401  001164                   TYPE    ,$BELL           ;;RING BELL
    (1)  015534  005237  001112          1$:      INC     $ERTTL           ;;COUNT THE NUMBER OF ERRORS
    (1)  015540  011637  001116                   MOV     (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
    (1)  015544  162737  000002  001116           SUB     #2,$ERRPC
    (1)  015552  117737  163340  001114           MOVB    @$ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
    (1)  015560  032777  020000  163352           BIT     #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
    (1)  015566  001004                           BNE     20$              ;;SKIP TYPEOUTS
    (1)  015570  004737  015700                   JSR     PC,$ERRTYP       ;;GO TO USER ERROR ROUTINE
    (1)  015574  104401  001171                   TYPE    ,$CRLF
    (1)  015600                          20$:
    (1)  015600  122737  000001  001214           CMPB    #APTENV,$ENV     ;;RUNNING IN APT MODE
    (1)  015606  001007                           BNE     2$               ;;NO,SKIP APT ERROR REPORT
    (1)  015610  113737  001114  015622           MOVB    $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
    (1)  015616  004737  016334                   JSR     PC,$ATY4         ;;REPORT FATAL ERROR TO APT
    (1)  015622     000                  21$:     .BYTE   0
    (1)  015623     000                           .BYTE   0
    (1)  015624  000777                  22$:     BR      22$              ;;APT ERROR LOOP
    (1)  015626  005777  163306          2$:      TST     @SWR             ;;HALT ON ERROR
    (1)  015632  100001                           BPL     3$               ;;SKIP IF CONTINUE
    (1)  015634  000000                           HALT                     ;;HALT ON ERROR!
    (1)  015636  032777  001000  163274  3$:      BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
    (1)  015644  001402                           BEQ     4$               ;;BR IF NO
    (1)  015646  013716  001110                   MOV     $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
    (1)  015652  005737  001162          4$:      TST     $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
    (1)  015656  001402                           BEQ     5$               ;;BR IF NONE
    (1)  015660  013716  001162                   MOV     $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
    (1)  015664                          5$:
    (1)  015664  022737  012074  000042           CMP     #$ENDAD,@#42     ;;ACT-11 AUTO-ACCEPT?
    (1)  015672  001001                           BNE     6$               ;;BRANCH IF NO
    (1)  015674  000000                     HALT                            ;;YES
    (1)  015676                       6$:
    (1)  015676  000002                     RTI                             ;;RETURN
   2606                               .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
    (1)
    (2)                               ;;*******************************************************************
    (1)                               ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
    (1)                               ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
    (1)                               ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
    (1)
    (1)  015700                       $ERRTYP:
    (1)  015700  104401  001171             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  015704  010046                     MOV     R0,-(SP)        ;;SAVE R0
    (1)  015706  005000                     CLR     R0              ;;PICKUP THE ITEM INDEX
    (1)  015710  153700  001114             BISB    @#$ITEMB,R0
    (1)  015714  001004                     BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
    (1)                                                             ;;TYPE THE PC OF THE ERROR
    (2)  015716  013746  001116             MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
    (2)                                                             ;;ERROR ADDRESS
    (2)  015722  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)  015724  000426                     BR      6$              ;;GET OUT
    (1)  015726  005300               1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
    (1)  015730  006300                     ASL     R0              ;;      WORK FOR THE ERROR TABLE
    (1)  015732  006300                     ASL     R0
    (1)  015734  006300                     ASL     R0
    (1)  015736  062700  001256             ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
    (1)  015742  012037  015752             MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
    (1)  015746  001404                     BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
    (1)  015750  104401                     TYPE                    ;;TYPE THE "ERROR MESSAGE"
    (1)  015752  000000               2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
    (1)  015754  104401  001171             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  015760  012037  015770       3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
    (1)  015764  001404                     BEQ     5$              ;;SKIP TYPEOUT IF 0
    (1)  015766  104401                     TYPE                    ;;TYPE THE "DATA HEADER"
    (1)  015770  000000               4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
    (1)  015772  104401  001171             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  015776  011000               5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
    (1)  016000  001004                     BNE     7$              ;;GO TYPE THE DATA
    (1)  016002  012600               6$:   MOV     (SP)+,R0        ;;RESTORE R0
    (1)  016004  104401  001171             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  016010  000207                     RTS     PC              ;;RETURN
    (1)  016012                       7$:
    (2)  016012  013046                     MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
    (2)  016014  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)  016016  005710                     TST     (R0)            ;;IS THERE ANOTHER NUMBER?
    (1)  016020  001770                     BEQ     6$              ;;BR IF NO
    (1)  016022  104401  016030             TYPE    ,8$             ;;TYPE TWO(2) SPACES
    (1)  016026  000771                     BR      7$              ;;LOOP
    (1)  016030  020040     000       8$:   .ASCIZ  /  /            ;;TWO(2) SPACES
    (1)          016034                     .EVEN
```

```
 2608                                    .SBTTL  TYPE ROUTINE
  (1)
  (2)                                    ;;********************************************************************
  (1)                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
  (1)                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
  (1)                                    ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
  (1)                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
  (1)                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
  (1)                                    ;*
  (1)                                    ;*CALL:
  (1)                                    ;*1) USING A TRAP INSTRUCTION
  (1)                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
  (1)                                    ;*OR
  (1)                                    ;*      TYPE
  (1)                                    ;*      MESADR
  (1)                                    ;*
  (1)
  (1) 016034 105737 001157               $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1) 016040 100002                              BPL     1$              ;;BR IF YES
  (1) 016042 000000                              HALT                    ;;HALT HERE IF NO TERMINAL
  (1) 016044 000430                              BR      3$              ;;LEAVE
  (1) 016046 010046                      1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1) 016050 017600 000002                       MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1) 016054 122737 000001 001214                CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1) 016062 001011                              BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
  (1) 016064 132737 000100 001215                BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
  (1) 016072 001405                              BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
  (1) 016074 010037 016104                       MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
  (1) 016100 004737 016324                       JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
  (1) 016104 000000                      61$:    .WORD   0               ;;MESSAGE ADDRESS
  (1) 016106 132737 000040 001215        62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
  (1) 016114 001003                              BNE     60$             ;;YES,SKIP TYPE OUT
  (1) 016116 112046                      2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
  (1) 016120 001005                              BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
  (1) 016122 005726                              TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
  (1) 016124 012600                      60$:    MOV     (SP)+,R0        ;;RESTORE R0
  (1) 016126 062716 000002               3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
  (1) 016132 000002                              RTI                     ;;RETURN
  (1) 016134 122716 000011               4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
  (1) 016140 001430                              BEQ     8$
  (1) 016142 122716 000200                       CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
  (1) 016146 001006                              BNE     5$
  (1) 016150 005726                              TST     (SP)+           ;;POP  <CR><LF> EQUIV
  (1) 016152 104401                              TYPE                    ;;TYPE A CR AND LF
  (1) 016154 001171                              $CRLF
  (1) 016156 105037 016312                       CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
  (1) 016162 000755                              BR      2$              ;;GET NEXT CHARACTER
  (1) 016164 004737 016246               5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
  (1) 016170 123726 001156               6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
  (1) 016174 001350                              BNE     2$              ;;IF NO GO GET NEXT CHAR.
  (1) 016176 013746 001154                       MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
  (1)                                                                    ;;AND THE NULL CHAR.
  (1) 016202 105366 000001               7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
  (1) 016206 002770                              BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
  (1) 016210 004737 016246                       JSR     PC,$TYPEC       ;;GO TYPE A NULL
  (1) 016214 105337 016312                       DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 (1)  016220  000770                           BR      7$              ;;LOOP
 (1)
 (1)                                  ;HORIZONTAL TAB PROCESSOR
 (1)
 (1)  016222  112716  000040          8$:      MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 (1)  016226  004737  016246          9$:      JSR     PC,$TYPEC       ;;TYPE A SPACE
 (1)  016232  132737  000007  016312           BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 (1)  016240  001372                           BNE     9$              ;;TAB STOP
 (1)  016242  005726                           TST     (SP)+           ;;POP SPACE OFF STACK
 (1)  016244  000724                           BR      2$              ;;GET NEXT CHARACTER
 (1)  016246  105777  162676          $TYPEC:  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 (1)  016252  100375                           BPL     $TYPEC
 (1)  016254  116677  000002  162670           MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 (1)  016262  122766  000015  000002           CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 (1)  016270  001003                           BNE     1$              ;;BRANCH IF NO
 (1)  016272  105037  016312                   CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 (1)  016276  000406                           BR      $TYPEX          ;;EXIT
 (1)  016300  122766  000012  000002  1$:      CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 (1)  016306  001402                           BEQ     $TYPEX          ;;BRANCH IF YES
 (1)  016310  105227                           INCB    (PC)+           ;;COUNT THE CHARACTER
 (1)  016312  000000                  $CHARCNT:.WORD   0               ;;CHARACTER COUNT STORAGE
 (1)  016314  000207                  $TYPEX:  RTS     PC
 (1)
2609                                  .SBTTL  APT COMMUNICATIONS ROUTINE
 (1)
 (2)                                  ;;*****************************************************************
 (1)  016316  112737  000001  016562  $ATY1:   MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
 (1)  016324  112737  000001  016560  $ATY3:   MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
 (1)  016332  000403                           BR      $ATYC
 (1)  016334  112737  000001  016562  $ATY4:   MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
 (1)  016342                          $ATYC:
 (3)  016342  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 (3)  016344  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (1)  016346  105737  016560                   TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
 (1)  016352  001450                           BEQ     5$              ;;IF NOT:  BR
 (1)  016354  122737  000001  001214           CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
 (1)  016362  001031                           BNE     3$              ;;IF NOT:  BR
 (1)  016364  132737  000100  001215           BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 (1)  016372  001425                           BEQ     3$              ;;IF NOT:  BR
 (1)  016374  017600  000004                   MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
 (1)  016400  062766  000002  000004           ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 (1)  016406  005737  001174          1$:      TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
 (1)  016412  001375                           BNE     1$              ;;IF NOT:  WAIT
 (1)  016414  010037  001210                   MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
 (1)  016420  105720                  2$:      TSTB    (R0)+           ;;FIND END OF MESSAGE
 (1)  016422  001376                           BNE     2$
 (1)  016424  163700  001210                   SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
 (1)  016430  006200                           ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
 (1)  016432  010037  001212                   MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
 (1)  016436  012737  000004  001174           MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
 (1)  016444  000413                           BR      5$
 (1)  016446  017637  000004  016472  3$:      MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
 (1)  016454  062766  000002  000004           ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 (3)  016462  013746  177776                   MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 (1)  016466  004737  016034                   JSR     PC,$TYPE        ;;CALL TYPE MACRO
 (1)  016472  000000                  4$:      .WORD   0
```

```
  (1)  016474                          5$:
  (1)  016474  105737  016562          10$:      TSTB     $FFLG              ;;SHOULD REPORT FATAL ERROR?
  (1)  016500  001416                            BEQ      12$                ;;IF NOT:  BR
  (1)  016502  005737  001214                    TST      $ENV               ;;RUNNING UNDER APT?
  (1)  016506  001413                            BEQ      12$                ;;IF NOT:  BR
  (1)  016510  005737  001174          11$:      TST      $MSGTYPE           ;;FINISHED LAST MESSAGE?
  (1)  016514  001375                            BNE      11$                ;;IF NOT:  WAIT
  (1)  016516  017637  000004  001176            MOV      @4(SP),$FATAL      ;;GET ERROR #
  (1)  016524  062766  000002  000004            ADD      #2,4(SP)                   ;;BUMP RETURN ADDR.
  (1)  016532  005237  001174                    INC      $MSGTYPE           ;;TELL APT TO TAKE ERROR
  (1)  016536  105037  016562          12$:      CLRB     $FFLG              ;;CLEAR FATAL FLAG
  (1)  016542  105037  016561                    CLRB     $LFLG              ;;CLEAR LOG FLAG
  (1)  016546  105037  016560                    CLRB     $MFLG              ;;CLEAR MESSAGE FLAG
  (3)  016552  012601                            MOV      (SP)+,R1           ;;POP STACK INTO R1
  (3)  016554  012600                            MOV      (SP)+,R0           ;;POP STACK INTO R0
  (1)  016556  000207                            RTS      PC                 ;;RETURN
  (1)  016560     000                  $MFLG:    .BYTE    0                  ;;MESSG. FLAG
  (1)  016561     000                  $LFLG:    .BYTE    0                  ;;LOG FLAG
  (1)  016562     000                  $FFLG:    .BYTE    0                  ;;FATAL FLAG
  (1)          016564                            .EVEN
  (1)          000200                  APTSIZE=200
  (1)          000001                  APTENV=001
  (1)          000100                  APTSPOOL=100
  (1)          000040                  APTCSUP=040
 2610
  (2)                                            ;*
  (2)                                            ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
  (2)                                            ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
  (2)                                            ;*NEXT WE WILL INIT BOTH UPROCESSORS
  (2)                                            ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
  (2)                                            ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
  (2)                                            ;*
  (2)                                            ;*       CALL=   JSR      R5,$LPAI
  (2)                                            ;*               .WORD    0                  ;ADDR. OF DEVICE ADDRESS.
  (2)                                            ;* ROUTINES REQUIRED:    .LOADLP
  (2)                                            ;* PROGRAMS REQUIRED:    DRLPX2
  (2)                                            ;*
  (2)
  (2)                                            ;*              ;RETURNS WITH $AERR=1 IF SLAVE
  (2)                                            ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
  (2)                                            ;*
  (2)  016564                          $LPAI:
  (2)  016564  013746  000004                    MOV      4,-(SP)
  (2)
  (2)  016570  000413                            BR       31$                ;FIELD DOES NOT HAVE A BUS SWITCH TO
  (2)                                                                        ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
  (2)                                                                        ;BRANCH ARROUD THE NEXT CODE THAT
  (2)                                                                        ;WORKS BASED ON A BUS SWITCH.
  (2)                                                                        ;CODE LEFT IN HERE FOR IN HOUSE
  (2)                                                                        ;PERSONAL WHO MAY PATCH THIS BRANCH
  (2)                                                                        ;INSTRUCTION TO A <NOP> OCTAL <240>
  (2)                                                                        ;IN ORDER TO RUN PROGRAM WITH A SWITCH.
  (2)
  (2)                                                                        ;NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
  (2)                                                                        ;TEST EQUIPMENT ONLY IT CONNECTS
  (2)                                                                        ;THE UNIBUS TO THE I/O BUS FOR
 (2)                                                                     ;CERTAIN TESTING.
 (2)  016572  012737  016616  000004           MOV     #30$,4
 (2)  016600  005237  170000                   INC     170000
 (3)  016604  104401  016612                   TYPE    ,65$             ;;TYPE ASCIZ STRING
 (3)  016610  000401                           BR      64$              ;;GET OVER THE ASCIZ
 (3)                                  ;;65$:   .ASCIZ  <7>##
 (3)  016614                          64$:
 (2)  016614  000401                           BR      31$
 (2)  016616  022626                  30$:     CMP     (SP)+,(SP)+
 (2)  016620  012637  000004          31$:     MOV     (SP)+,4          ;ALL THIS JUNK MUST BE REMOVED!.
 (2)  016624  005037  017252                   CLR     $AERR
 (2)  016630  004537  017254                   JSR     R5,$LOAD         ;LOAD MICRO-CODE.
 (2)  016634  000000G                          .WORD   DRLPX2           ;FILE "DRLPX2.OBJ"
 (2)
 (2)  016636  052777  040000  162570           BIS     #BIT14,@KMAD0    ;ISSUE KMC+DMC INIT.
 (2)
 (2)  016644                          1$:
 (2)                                                                     ;"HANGS" HERE THEN KMC-11 ERROR.
 (2)  016644  010146                           MOV     R1,-(SP)
 (2)  016646  005001                           CLR     R1
 (2)  016650  005201                  2$:      INC     R1               ;STALL FOR DMC-UP
 (2)  016652  001376                           BNE     2$
 (2)  016654  012777  104000  162552           MOV     #BIT15.BIT11,@KMAD0      ;SET RUN, AND ENABLE ARBITRATION.
 (2)  016662  105201                  25$:     INCB    R1
 (2)  016664  001376                           BNE     25$
 (2)
 (2)  016666  032777  000040  162540           BIT     #BIT5,@KMAD0     ;SLAVE READY? (READING IPBM SR)
 (2)  016674  001401                           BEQ     3$
 (2)                                                                     ;FATAL LPA-11 ERROR SLAVE NOT READY.
 (2)  016676  104000                           ERROR
 (2)
 (2)  016700  012777  000004  162532  3$:      MOV     #4,@KMAD2        ;READ FAST PATH
 (2)  016706                          4$:
 (3)  016706  004537  020164                   JSR     R5, $TOUT        ;-TOUT-CHECK FOR TIMEOUT
 (3)
 (3)  016712  104000                           ERROR                    ;/TIME-OUT ERROR
 (3)                                                                     ;/WE FAILED TO COMPLETE
 (3)                                                                     ;/CURRENT OPERATION.
 (3)                                                                     ;/CONTINUES IN THIS LOOP
 (3)                                                                     ;/WOULD MAKE US "HANG" HERE
 (3)
 (3)  016714  000774                           BR              4$
 (3)
 (3)                                                                     ;/RETURNS HERE-FROM-TIMED OUT.
 (2)  016716  122777  000377  162514           CMPB    #377,@KMAD2      ;WAIT TILL KMC DONE COMMAND.
 (2)  016724  001370                           BNE     4$
 (2)  016726  122777  000377  162510           CMPB    #377,@KMAD4      ;IF FAST PATH=377 THEN ERROR.
 (2)  016734  001001                           BNE     35$
 (2)  016736  104000                           ERROR                    ;IPBM ERROR (SLAVE SIDE)
 (2)                                                                     ;YOU MUST RUN IPBM DIAGNOSTIC.
 (2)
 (2)  016740  117737  162500  017220  35$:     MOVB    @KMAD4,11$       ;GET THE VERSION NUMBER FROM DMC-11
 (2)  016746  005227  177777                   INC     #-1
 (2)  016752  001045                           BNE     5$
 (2)  016754  005227  177777                   INC     #-1
 (2)  016760  001042                           BNE     5$
```

```
    (3)  016762  104401  016770                    TYPE    ,67$            ;;TYPE ASCIZ STRING
    (3)  016766  000426                            BR      66$             ;;GET OVER THE ASCIZ
    (3)                                   ;;67$:   .ASCIZ  <200>"M8200-YC (DMC) MICROCODE VERSION NUMBER = "
    (3)  017044                           66$:
    (2)  017044  013746  017220                    MOV     11$,-(SP)
    (2)  017050  104403                            TYPOS
    (2)  017052     002     000                    .BYTE   2,0
    (3)  017054  104401  017062                    TYPE    ,69$            ;;TYPE ASCIZ STRING
    (3)  017060  000402                            BR      68$             ;;GET OVER THE ASCIZ
    (3)                                   ;;69$:   .ASCIZ  <200>"  "
    (3)  017066                           68$:
    (2)
    (2)  017066  112737  177777  017220   5$:      MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
    (2)  017074  012501                            MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
    (2)  017076  021127  000000           6$:      CMP     (R1),#0         ;TERM REACHED?
    (2)  017102  001444                            BEQ     10$
    (2)  017104  105237  017220                    INCB    11$
    (2)  017110  113777  017220  162326            MOVB    11$,@KMAD4      ;FIFO DATA
    (2)  017116  004737  017222                    JSR     PC,20$          ;ISSUE SEND
    (2)  017122  112177  162316                    MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
    (2)  017126  004737  017222                    JSR     PC,20$          ;ISSUE SEND
    (2)  017132  112177  162306                    MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
    (2)  017136  004737  017222                    JSR     PC,20$
    (2)
    (2)  017142  032777  000002  162264   7$:      BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
    (2)  017150  001374                            BNE     7$              ;=1 NO DATA. =0 DATA.
    (2)  017152  112777  000002  162260            MOVB    #2,@KMAD2       ;READ FIFO.
    (2)
    (2)  017160                           8$:
    (3)  017160  004537  020164                    JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  017164  104000                            ERROR                   ;/TIME-OUT ERROR
    (3)                                                                    ;/WE FAILED TO COMPLETE
    (3)                                                                    ;/CURRENT OPERATION.
    (3)                                                                    ;/CONTINUES IN THIS LOOP
    (3)                                                                    ;/WOULD MAKE US "HANG" HERE
    (3)
    (3)  017166  000774                            BR              8$
    (3)
    (3)                                                                    ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  017170  122777  000377  162242            CMPB    #377,@KMAD2     ;WAIT FOR READ.
    (2)  017176  001370                            BNE     8$
    (2)  017200  105777  162240                    TSTB    @KMAD4          ;WAS A ZERO RETURNED?
    (2)  017204  001734                            BEQ     6$              ;YES GET NEXT ADDR.
    (2)                                                                    ;SLAVE WILL RETURN CODE 0 IF
    (2)  017206  005237  017252                    INC     $AERR           ;DEV PRESENT.   ELSE
    (2)                                                                    ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
    (2)  017212  005041                            CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
    (2)  017214  012601                   10$:     MOV     (SP)+,R1
    (2)  017216  000205                            RTS     R5              ;RETURN ALL ADDR. CHECKED.
    (2)
    (2)  017220  000000                   11$:     .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
    (2)                                                                    ;TO SLAVES ADDR. TABLE.
    (2)
    (2)  017222  112777  000003  162210   20$:     MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
    (2)  017230                           21$:
```

```
(3)  017230  004537  020164                    JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  017234  104000                            ERROR                      ;/TIME-OUT ERROR
(3)                                                                       ;/WE FAILED TO COMPLETE
(3)                                                                       ;/CURRENT OPERATION.
(3)                                                                       ;/CONTINUES IN THIS LOOP
(3)                                                                       ;/WOULD MAKE US ''HANG'' HERE
(3)
(3)  017236  000774                            BR              21$
(3)
(3)                                                                       ;/RETURNS HERE-FROM-TIMED OUT.
(2)  017240  122777  000377  162172            CMPB    #377,@KMAD2        ;KMC CODE WILL RETURN A ''377''
(2)  017246  001370                            BNE     21$                ;WHEN DONE COMMAND.
(2)  017250  000207                            RTS     PC
(2)
(2)  017252  000000                   $AERR:   .WORD   0                  ;=0 IF ADDR. LIST OK,=1 IF BAD.
(2)
(2)                                            ;*
(2)                                            ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
(2)                                            ;*      CALL =  JSR     R5,$LOAD
(2)                                            ;*              .WORD   XX                 ;ADDR. OF MICRO CODE.
(2)                                            ;*              ;RETURNS HERE
(2)                                            ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
(2)                                            ;*
(2)
(2)  017254  010446                   $LOAD:   MOV     R4,-(SP)           ;SAVE R4.
(2)  017256  010046                            MOV     R0,-(SP)           ;SAVE R0.
(2)  017260  012500                   1$:      MOV     (5)+,R0            ;GET PROG. ADDR.
(2)  017262  005077  162146                    CLR     @KMAD0             ;CLEAR CSR
(2)  017266  005077  162152                    CLR     @KMAD4             ;CLEAR CRAM ADDR.
(2)  017272  052777  002000  162134   2$:      BIS     #2000,@KMAD0       ;SELECT CRAM.
(2)  017300  012077  162144                    MOV     (0)+,@KMAD6        ;WRITE DATA.
(2)  017304  052777  020000  162122            BIS     #20000,@KMAD0      ;SET CRAM WRITE
(2)  017312  005077  162116                    CLR     @KMAD0             ;DISABLE CRAM.
(2)  017316  005277  162122                    INC     @KMAD4             ;UPDATE CRAM ADDR.
(2)  017322  021027  177777                    CMP     (0),#-1            ;ALL DONE?
(2)  017326  001361                            BNE     2$                 ;NO LOOP.
(2)  017330  005077  162110                    CLR     @KMAD4             ;CLEAR CRAM ADDR.
(2)  017334  016500  177776                    MOV     -2(5),R0           ;GET MICRO CODE ADDR.
(2)
(2)  017340  052777  002000  162066   3$:      BIS     #2000,@KMAD0       ;SELECT CRAM
(2)  017346  022077  162076                    CMP     (R0)+,@KMAD6       ;DATA OK?
(2)  017352  001013                            BNE     5$                 ;NO - REPORT AN ERROR.
(2)  017354  021027  177777                    CMP     (0),#-1            ;ALL DONE?
(2)  017360  001405                            BEQ     4$                 ;YES - EXIT
(2)  017362  005077  162046                    CLR     @KMAD0             ;NO - DESELECT CRAM.
(2)  017366  005277  162052                    INC     @KMAD4             ;UPDATE CRAM ADDR.
(2)  017372  000762                            BR      3$
(2)
(2)  017374  012600                   4$:      MOV     (SP)+,R0           ;RESTORE R0
(2)  017376  012604                            MOV     (SP)+,R4           ;RESTORE R4
(2)  017400  000205                            RTS     R5                 ;EXIT
(2)
(2)  017402                           5$:                                 ;COME HERE ON LOAD ERROR
(2)  017402  005745                            TST     -(5)
(2)  017404  105204                            INCB    R4                 ;UPDATE ERROR COUNTER.
```

```
(2)  017406  100324                        BPL     1$                ;IF NOT TOO MANY, TRY AGAIN.
(2)  017410  000000                        HALT                      ;MICRO CODE LOAD ERROR.
(2)                                                                  ;KMC-11 FAULT. YOU COULD TRY
(2)  017412  000722                        BR      1$                ;TO PRESS CONTINUE TO GIVE IT
(2)                                                                  ;ANOTHER CHANCE, BUT I DOUBT
(2)                                                                  ;THAT THAT WOULD WORK. SINCE I'VE
(2)                                                                  ;ALREADY GIVEN IT 177 (OCTAL) CHANCES.
(2)                                                                  ;TRY RUNNING THE KMC-11 DIAGNOSTIC.
(2)
(2)
(2)                                        ;
(2)                                        ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
(2)                                        ;*
(2)                                        ;*      CALL =  JSR     R5,$TLKW
(2)                                        ;*              .WORD   0                   ;OFFSET OF DEVICE ADDR.
(2)                                        ;*              .WORD   0                   ;DATA TO BE WRITTEN
(2)                                        ;*
(2)  017414  010046                $TLKW:  MOV     R0,-(SP)          ;SAVE R0
(2)  017416  012500                        MOV     (5)+,R0           ;GET DEVICE OFFSET
(2)  017420  052700  000340                BIS     #340,R0           ;ADD WRITE CODE.
(2)  017424  004737  017676                JSR     PC,$LPW           ;WAIT FOR FAST PATH READY
(2)  017430  010037  017522                MOV     R0,W1
(2)  017434  010077  162004                MOV     R0,@KMAD4
(2)  017440  112777  000005  161772        MOVB    #5,@KMAD2         ;ISSUE FAST PATH WRITE
(2)  017446  004737  017676                JSR     PC,$LPW           ;WAIT FOR RDY
(2)  017452  011537  017524                MOV     (5),W2
(2)  017456  112577  161762                MOVB    (5)+,@KMAD4       ;WRITE LOW BYTE DATA.
(2)
(2)  017462  112777  000005  161750        MOVB    #5,@KMAD2         ;FP WRITE
(2)  017470  004737  017676                JSR     PC,$LPW
(2)  017474  111537  017526                MOVB    (5),W3
(2)  017500  112577  161740                MOVB    (5)+,@KMAD4       ;WRITE HIGH BYTE
(2)  017504  112777  000005  161726        MOVB    #5,@KMAD2
(2)  017512  004737  017676                JSR     PC,$LPW
(2)  017516  012600                        MOV     (SP)+,R0
(2)  017520  000205                        RTS     R5                ;EXIT DONE.
(2)  017522  000000                W1:     0
(2)  017524  000000                W2:     0
(2)  017526  000000                W3:     0
(2)
(2)                                        ;*
(2)                                        ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
(2)                                        ;*
(2)                                        ;*      CALL =  JSR     R5,$TLKR
(2)                                        ;*              .WORD   0                   ;OFFSET OF DEVICE
(2)                                        ;*              ;RETURNS HERE
(2)                                        ;*DATA IN WORD $DATR
(2)                                        ;*
(2)
(2)  017530  010046                $TLKR:  MOV     R0,-(SP)          ;SAVE R0
(2)  017532  012500                        MOV     (5)+,R0           ;GET OFFSET
(2)  017534  052700  000300                BIS     #300,R0           ;ADD READ CODE
(2)  017540  004737  017676                JSR     PC,$LPW           ;WAIT TILL READY
(2)  017544  110077  161674                MOVB    R0,@KMAD4
(2)  017550  112777  000005  161662        MOVB    #5,@KMAD2         ;ISSUE WRITE FP
(2)  017556  004737  017676                JSR     PC,$LPW
```

```
    (2)  017562  010037  017672                     MOV     RO,RD1
    (2)  017566                            1$:
    (3)  017566  004537  020164                     JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  017572  104000                             ERROR                      ;/TIME-OUT ERROR
    (3)                                                                        ;/WE FAILED TO COMPLETE
    (3)                                                                        ;/CURRENT OPERATION.
    (3)                                                                        ;/CONTINUES IN THIS LOOP
    (3)                                                                        ;/WOULD MAKE US "HANG" HERE
    (3)
    (3)  017574  000774                             BR              1$
    (3)
    (3)                                                                        ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  017576  032777  000040  161630             BIT     #BIT5,@KMAD0       ;FAST PATH GOT DATA?
    (2)  017604  001370                             BNE     1$
    (2)  017606  112777  000004  161624             MOVB    #4,@KMAD2          ;ISSUE FAST PATH READ
    (2)  017614  004737  017676                     JSR     PC,$LPW
    (2)  017620  117737  161620  017674             MOVB    @KMAD4,$DATR       ;GET LOW BYTE
    (2)  017626                            2$:
    (3)  017626  004537  020164                     JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  017632  104000                             ERROR                      ;/TIME-OUT ERROR
    (3)                                                                        ;/WE FAILED TO COMPLETE
    (3)                                                                        ;/CURRENT OPERATION.
    (3)                                                                        ;/CONTINUES IN THIS LOOP
    (3)                                                                        ;/WOULD MAKE US "HANG" HERE
    (3)
    (3)  017634  000774                             BR              2$
    (3)
    (3)                                                                        ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  017636  032777  000040  161570             BIT     #BIT5,@KMAD0       ;FAST PATH READY?
    (2)  017644  001370                             BNE     2$
    (2)  017646  112777  000004  161564             MOVB    #4,@KMAD2          ;ISSUE FAST PATH READ
    (2)  017654  004737  017676                     JSR     PC,$LPW
    (2)  017660  117737  161560  017675             MOVB    @KMAD4,$DATR+1     ;SAVE HIGH BYTE
    (2)  017666  012600                             MOV     (SP)+,R0
    (2)  017670  000205                             RTS     R5
    (2)  017672  000000                    RD1:     0
    (2)  017674  000000                    $DATR:   .WORD   0
    (2)                                             ;
    (2)                                             ;THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
    (2)                                             ;AS FAST PATH TO BE READ.
    (2)                                             ;
    (2)                                             ;       CALL =  JSR     PC,$LPW
    (2)                                             ;
    (2)                                             ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
    (2)                                             ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
    (2)                                             ;
    (2)
    (2)  017676  010146                    $LPW:    MOV     R1,-(SP)           ;SAVE R1
    (2)  017700  005001                             CLR     R1
    (2)  017702  122777  000377  161530    1$:      CMPB    #377,@KMAD2        ;FINISHED INSTRUCTION?
    (2)  017710  001403                             BEQ     2$
    (2)  017712  005201                             INC     R1                 ;TIME OUT?
    (2)  017714  001372                             BNE     1$
    (2)  017716  000411                             BR      10$
```

```
(2)
(2)  017720  032777  000020  161506  2$:     BIT     #BIT4,@KMAD0     ;FAST PATH READ?
(2)  017726  001403                          BEQ     3$
(2)  017730  005201                          INC     R1               ;NO - TIME OUT?
(2)  017732  001372                          BNE     2$
(2)  017734  000402                          BR      10$              ;YES - REPORT AN ERROR
(2)
(2)  017736  012601                  3$:     MOV     (SP)+,R1         ;RESTORE R1
(2)  017740  000207                          RTS     PC               ;EXIT
(2)
(2)  017742                          10$:
(3)  017742  104401  017750                  TYPE    ,65$             ;;TYPE ASCIZ STRING
(3)  017746  000407                          BR      64$              ;;GET OVER THE ASCIZ
(3)                                  ;;65$:  .ASCIZ  <200>#LPA-11 FAULT#
(3)  017766                          64$:
(2)
(2)  017766  000000                  11$:    HALT                     ;LPA-11 FAULT RUN LPA-11
(2)  017770  000776                          BR      11$              ;DIAGNOSTICS.
(2)
(2)
(2)
(2)                                          ;*
(2)                                          ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
(2)                                          ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
(2)                                          ;*
(2)                                          ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
(2)                                          ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
(2)                                          ;* THAT ADDRESS.
(2)                                          ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
(2)                                          ;* $TLKW
(2)                                          ;*
(2)
(2)  017772  010046                  $OUTLP: MOV     R0,-(SP)         ;SAVE R0
(2)  017774  010146                          MOV     R1,-(SP)         ;SAVE R1
(2)
(2)  017776  012700  001462                  MOV     #.DVLS,R0        ;PROGRAM DEFINED LIST.
(2)  020002  005001                          CLR     R1
(2)  020004  005710                  1$:     TST     (0)              ;TERMINATOR REACHED?
(2)  020006  001421                          BEQ     10$              ;YES NEXT STEP.
(2)  020010  027520  000000                  CMP     @(5),(0)+        ;MATCH WITH ADDR IN LIST?
(2)  020014  001402                          BEQ     2$
(2)  020016  005201                          INC     R1
(2)  020020  000771                          BR      1$
(2)
(2)  020022  010137  020040          2$:     MOV     R1,3$            ;SAVE OFFSET, DEVICE KNOWN.
(2)  020026  005725                          TST     (5)+
(2)  020030  013537  020042                  MOV     @(5)+,4$         ;GET DATA TO BE WRITTEN
(2)  020034  004537  017414                  JSR     R5,$TLKW         ;DO WRITE
(2)  020040  000000                  3$:     .WORD   0                ;DEVICE OFFSET
(2)  020042  000000                  4$:     .WORD   0                ;DATA TO BE WRITTEN.
(2)  020044  012601                          MOV     (SP)+,R1
(2)  020046  012600                          MOV     (SP)+,R0
(2)  020050  000205                          RTS     R5
(2)  020052  017520  000000          10$:    MOV     @(5),(0)+        ;SAVE ADDR.
(2)  020056  005010                          CLR     (0)
(2)  020060  004537  016564                  JSR     R5,$LPAI
```

```
  (2)  020064  001462                          .WORD   .DVLS
  (2)  020066  000755                          BR      2$
  (2)
  (2)                                          ;*
  (2)                                          ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
  (2)                                          ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
  (2)                                          ;*
  (2)                                          ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
  (2)                                          ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
  (2)                                          ;*WITH THE NEW ADDR.
  (2)                                          ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
  (2)                                          ;*$TLKR
  (2)                                          ;*      CALL THROUGH    MOVEI   DATA,ADDR.
  (2)                                          ;*              WHICH EQUALS:
  (2)                                          ;*                      JSR     R5,$INLP
  (2)                                          ;*                      .WORD   XX      ADDR OF DEVICE
  (2)                                          ;*                      .WORD   YY      ADDR TO STORE READ DATA.
  (2)
  (2)  020070  010046                  $INLP:  MOV     R0,-(SP)        ;SAVE R0
  (2)  020072  010146                          MOV     R1,-(SP)        ;SAVE R1
  (2)
  (2)  020074  012700  001462                  MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
  (2)  020100  005001                          CLR     R1
  (2)  020102  005710                  1$:     TST     (0)             ;EOL REACHED?
  (2)  020104  001420                          BEQ     10$             ;YES - DEFINE NEW ADDR.
  (2)
  (2)  020106  027520  000000                  CMP     @(5),(0)+       ;ADDR. MATCH?
  (2)  020112  001402                          BEQ     2$
  (2)  020114  005201                          INC     R1
  (2)  020116  000771                          BR      1$
  (2)
  (2)  020120  010137  020132          2$:     MOV     R1,3$           ;SAVE LIST OFFSET
  (2)  020124  005725                          TST     (5)+
  (2)  020126  004537  017530                  JSR     R5,$TLKR        ;GO READ DEVICE
  (2)          020132                  $OFS=.
  (2)  020132  000000                  3$:     .WORD   0               ;OFFSET OF DEVICE
  (2)
  (2)  020134  013735  017674                  MOV     $DATR,@(5)+     ;STORE DATA.
  (2)  020140  012601                          MOV     (SP)+,R1        ;RESTORE R1
  (2)  020142  012600                          MOV     (SP)+,R0        ;RESTORE R2
  (2)  020144  000205                          RTS     R5              ;EXIT
  (2)
  (2)  020146  017520  000000          10$:    MOV     @(5),(0)+
  (2)  020152  005010                          CLR     (0)
  (2)  020154  004537  016564                  JSR     R5,$LPAI
  (2)  020160  001462                          .WORD   .DVLS
  (2)  020162  000756                          BR      2$
  (2)                                                  ;*
  (2)                                                  ;*$TOUT ROUTINE USED TO WATCH IF
  (2)                                                  ;*      WE'RE IN A LOOP TOO-LONG
  (2)                                                  ;*      CALL=   JSR R5, $TOUT
  (2)                                                  ;*              ERROR X    ;RETURNS HERE ON TIMEOUT
  (2)                                                  ;*              BR
  (2)                                                  ;*              ;RETURNS HERE NO ERROR
  (2)                                                  ;*
  (2)
```

```
(2)  020164  020537  020220           $TOUT:  CMP     R5,$SAD          ;SAME ADDR?
(2)  020170  001405                           BEQ     1$
(2)  020172  010537  020220                   MOV     R5,$SAD          ;NO-SAVE THIS ADDR.
(2)  020176  005037  020222                   CLR     $CNT             ;CLR CNT AT ADDR.
(2)  020202  000403                           BR      2$
(2)  020204  005237  020222           1$:     INC     $CNT             ;OVERFLOW?
(2)  020210  100402                           BMI     3$               ;YES-ERROR RETURN
(2)  020212  062705  000004           2$:     ADD     #4,R5            ;NO-NON ERROR RETURN
(2)  020216  000205                   3$:     RTS     R5               ;RETURN.
(2)
(2)  020220  000000                   $SAD:   .WORD   0                ;CONTAINS LOOP ADDR.
(2)  020222  000000                   $CNT:   .WORD   0                ;# OF TIMES AT ADDR.
(2)
(2)                                           ;*
(2)                                           ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
(2)                                           ;*USE FOR A RESET.  FIRST,WE DO A RESET INSTRUCTION.
(2)                                           ;*THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
(2)                                           ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
(2)                                           ;*
(2)                                           ;*      CALL=JSR        PC,$RESET       ;REPLACES "RESET INSTRUCTION
(2)                                           ;*              ;RETURNS HERE.
(2)                                           ;*
(2)  020224  000005                   $RESET: RESET                    ;RESET THE WORLD.
(3)
(3)                                   .*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
(2)  020236  005737  017252                   TST     $AERR            ;IF NO ERROR,LOOP
(2)  020242  001004                           BNE     10$              ;THERE WAS AN ERROR.
(2)  020244  062737  000002  020260           ADD     #2,2$            ;UPDATE DEVICE ADDR.
(2)                                                                    ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
(2)                                                                    ;IF 2$ CONTAINED A VALID ADDR,WE
(2)                                                                    ;MUST KEEP TRYING UNTIL WE GENERATE
(2)                                                                    ;AN INVALID ADDR.
(2)  020252  000764                           BR      $RESET
(2)  020254                           10$:
(2)  020254  000207                           RTS     PC
(2)  020256  000000                   1$:     .WORD   0                ;JUNK LOC.
(2)  020260  160000                   2$:     .WORD   160000           ;DUMB ADDR. FORCES INIT OF DMC/KMC.
(2)
(2)
(2)                                           ;
(2)                                           ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
(2)                                           ;        IS NOT TIME DEPENDENT CODE SENCE
(2)                                           ;        NOT USED TO GET SPECIFIC TIME BUT
(2)                                           ;        JUST A LITTLE DELAY.
(2)
(2)                                           ;       THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
(2)                                           ;       THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
(2)                                           ;
(2)                                           ;
(2)                                           ;       CALL=   JSR PC,  SDELAY
(2)                                           ;
(2)  020262                           SDELAY:
(2)  020262  005737  020344                   TST     RTCCSR           ;CLOCK PRESENT?
(2)  020266  100016                           BPL     10$
(2)  020270  012737  000002  020334           MOV     #2,TIME
(2)  020276  052777  000115  000040           BIS     #115,@RTCCSR     ;START CLOCK
```

```
(2)  020304  005037  177776                    CLR     PS
(2)  020310  005737  020334           1$:      TST     TIME
(2)  020314  001375                            BNE     1$
(2)  020316  005077  000022                    CLR     @RTCCSR          ;STOP CLOCK
(2)
(2)  020322  000207                            RTS PC
(2)  020324  105237  020334           10$:     INCB    TIME
(2)  020330  001375                            BNE     10$
(2)  020332  000207                            RTS     PC
(2)
(2)  020334  000000                   TIME:    .WORD   0
(2)
(2)  020336  005337  020334           CLKINT:  DEC     TIME
(2)  020342  000002                            RTI
(2)  020344  000000                   RTCCSR:  .WORD   0                ;CLOCK CSR IF USED.
(2)                                            ;
(2)                                            ;*
(2)                                            ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
(2)                                            ;*ANY DEVICE ON THE I/O BUS
(2)                                            ;*USER MUST START AT THIS ADDR.
(2)                                            ;*HE MUST SAY EITHER "E" FOR EXAMINE, OR "D" FOR DEPOSIT.
(2)                                            ;*"E" IS DEFAULT.
(2)                                            ;*NEXT, HE MUST SUPPLY AN ADDR.
(2)                                            ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
(2)                                            ;*WILL OCCUR.
(2)
(2)  020346                           $UTK:
(2)  020346  005037  001462                    CLR     .DVLS
(2)  020352                           21$:
(3)  020352  104401  020360                    TYPE    ,65$             ;;TYPE ASCIZ STRING
(3)  020356  000405                            BR      64$              ;;GET OVER THE ASCIZ
(3)                                   ;;65$:   .ASCIZ  <200>#E OR D?#
(3)  020372                           64$:
(2)  020372  105777  160546           1$:      TSTB    @$TKS
(2)  020376  100375                            BPL     1$
(2)  020400  117737  160542  020522            MOVB    @$TKB,20$        ;GET INPUT
(2)  020406  104401  020522                    TYPE,   20$              ;ECHO, NEXT MESSAGE.
(2)  020412  142737  000240  020522            BICB    #240,20$         ;STRIP PARITY, LC
(2)  020420  104410                            RDOCT                    ;GET ADDR.
(2)  020422  012637  020520                    MOV     (SP)+,14$
(2)  020426  123727  020522  000104            CMPB    20$,#'D          ;DEPOSIT?
(2)  020434  001411                            BEQ     10$
(2)
(2)  020436  004537  020070                    JSR     R5,$INLP         ;GET DATA
(2)  020442  020520                   2$:      .WORD   14$
(2)  020444  020456                            .WORD   5$
(2)
(3)  020446  013746  020456                    MOV     5$,-(SP)         ;;SAVE 5$ FOR TYPEOUT
(3)  020452  104402                            TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(2)  020454  000736                            BR      21$              ;LOOP.
(2)  020456  000000                   5$:      .WORD   0
(2)
(2)  020460                           10$:
(3)  020460  104401  020466                    TYPE    ,67$             ;;TYPE ASCIZ STRING
(3)  020464  000404                            BR      66$              ;;GET OVER THE ASCIZ
(3)                                   ;;67$:   .ASCIZ  <200>#DATA= #
```

```
 (3)  020476                             66$:
 (2)  020476  104410                              RDOCT
 (2)  020500  012637  020516                      MOV     (SP)+,13$
 (2)
 (2)  020504  004537  017772             11$:     JSR     R5,$OUTLP          ;OUTPUT ROUTINE.
 (2)  020510  020520                     12$:     .WORD   14$                ;DEVICE ADDR.
 (2)  020512  020516                              .WORD   13$                ;DATA
 (2)  020514  000716                              BR      21$
 (2)
 (2)  020516  000000                     13$:     .WORD   0
 (2)  020520  000000                     14$:     .WORD   0
 (2)  020522  100001  042504  044526     20$:     .ASCIZ  <1><200>#DEVICE ADDR= #
 (2)  020530  042503  040440  042104
 (2)  020536  036522  000040
 (2)                                              .EVEN
 (2)
 (1)
 (1)
 (2)                                              ;
 (2)                                              ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
 (2)                                              ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
 (2)                                              ;TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
 (2)                                              ;SAMPLE TAKEING PURPOSES.
 (2)                                              ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
 (2)                                              ;       A/D CSR IN BSEL 4,AND 5.
 (2)                                              ;       (2) HE MUST CALL THIS ROUTINE:
 (2)                                              ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
 (2)                                              ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
 (2)                                              ;               ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
 (2)                                              ;                               ;(UNTILL ONE DOES A RESET)
 (2)                                              ;
 (2)                                              ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
 (2)                                              ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR.!
 (2)                                              ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
 (2)                                              ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
 (2)                                              ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
 (2)                                              ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
 (2)                                              ;
 (2)  020542  012537  020552             $PUTS:   MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
 (2)  020546  004537  020070                      JSR     R5,$INLP
 (2)  020552  000000                     1$:      .WORD   0
 (2)  020554  020650                              .WORD   10$
 (2)  020556  113777  020132  160664              MOVB    $OFS,@KMAD6
 (2)  020564  113777  020132  160660              MOVB    $OFS,@KMAD7
 (2)  020572  013737  020552  020612              MOV     1$,2$
 (2)  020600  062737  000002  020612              ADD     #2,2$
 (2)  020606  004537  020070                      JSR     R5,$INLP
 (2)  020612  000000                     2$:      .WORD   0
 (2)  020614  020650                              .WORD   10$
 (2)  020616  113777  020132  160616              MOVB    $OFS,@KMAD3
 (2)  020624  152777  000340  160616              BISB    #340,@KMAD6
 (2)  020632  152777  000300  160612              BISB    #300,@KMAD7
 (2)  020640  152777  000300  160574              BISB    #300,@KMAD3
 (2)  020646  000205                              RTS     R5
 (2)  020650  000000                     10$:     .WORD   0
 (2)
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 (1)
 (2)
 (1)                                    ;;*************************************************************
 (1)                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 (1)                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 (1)                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 (1)                                    ;*CALL:
 (1)                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                    ;*      TYPOS                   ;;CALL FOR TYPEOUT
 (1)                                    ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 (1)                                    ;*      .BYTE   M               ;;M=1 OR 0
 (1)                                    ;*                                      ;;1=TYPE LEADING ZEROS
 (1)                                    ;*                                      ;;0=SUPPRESS LEADING ZEROS
 (1)                                    ;*
 (1)                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 (1)                                    ;*$TYPOS OR $TYPOC
 (1)                                    ;*CALL:
 (1)                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                    ;*      TYPON                   ;;CALL FOR TYPEOUT
 (1)                                    ;*
 (1)                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 (1)                                    ;*CALL:
 (1)                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                    ;*      TYPOC                   ;;CALL FOR TYPEOUT
 (1)
 (1) 020652 017646 000000               $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 (1) 020656 116637 000001 021075                MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
 (1) 020664 112637 021077                       MOVB    (SP)+,$0MODE+1  ;;NUMBER OF DIGITS TO TYPE
 (1) 020670 062716 000002                       ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 (1) 020674 000406                              BR      $TYPON
 (1) 020676 112737 000001 021075        $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
 (1) 020704 112737 000006 021077                MOVB    #6,$0MODE+1     ;;SET FOR SIX(6) DIGITS
 (1) 020712 112737 000005 021074        $TYPON: MOVB    #5,$0CNT        ;;SET THE ITERATION COUNT
 (1) 020720 010346                              MOV     R3,-(SP)        ;;SAVE R3
 (1) 020722 010446                              MOV     R4,-(SP)        ;;SAVE R4
 (1) 020724 010546                              MOV     R5,-(SP)        ;;SAVE R5
 (1) 020726 113704 021077                       MOVB    $0MODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 (1) 020732 005404                              NEG     R4
 (1) 020734 062704 000006                       ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 (1) 020740 110437 021076                       MOVB    R4,$0MODE       ;;SAVE IT FOR USE
 (1) 020744 113704 021075                       MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
 (1) 020750 016605 000012                       MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 (1) 020754 005003                              CLR     R3              ;;CLEAR THE OUTPUT WORD
 (1) 020756 006105                      1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 (1) 020760 000404                              BR      3$              ;;GO DO MSB
 (1) 020762 006105                      2$:     ROL     R5              ;;FORM THIS DIGIT
 (1) 020764 006105                              ROL     R5
 (1) 020766 006105                              ROL     R5
 (1) 020770 010503                              MOV     R5,R3
 (1) 020772 006103                      3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 (1) 020774 105337 021076                       DECB    $0MODE          ;;TYPE THIS DIGIT?
 (1) 021000 100016                              BPL     7$              ;;BR IF NO
 (1) 021002 042703 177770                       BIC     #177770,R3      ;;GET RID OF JUNK
 (1) 021006 001002                              BNE     4$              ;;TEST FOR 0
 (1) 021010 005704                              TST     R4              ;;SUPPRESS THIS 0?
 (1) 021012 001403                              BEQ     5$              ;;BR IF YES
 (1)  021014  005204                    4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
 (1)  021016  052703  000060                    BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
 (1)  021022  052703  000040            5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
 (1)  021026  110337  021072                    MOVB    R3,8$                   ;;SAVE FOR TYPING
 (1)  021032  104401  021072                    TYPE    ,8$                     ;;GO TYPE THIS DIGIT
 (1)  021036  105337  021074            7$:     DECB    $OCNT                   ;;COUNT BY 1
 (1)  021042  003347                            BGT     2$                      ;;BR IF MORE TO DO
 (1)  021044  002402                            BLT     6$                      ;;BR IF DONE
 (1)  021046  005204                            INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
 (1)  021050  000744                            BR      2$                      ;;GO DO THE LAST DIGIT
 (1)  021052  012605                    6$:     MOV     (SP)+,R5                ;;RESTORE R5
 (1)  021054  012604                            MOV     (SP)+,R4                ;;RESTORE R4
 (1)  021056  012603                            MOV     (SP)+,R3                ;;RESTORE R3
 (1)  021060  016666  000002  000004            MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
 (1)  021066  012616                            MOV     (SP)+,(SP)
 (1)  021070  000002                            RTI                             ;;RETURN
 (1)  021072     000                    8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
 (1)  021073     000                            .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
 (1)  021074     000                    $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
 (1)  021075     000                    $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
 (1)  021076  000000                    $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
2612                                    .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 (1)
 (2)                                    ;;*************************************************************
 (1)                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 (1)                                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 (1)                                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 (1)                                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 (1)                                    ;*REPLACED WITH SPACES.
 (1)                                    ;*CALL:
 (1)                                    ;*      MOV     NUM,-(SP)               ;;PUT THE BINARY NUMBER ON THE STACK
 (1)                                    ;*      TYPDS                           ;;GO TO THE ROUTINE
 (1)
 (1)  021100                            $TYPDS:
 (3)  021100  010046                            MOV     R0,-(SP)                ;;PUSH R0 ON STACK
 (3)  021102  010146                            MOV     R1,-(SP)                ;;PUSH R1 ON STACK
 (3)  021104  010246                            MOV     R2,-(SP)                ;;PUSH R2 ON STACK
 (3)  021106  010346                            MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 (3)  021110  010546                            MOV     R5,-(SP)                ;;PUSH R5 ON STACK
 (1)  021112  012746  020200                    MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
 (1)  021116  016605  000020                    MOV     20(SP),R5               ;;GET THE INPUT NUMBER
 (1)  021122  100004                            BPL     1$                      ;;BR IF INPUT IS POS.
 (1)  021124  005405                            NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
 (1)  021126  112766  000055  000001            MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
 (1)  021134  005000                    1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
 (1)  021136  012703  021314                    MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
 (1)  021142  112723  000040                    MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
 (1)  021146  005002                    2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
 (1)  021150  016001  021304                    MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
 (1)  021154  160105                    3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
 (1)  021156  002402                            BLT     4$                      ;;BR IF DONE
 (1)  021160  005202                            INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
 (1)  021162  000774                            BR      3$
 (1)  021164  060105                    4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
 (1)  021166  005702                            TST     R2                      ;;CHECK IF BCD DIGIT=0
 (1)  021170  001002                            BNE     5$                      ;;FALL THROUGH IF 0
 (1)  021172  105716                            TSTB    (SP)              ;;STILL DOING LEADING 0'S?
 (1)  021174  100407                            BMI     7$                ;;BR IF YES
 (1)  021176  106316                    5$:     ASLB    (SP)              ;;MSD?
 (1)  021200  103003                            BCC     6$                ;;BR IF NO
 (1)  021202  116663  000001  177777            MOVB    1(SP),-1(R3)      ;;YES--SET THE SIGN
 (1)  021210  052702  000060            6$:     BIS     #'0,R2            ;;MAKE THE BCD DIGIT ASCII
 (1)  021214  052702  000040            7$:     BIS     #' ,R2            ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 (1)  021220  110223                            MOVB    R2,(R3)+          ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 (1)  021222  005720                            TST     (R0)+             ;;JUST INCREMENTING
 (1)  021224  020027  000010                    CMP     R0,#10            ;;CHECK THE TABLE INDEX
 (1)  021230  002746                            BLT     2$                ;;GO DO THE NEXT DIGIT
 (1)  021232  003002                            BGT     8$                ;;GO TO EXIT
 (1)  021234  010502                            MOV     R5,R2             ;;GET THE LSD
 (1)  021236  000764                            BR      6$                ;;GO CHANGE TO ASCII
 (1)  021240  105726                    8$:     TSTB    (SP)+             ;;WAS THE LSD THE FIRST NON-ZERO?
 (1)  021242  100003                            BPL     9$                ;;BR IF NO
 (1)  021244  116663  177777  177776            MOVB    -1(SP),-2(R3)     ;;YES--SET THE SIGN FOR TYPING
 (1)  021252  105013                    9$:     CLRB    (R3)              ;;SET THE TERMINATOR
 (3)  021254  012605                            MOV     (SP)+,R5          ;;POP STACK INTO R5
 (3)  021256  012603                            MOV     (SP)+,R3          ;;POP STACK INTO R3
 (3)  021260  012602                            MOV     (SP)+,R2          ;;POP STACK INTO R2
 (3)  021262  012601                            MOV     (SP)+,R1          ;;POP STACK INTO R1
 (3)  021264  012600                            MOV     (SP)+,R0          ;;POP STACK INTO R0
 (1)  021266  104401  021314                    TYPE    ,$DBLK            ;;NOW TYPE THE NUMBER
 (1)  021272  016666  000002  000004            MOV     2(SP),4(SP)       ;;ADJUST THE STACK
 (1)  021300  012616                            MOV     (SP)+,(SP)
 (1)  021302  000002                            RTI                       ;;RETURN TO USER
 (1)  021304  023420                    $DTBL:  10000.
 (1)  021306  001750                            1000.
 (1)  021310  000144                            100.
 (1)  021312  000012                            10.
 (1)  021314  000004                    $DBLK:  .BLKW   4
 2614                                  .SBTTL  TRAP DECODER
  (1)
  (2)                                  ;;*****************************************************************
  (1)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
  (1)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
  (1)                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
  (1)                                  ;*GO TO THAT ROUTINE.
  (1)
  (1)  021324  010046                  $TRAP:  MOV     R0,-(SP)          ;;SAVE R0
  (1)  021326  016600  000002                  MOV     2(SP),R0          ;;GET TRAP ADDRESS
  (1)  021332  005740                          TST     -(R0)             ;;BACKUP BY 2
  (1)  021334  111000                          MOVB    (R0),R0           ;;GET RIGHT BYTE OF TRAP
  (1)  021336  006300                          ASL     R0                ;;POSITION FOR INDEXING
  (1)  021340  016000  021360                  MOV     $TRPAD(R0),R0     ;;INDEX TO TABLE
  (1)  021344  000200                          RTS     R0                ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
  (1)
  (1)  021346  011646                  $TRAP2: MOV     (SP),-(SP)        ;;MOVE THE PC DOWN
  (1)  021350  016666  000004  000002          MOV     4(SP),2(SP)       ;;MOVE THE PSW DOWN
  (1)  021356  000002                          RTI                       ;;RESTORE THE PSW
  (1)
  (3)                                  .SBTTL  TRAP TABLE
  (3)
  (3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
  (3)                                  ;*BY THE "TRAP" INSTRUCTION.
  (3)
  (3)                                  ;       ROUTINE
  (3)                                  ;       -------
  (3)  021360  021346                  $TRPAD: .WORD   $TRAP2
  (3)  021362  016034                          $TYPE   ;;CALL=TYPE       TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)  021364  020676                          $TYPOC  ;;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)  021366  020652                          $TYPOS  ;;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)  021370  020712                          $TYPON  ;;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)  021372  021100                          $TYPDS  ;;CALL=TYPDS      TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (1)
  (1)
  (3)  021374  014632                          $RDCHR  ;;CALL=RDCHR      TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
  (3)  021376  014752                          $RDLIN  ;;CALL=RDLIN      TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
  (3)  021400  015124                          $RDOCT  ;;CALL=RDOCT      TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
 2615                                  .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                  ;;*****************************************************************
  (1)                                  ;POWER DOWN ROUTINE
  (1)  021402  012737  021546  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  021410  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)  021416  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
  (3)  021420  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
  (3)  021422  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
  (3)  021424  010346                          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
  (3)  021426  010446                          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
  (3)  021430  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
  (3)  021432  017746  157502                  MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
  (1)  021436  010637  021552                  MOV     SP,$SAVR6         ;;SAVE SP
  (1)  021442  012737  021454  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
 (1)  021450  000000                                    HALT
 (1)  021452  000776                                    BR      .-2             ;;HANG UP
 (1)
 (2)                                            ;;*************************************************************
 (1)                                            ;POWER UP ROUTINE
 (1)  021454  012737  021546  000024    $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
 (1)  021462  013706  021552                    MOV     $SAVR6,SP       ;;GET SP
 (1)  021466  005037  021552                    CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
 (1)  021472  005237  021552            1$:     INC     $SAVR6          ;;WAIT FOR THE INC
 (1)  021476  001375                            BNE     1$              ;;OF  WORD
 (3)  021500  012677  157434                    MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
 (3)  021504  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
 (3)  021506  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
 (3)  021510  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
 (3)  021512  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
 (3)  021514  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
 (3)  021516  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
 (1)  021520  012737  021402  000024            MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
 (1)  021526  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
 (1)  021534  104401                            TYPE                    ;;REPORT THE POWER FAILURE
 (1)  021536  021554                    $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
 (1)  021540  012716                            MOV     (PC)+,(SP)      ;;RESTART AT BEG2
 (1)  021542  002402                    $PWRAD: .WORD   BEG2            ;;RESTART ADDRESS
 (1)  021544  000002                            RTI
 (1)  021546  000000                    $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 (1)  021550  000776                            BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
 (1)  021552  000000                    $SAVR6: 0                       ;;PUT THE SP HERE
      021554  005015  042522  052123    PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
      021562  051101  044524  043516
      021570  040440  052106  051105
      021576  040440  050040  053517
      021604  051105  043040  044501
      021612  052514  042522  005015
      021620     000
              021622                    .EVEN
      021622  000310                    DIST:   .BLKW   200.            ;STATE-WIDTH DISTRIBUTION

              042000                            .=42000
                                        ;THE MICRO-CODE FOR THIS PROGRAM RESIDES HERE.
              042300                            .=42300

      042300  010000                    BUFFER: .BLKW   4096.           ;BUFFER AREA

              000001                    .END
```

```
ABASE = 170400          1165#   1223    1264    1265
ACDW1 = 000000          1223
ACDW2 = 000000          1223
ACPUOP= 000000          1223
ADBUFF  001320          1265#   1484    1492    1510    1545    1551    1596    1608    1729    1784*   1792*   1793*   1861
                        1911    1923    1962    2345    2366
ADDW0 = 000000          1223
ADDW1 = 000000          1223
ADDW10= 000000          1223
ADDW11= 000000          1223
ADDW12= 000000          1223
ADDW13= 000000          1223
ADDW14= 000000          1223
ADDW15= 000000          1223
ADDW2 = 000000          1223
ADDW3 = 000000          1223
ADDW4 = 000000          1223
ADDW5 = 000000          1223
ADDW6 = 000000          1223
ADDW7 = 000000          1223
ADDW8 = 000000          1223
ADDW9 = 000000          1223
ADEVCT= 000000          1223
ADEVM = 000000          1223
AENV  = 000000          1223
AENVM = 000000          1223
AFATAL= 000000          1223
AGAIN   007110          2023#   2055
AGATST  011772          2504#   2508
AGTST   012004          1750*   1767*   1777*   2505    2506#
AMADR1= 000000          1223
AMADR2= 000000          1223
AMADR3= 000000          1223
AMADR4= 000000          1223
AMAMS1= 000000          1223
AMAMS2= 000000          1223
AMAMS3= 000000          1223
AMAMS4= 000000          1223
AMSG    012241          1325    2516#
AMSGAD= 000000          1223
AMSGLG= 000000          1223
AMSGTY= 000000          1223
AMTYP1= 000000          1223
AMTYP2= 000000          1223
AMTYP3= 000000          1223
AMTYP4= 000000          1223
APASS = 000000          1223
APRIOR= 000300          1167#   1223    1266
APTCSU= 000040          2608    2609#
APTENV= 000001          2605    2608    2609#
APTSIZ= 000200          1348    2609#
APTSPO= 000100          2608    2609#
AROUND  007250          2051    2054#
ASKCH   013634          1822    2559#
ASWREG= 000000          1223
ATESTN= 000000          1223
```

```
ATMSG    012366             1857     2526#
AUNIT  = 000000             1223
AUSWR  = 000000             1223
AVECT1= 140340              1166#    1223     1267     1305
AVECT2= 000000              1223
BASEBR   001322             1266#    1796*    1797*
BASECH   001332             1270#    1538     1656     1657     1859     1888     1889     2014     2306     2341     2459
BEGIN    001712             1215     1344#    2506
BEGINA   005160             1392     1754#
BEGINC   004664             1395     1711#
BEGINL   002716             1445#    1747     1757
BEGINN   005526             1401     1820#
BEGINS   005576             1404     1831#
BEGINW   005246             1407     1771#
BEGIN2   001720             1217     1346#
BEGL     005122             1383     1398     1745#
BEG2     002402             1216     1322     1380#    2615
BITPNT   001412             1294#    1956*    1961     1987     1988*
BIT0   = 000001             1163#    1487     1497     1502
BIT00  = 000001             1163#
BIT01  = 000002             1163#
BIT02  = 000004             1163#
BIT03  = 000010             1163#
BIT04  = 000020             1163#
BIT05  = 000040             1163#
BIT06  = 000100             1163#
BIT07  = 000200             1163#
BIT08  = 000400             1163#    2604
BIT09  = 001000             1163#    2604     2605
BIT1   = 000002             1163#    2610
BIT10  = 002000             1163#    2605
BIT11  = 004000             1163#    1950     2604     2610
BIT12  = 010000             1163#    2244
BIT13  = 020000             1163#    1717     1730     2605
BIT14  = 040000             1163#    1452     2604     2610
BIT15  = 100000             1163#    1471     1481     1501     2487     2610
BIT2   = 000004             1163#
BIT3   = 000010             1163#
BIT4   = 000020             1163#    1467     1540     1543     2610
BIT5   = 000040             1163#    1462     2610
BIT6   = 000100             1163#    1457
BIT7   = 000200             1163#    1478     1501     1543     2358
BIT8   = 000400             1163#    1448
BIT9   = 001000             1163#    1476     1504     1542
BK3      005630             1838#    1849
BPTVEC= 000014              1163#
BUFFER   042300             1997     2040*    2042*    2058     2086     2152     2164     2166     2186     2198     2199     2200     2222
                            2225     2624#
BUFF1    013755             2134     2568#
BUFF2    014003             2221     2569#
BUMPAD   005310             1748     1765     1775     1781#
BYPASS   005462             1789     1803#
B10      011422             2410#    2411
B11      011456             2420#    2421
CH       012321             1719     2522#
CHAN     012403             2277     2528#
```

```
                                                                       M  7
LPA-AD11K TEST MD-11-CRLPKC      MACY11 30G(1063)  24-OCT-80  09:48  PAGE 45-2
CRLPKC.P11     14-AUG-80 13:59           CROSS REFERENCE TABLE -- USER SYMBOLS                                    SEQ 0090

CHANL    001362             1282#   1656*   1839*   1912    1967    2250*   2251    2278    2305*   2306*   2307    2342*   2593
CHANNL   007030             2013#
CH1      001350             1277#   1823*   1824*   1827    1834*   1856    1858    1880    1886*   1888*
CH2      001352             1278#   1837*   1839    1852    1876    1887*   1889*
CLEAR1   006756             1999#   2001
CLEAR2   007022             2010#   2012
CLKINT   020336             2610#
CMSG     012246             1319    2517#
CNNO     006734             1949*   1953*   1966    1992#
COMPAR   011302             1532    1556    1564    1572    1581    1589    1601    1624    1636    1644    1651    1673    2379#
CONV     006266             1920#   1925
CONVR    007136             2027    2030#
CONVRT   011060             1530    1554    1562    1570    1579    1587    1598    1610    1634    1642    1649    1862    2340#
CR     = 000015             1163#   2608
CRLF   = 000200             1163#   2608
C0       013673             1356    2560#
C1       013676             1937    2561#
C2       013700             2143    2227    2562#
C3       013703             2224    2563#
DAC      001404             1291#   1661    1664    1879    1883    1957*   1961*   1962    1987*   2293    2296    2299    2302
DASH     012275             2089    2520#
DAWAIT   004654             1597    1609    1633    1705#
DDISP  = 177570             1163#   1223    1348
DECPNT   014564             2094*   2123*   2587#
DECTYP   011472             1671    1850    2090    2096    2104    2108    2111    2122    2173    2271    2274    2429#
DELAY    001406             1292#   1963*   1964*
DELAY1   007216             2036    2044#
DELAY2   007224             2038    2045    2046#
DELAY3   007132             2028#   2029
DELCLR   010330             2132    2144    2228    2238#
DF1      014630             1240    1247    1253    1259    2596#
DH1      014405             1238    2582#
DH2      014443             1257    2583#
DH3      014526             1245    1251    2584#
DIFLIN   006736             1702    1995#
DISPLA   001142             1223#   1348*   1732*   2604*   2605*
DISPRE   000174             1215#   1348
DIST     021622             2002    2069*   2135    2618#
DONE     012420             2529#
DRLPX2=  ****** G             52#   2610
DSWR   = 177570             1163#   1223    1348
DT1      014570             1239    2592#
DT2      014602             1258    2593#
DT3      014620             1246    1252    2594#
DUMMY    001360             1281#   1657*   1876*   1880*   1969    2251*   2307*
EDGE     001410             1293#   1658*   1894*   1897*   1918*   1923*   1926*   1927*   1928*   1929*   1932    1980    2313*
                            2316*
EDGFLG   006436             1847*   1902*   1934    1940#   2254*   2321*
EMTVEC=  000030             1163#   1348*
EM1      014245             1237    2578#
EM2      014273             1244    2579#
EM3      014323             1250    2580#
EM4      014354             1256    2581#
ER       010552             2280    2282    2285#
ERMSG    012507             1678    1870    2100    2115    2127    2181    2285    2534#
ERR      006040             1866    1870#
```

```
ERRVEC= 000004           1163#   1348*   2604*
FIRST   001342           1274#   2008*   2081    2083*
FIXADR  005360           1782    1790#
FIXONE  005364           1372    1791#
FLAG    001400           1289#   1351*   1368*   2130    2184
GETDAT  010106           2189#   2197
GETEDG  006214           1840    1890    1911#   2252    2308
GMSG    012253           1333    2518#
GNS   = ****** U         1215    2610    2614
HAFMSG  012677           2124    2539#
HALF    007566           2116    2118#
HEAD1   014036           1371    2571#
HEAD5   013615           1713    2558#
HT    = 000011           1163#   2608
HUNS    014563           2438*   2450*   2452*   2455    2586#
INRNGE  007312           2065    2068#
IOTVEC= 000020           1163#   1348*
ISERV   001546           1314#   1374
KBVECT  001334           1271#   1373    1805
KMAD0   001434           1305#   1350    2610*
KMAD1   001436           1305#   1350
KMAD2   001440           1305#   1973*   1974    1977*   1978    2032*   2033    2610*
KMAD3   001442           1305#   2610*
KMAD4   001444           1305#   1972*   1976*   1980    2031*   2035    2610*
KMAD5   001446           1305#
KMAD6   001450           1305#   2610*
KMAD7   001452           1305#   1350    2610*
LAST    007430           2079    2092#
LEND    004652           1701    1703#
LESS    011410           2405    2407#
LF    = 000012           1163#   2608
LINEA   013574           2174    2557#
LOAD    010306           2223    2226    2230#
LOADY   011370           2138    2140    2233    2401#
LOADO   010312           2231#   2235
LO2     010304           2185    2229#
LPADH   001446           1305#
LPADL   001444           1305#
LPCI    001434           1305#
LPCO    001440           1305#
LPMR    001436           1305#
LPMS1   001450           1305#
LPMS2   001452           1305#
LPSO    001442           1305#
LSB     012331           1851    2524#
LSBMSG  012266           2091    2519#
MAT     013726           1853    2566#
MAX     001420           1297#   1605*   1612    1616*   2205*   2210    2212*   2216
MAXTST  010204           2208    2210#
MEND    012777           1758    2542#
MESP    012761           2275    2541#
MESR    012746           2272    2540#
METST   013733           2567#
MIN     001414           1295#   1617*   1666    2204*   2207    2209*   2215
MINUS   012235           2431    2514#
MLSB    013720           1672    2565#
```

```
MOFSET  013705          1669    2564#
MSG16   013154          2133    2551#
MSG18   013062          2220    2545#
MSG20   013122          1995    2550#
MSG21   013547          2151    2556#
MSG50   013041          1429    2544#
MSG71   013501          1384    2555#
MYTEMP  001426          1300#   1477*   1481*   1484    1487*   1488    1490    1492    1497*   1498    1499    1502*   1503
                        1510    1540*   1541    1544*   1545    1551    1552*   1595*   1596    1607*   1608    1712*   1723*
                        1725*   1727    1729    1860*   1861    1914*   1920*   1921    1923    2018*   2345    2346*   2352
                        2353*   2354    2359    2364    2366    2367
NAR     007476          2101    2103#
NARMSG  012546          2105    2536#
NARROW  001340          1273#   2007*   2073*   2103    2118
NBEXT   001354          1279#   1417*   1423*   1428    1430*   1431    1781    1788*   1801*
NEXT    007114          2024#   2050
NMBEXT  001356          1280#   1431*   1801
NOI     012373          2263    2527#
NOIA    010560          2253    2289#
NOIMSG  012130          1821    2512#
NOITST  010366          1826    2250#
NOI1    010570          2291#   2312
NOI8    010652          1684    1686    1688    2305#
NONE    001704          1330    1332    1340#
NOTNAR  007346          2072    2078#
NOTNEW  007776          2161    2166#
NOTOK   007242          2048    2052#
NOVT55  002330          1359    1362    1367    1371#   1440
NXTCMP  010170          2206#   2214
NXTCVT  006622          1971#   1984
NXTSTA  007740          2153#   2167
NXTY1   007674          2137#   2142
NXT8    010152          2202#   2219
OFFERR  004520          1676    1678#
OFFOK   004526          1677    1680#
OKAY    007234          2041    2043    2049#   2053
OKAYD   011516          2434    2436#
OKMSG   012476          1680    1867    2102    2117    2129    2183    2283    2532#
ONAD    013023          2543#
ONES    014566          2436*   2442*   2443    2445*   2454*   2589#
OUT     001424          1299#   1996*   2005*   2066*   2107    2110    2113    2120
OUTMSG  012644          2112    2538#
PEAK    001376          1288#   2267    2269*   2273    2281    2290*   2299*   2302*   2310*   2319*   2320*
PERCNT  001422          1298#   1946*   1947*   1948*   1954*   1955*   1985
PIRQ  = 177772          1163#
PIRQVE= 000240          1163#
PLUS    007760          2158    2160#
PLUSR2  011376          2402    2404#
POPRO   001700          1338#   1341
POS     011504          2430    2433#
POSPEA  010454          2268    2270#
POSRMS  010442          2265    2267#
POSR2   005666          1844    1846#
PR0   = 000000          1163#
PR1   = 000040          1163#
PR2   = 000100          1163#
```

```
PR3    = 000140          1163#
PR4    = 000200          1163#
PR5    = 000240          1163#
PR6    = 000300          1163#
PR7    = 000340          1163#
PS     = 177776          1163#   2610*
PSW    = 177776          1163#   1387*   1714*   2395*
PWRMSG  021554           2615    2616#
PWRVEC= 000024           1163#   1348*   2615*
QUEST   012237           1340    1408    2515#
RANDY   010776           2024    2328#
RBEG    001726           1345    1347#
RDCHR = 104406           2600    2614#
RDLIN = 104407           1385    2602    2614#
RDOCT = 104410           1336    1833    1836    2610    2614#
RD1     017672           2610#*
READ    007264           2059#   2093
RELACC  007724           2131    2149#
REST1   002334           1353    1372#
RESVEC= 000010           1163#
RET     006434           1935    1939#
RETERR  003442           1520    1522#
RETURN  001702           1339#   2022    2350
RMS     001374           1287#   2264    2266*   2270    2279    2289*   2293*   2296*   2309*   2317*   2318*
RNA     001366           1284#   1376*   2025    2328*   2329*   2330*   2331    2334
RNB     001370           1285#   1377*   2328    2331*   2332*   2333*   2335
RNC     001372           1286#   1378*   2329    2332    2334*   2335*   2336*
RST     011350           1321    1327    2393#
RTCCSR  020344           2610#*
R2POS   006176           1899    1901#
SARSUB  006440           1659    1662    1877    1881    1946#   2291    2294    2297    2300
SAR1    006512           1951    1952    1956#
SDELAY  020262           2610#
SETAA   006146           1893#   1896
SETCH   012344           1855    2525#
SETMSG  012150           1832    2513#
SET1A   006046           1842    1843    1876#   1893
SET8    006110           1692    1696    1886#
SHIFT   006724           1986    1988#
SKIPST  001344           1275#   2009*   2076*   2095    2098
SKPMSG  012524           2097    2535#
SLASH   012433           1335    2176    2530#
SPACE   012324           1734    2523#
SPREAD  001402           1290#   2380*   2387    2593
STACK = 001100           1163#   1320    1326    1348    1380
STATE   012301           2084    2521#
STKLMT= 177774           1163#
STREG   001316           1264#   1477    1481    1488    1490    1498    1499    1503    1517    1518    1541    1552    1712
                         1723    1725    1727    1759    1783*   1791*   1914    1920    1921    1959    2017    2018    2346
                         2352    2354    2364    2592    2593    2594
SWDIST  007642           2128    2130#
SWR     001140           1223#   1331    1334    1337*   1348*   1715    1717    1730    1823    1950    2244    2604    2605
                         2615*
SWREG   000176           1215#   1348
SW0   = 000001           1163#
SW00  = 000001           1163#
```

```
SW01   = 000002          1163#
SW02   = 000004          1163#
SW03   = 000010          1163#
SW04   = 000020          1163#
SW05   = 000040          1163#
SW06   = 000100          1163#
SW07   = 000200          1163#
SW08   = 000400          1163#
SW09   = 001000          1163#
SW1    = 000002          1163#
SW10   = 002000          1163#
SW11   = 004000          1163#
SW12   = 010000          1163#
SW13   = 020000          1163#
SW14   = 040000          1163#
SW15   = 100000          1163#
SW2    = 000004          1163#
SW3    = 000010          1163#
SW4    = 000020          1163#
SW5    = 000040          1163#
SW6    = 000100          1163#
SW7    = 000200          1163#
SW8    = 000400          1163#
SW9    = 001000          1163#
TADDR    001364          1283#    1328     1711*    1745*    1754*    1771*    1820*    1831*
TBITVE=  000014          1163#
TEMP     001346          1276#    1600     1614     1620     1621     1622*    1661*    1664*    1665*    1668*    1670     1683*    1695*
                         1825*    1838*    1864     1911     2047*    2052*    2344*    2367*    2372*    2373*    2374*    2375*    2381
TENS     014565          2437*    2446*    2447     2449*    2453*    2588#
TEST     003416          1482     1493     1507     1518#    1549
TESTAD   002550          1414#    1746     1756     1773
TESTIT   003406          1449     1458     1463     1468     1472     1517#
TESTR2   011532          2439#    2444     2448     2451
TIME     020334          2610#*
TKVEC  = 000060          1163#
TOMSG    012435          1835     2531#
TPVEC  = 000064          1163#
TRAPVE=  000034          1163#    1348*
TRTVEC=  000014          1163#
TRY      006532          1960#    1989
TRYAG    002424          1385#    1409
TST1     002716          1446#    1447
TST10    003260          1496#
TST11    003444          1526#
TST12    003516          1537#
TST13    003644          1539     1553#
TST14    003674          1561#
TST15    003724          1569#
TST16    003754          1578#
TST17    004004          1586#
TST2     002762          1455#
TST20    004034          1594#
TST21    004302          1641#
TST22    004332          1648#
TST23    004362          1655#
TST24    004532          1679     1682#
```

```
TST25    004572        1691#
TST26    004630        1699#
TST3     003006        1461#
TST4     003024        1466#
TST5     003042        1470#
TST6     003060        1475#
TST7     003174        1486#
TST8     010216        2211     2213#
TYPBAD   007360        2067     2077     2081#
TYPDS  = 104405        2508     2614#
TYPE   = 104401        1319     1325     1333     1335     1340     1356     1371     1384     1408     1429     1669     1672     1678
                       1680     1713     1719     1734     1741     1758     1763     1821     1822     1832     1835     1851     1853
                       1855     1857     1867     1870     1937     1995     2084     2089     2091     2097     2100     2102     2105
                       2109     2112     2115     2117     2124     2127     2129     2133     2134     2143     2151     2174     2176
                       2181     2183     2220     2221     2224     2227     2247     2263     2272     2275     2277     2283     2285
                       2431     2455     2508     2600     2605     2606     2608     2610     2611     2612     2614#    2615
TYPEDG   006374        1854     1932#    2276
TYPOC  = 104402        2606     2610     2614#
TYPON  = 104404        2614#
TYPOS  = 104403        1334     1428     1720     1735     1760     1852     1856     1864     1933     1938     2088     2175     2178
                       2278     2610     2614#
TYPOUT   011606        2440     2452#
TYPRP    010424        2255     2263#    2322
TYPSET   005704        1848     1850#    1903
UNEXP    001524        1307#    1456
VADR     001326        1268#    1424     1783     1784
VARLT1   011742        2465     2489#
VARLT2   011752        2467     2494#
VARLT3   011762        2461     2499#
VECTOR   001454        1305#    1456*    1785*    1794*    1795*    1798     2022*    2350*
VECTPS   001456        1305#
VECTR1   001324        1267#    1786*    1787*    1798*    1799*    1800*
VERSN    001460        1305#
VLIN     011736        2179     2486#
VNP      011732        2281     2484#
VNR      011730        2279     2458     2483#
VSET     011734        1865     2485#
VTFLG    002664        1357     1360     1363     1434#
VTINIT   014023        2247     2570#
VT55     002324        1365     1368#
VVCT     001330        1269#    1785     1786
V1       011706        1558     2473#
V10      011712        1566     2475#
V115     011720        1603     2478#
V144     011716        1646     2477#
V2       011710        1638     2474#
V240     011722        1583     1591     1653     2479#
V5       011724        1626     2480#
V50      011714        1534     1574     2476#
V50D     011726        1675     2481#
WFADJ    011636        1379     2458#
WFTEST   001416        1296#    1344*    1346*    2463
WIDE     001336        1272#    2006*    2080*    2106     2119
WIDMSG   012605        2109     2537#
WRAP     003444        1525#    1764     1774
W1       017522        2610#*
```

```
W2       017524            2610#*
W3       017526            2610#*
$AERR    017252            1420     2610#*
$APTHD   001000            1222#
$ASTAT= ****** U           2609
$ATYC    016342            2609#
$ATY1    016316            2609#
$ATY3    016324            2608     2609#
$ATY4    016334            2605     2609#
$AUTOB   001134            1223#    1426
$BASE    001250            1223#    1414     1791     1792
$BDADR   001122            1223#
$BDDAT   001126            1223#    1414*    1419     1424*    1518     1519     2359*    2381*    2382     2592     2593     2594
$BELL    001164            1223#    2605
$CDW1    001254            1223#
$CHARC   016312            2608#*
$CKSWR= ****** U           2614
$CMTAG   001100            1223#    1348
$CM3   = 000000            1223#
$CNT     020222            2610#*
$CNTLG   015075            2600#
$CNTLU   015070            2600#
$CPUOP   001222            1223#
$CRLF    001171            1223#    1741     1763     2600     2605     2606     2608
$DATR    017674            2610#*
$DBLK    021314            2612#
$DEVCT   001204            1223#
$DEVM    001252            1223#
$DOAGN   012104            2508#
$DTBL    021304            2612#
$ENDAD   012074            1220     2508#    2605
$ENDCT   012042            1348     2508#
$ENDMG   012113            2508#
$ENULL   012110            2508#
$ENV     001214            1223#    2605     2608     2609
$ENVM    001215            1223#    1348     2608     2609
$EOP     012006            1751     1768     1778     2508#
$EOPCT   012034            1348*    2508#
$ERFLG   001103            1223#    1308*    2604*    2605*
$ERMAX   001115            1223#    1348*    2604*
$ERROR   015504            1348     2605#
$ERRPC   001116            1223#    2592     2593     2594     2605*    2606
$ERRTB   001256            1223#    2606
$ERRTY   015700            2605     2606#
$ERTTL   001112            1223#    2605*
$ESCAP   001162            1223#    1307*    1310*    1348*    2604*    2605
$ETABL   001214            1223#
$ETEND   001256            1222     1223#
$FATAL   001176            1223#    2609*
$FFLG    016562            2609#*
$FILLC   001156            1223#    2608
$FILLS   001155            1223#    2608
$GDADR   001120            1223#
$GDDAT   001124            1223#    1419     1448*    1451*    1452     1457*    1462*    1467*    1471*    1478*    1489*    1501*    1517
                           1519     1543*    2358*    2379*    2383     2592     2593
$GET42   012064            2508#
```

```
$GTSWR= ****** U         2614
$HD    = 000000          1162
$HIBTS   001000          1222#
$HIOCT   015224          2602#*
$ICNT    001104          1223#    2604*
$ILLUP   021546          2615#
$INLP    020070          1477     1481     1484     1490     1492     1499     1510     1518     1551     1725     1727     1729     1920
                         1921     1923     2345     2352     2364     2366     2610#
$INTAG   001135          1223#
$ITEMB   001114          1223#    2605*    2606
$LF      001172          1223#    2600     2605     2608
$LFLG    016561          2609#*
$LOAD    017254          2610#
$LPADR   001106          1223#    1348*    1446*    1529*    2604*
$LPAI    016564          2610#
$LPERR   001110          1223#    1348*    1447*    1528*    2604*    2605
$LPW     017676          2610#
$MADR1   001226          1223#
$MADR2   001232          1223#
$MADR3   001236          1223#
$MADR4   001242          1223#
$MAIL    001174          1222     1223#    1348     2604     2605     2608
$MAMS1   001224          1223#
$MAMS2   001230          1223#
$MAMS3   001234          1223#
$MAMS4   001240          1223#
$MBADR   001002          1222#
$MFLG    016560          2609#*
$MNEW    015113          2600#
$MSGAD   001210          1223#    2609*
$MSGLG   001212          1223#    2609*
$MSGTY   001174          1223#    2609*
$MSWR    015102          2600#
$MTYP1   001225          1223#
$MTYP2   001231          1223#
$MTYP3   001235          1223#
$MTYP4   001241          1223#
$MXCNT   015502          2604#
$NULL    001154          1223#    2608
$NWTST= 000001           1446#    1455#    1461#    1466#    1470#    1475#    1486#    1496#    1526#    1537#    1553#    1561#    1569#
                         1578#    1586#    1594#    1641#    1648#    1655#    1682#    1691#    1699#
$OCNT    021074          2611#*
$OFS   = 020132          2610#
$OMODE   021076          2611#*
$OUTLP   017772          1419     1477     1481     1488     1498     1503     1517     1541     1545     1552     1596     1608     1712
                         1723     1725     1861     1911     1914     1920     1962     2018     2346     2354     2610#
$OVER    015466          2604#
$PASS    001202          1223#    1348*    1700     1755*    1772*    2508*    2604
$PASTM   001006          1222#
$PUTS    020542          1958     2016     2610#
$PWRAD   021542          2615#
$PWRDN   021402          1348     2615#
$PWRMG   021536          2615#
$PWRUP   021454          2615#
$QUES    001170          1223#    2600     2605     2608
$RDCHR   014632          2600#    2614
```

```
                                                                            H  8
LPA-AD11K TEST MD-11-CRLPKC    MACY11 30G(1063)  24-OCT-80  09:48  PAGE 45-10
CRLPKC.P11    14-AUG-80 13:59         CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0098

$RDDEC= ****** U         2614
$RDLIN  014752           2600#   2614
$RDOCT  015124           2602#   2614
$RDSZ = 000010           2600#
$RESET  020224           1631    2393    2610#
$RTNAD  012106           2508#
$R2A  = ****** U         2614
$SAD    020220           2610#*
$SAVRE= ****** U         2614
$SAVR6  021552           2615#*
$SCOPE  015226           1348    2604#
$SETUP= 000037           1224#   1348    2508    2600    2604    2605
$STUP = 177777           1224#
$SVLAD  015432           2604#
$SVPC = 000220           1220#
$SWR  = 167400           1155#   1162    1164    1223    1348    1446    1455    1461    1466    1470    1475    1486    1496
                         1526    1537    1553    1561    1569    1578    1586    1594    1641    1648    1655    1682    1691
                         1699    2508    2604    2605    2615
$SWREG  001216           1223#   1348
$SWRMK= 000000           1164    2604
$TBF4   007136           2031#
$TEMP1  001430           1301#   1969*   1970*   1972    2019*   2020*   2031
$TEMP2  001432           1302#   1967*   1968*   1976
$TESTN  001200           1223#   2604*
$TIMES  001160           1223#   1348*   1496*   1526*   1537*   1553*   1561*   1569*   1578*   1586*   1594*   1641*   1648*
                         1655*   1682*   1691*   1699*   2508*   2604*
$TKB    001146           1223#   1315    1441    2600    2610
$TKS    001144           1223#   1355*   1386*   1435    1632*   2394*   2504*   2600    2610
$TLKR   017530           2610#
$TLKW   017414           2610#
$TN   = 000027           1156#   1162    1446#   1455#   1461#   1466#   1470#   1475#   1486#   1496#   1526#   1527    1537#
                         1539    1553#   1561#   1569#   1578#   1586#   1594#   1641#   1648#   1655#   1679    1682#   1691#
                         1699#
$TOUT   020164           2610#
$TPB    001152           1223#   2412*   2422*   2608*
$TPFLG  001157           1223#   2608
$TPS    001150           1223#   2410    2420    2608
$TRAP   021324           1348    2614#
$TRAP2  021346           2614#
$TRP  = 000011           2614#
$TRPAD  021360           2614#
$TSTM   001004           1222#
$TSTNM  001102           1223#   1527*   2508*   2604*   2605
$TTYIN  015060           2600#
$TYPBN= ****** U         2614
$TYPDS  021100           2612#   2614
$TYPE   016034           2608#   2609    2614
$TYPEC  016246           2608#
$TYPEX  016314           2608#
$TYPOC  020676           2611#   2614
$TYPON  020712           2611#   2614
$TYPOS  020652           2611#   2614
$T6MP   006622           1972#
$UNIT   001206           1223#
$UNITM  001010           1222#
$USWR   001220           1223#
```

```
$UTK     020346            1218     2610#
$VECT1   001244            1223#    1794     1796
$VECT2   001246            1223#
$XTSTR   015236            2604#
$$GET4=  000000            2508#
$OFILL   021075            2611#*
$4OCAT=  ****** U          2604     2605
.     =  062300            1215#    1220#    1221#    1222#    1223#    1305#    1348     2508     2600#    2604     2605     2606#    2608
                           2609#    2610#    2612#    2615     2617#    2618#    2620#    2622#    2624#
.DVLS    001462            1305#    1350*    1415*    1416*    2610*
.$ASTA=  ****** U          2609
.$X   =  001000            1222#
```

```
ADDM      1198#   1923
BICM      1186#   1481
CLRM      1181#   1552    1712
CMPM      1204#
COMMEN    1163#
DUMWRN    1228#
ENDCOM    1163#
ERROR     1163#   1309    1450    1459    1464    1469    1473    1483    1494    1508    1535    1550    1559    1567    1575
          1584    1592    1604    1627    1639    1647    1654    2361    2610
ESCAPE    1163#   1307
GETPRI    1163#
GETSWR    1163#
INCRM     1174#   1477    1725    1920
MOVEI      170#   1477    1481    1484    1490    1492    1499    1510    1518    1551    1725    1727    1729    1920    1921
          1923    2345    2352    2364    2366    2610
MOVEM      157#   1419    1477    1481    1488    1498    1503    1517    1541    1545    1552    1596    1608    1712    1723
          1725    1861    1911    1914    1920    1962    2018    2346    2354
MOVEMR    1210#   1723    1914    2018    2346
MOVERO    1169#   1484    1492    1510    1551    1729
MULT      1163#
NEWTST    1163#   1446    1455    1461    1466    1470    1475    1486    1496    1526    1537    1553    1561    1569    1578
          1586    1594    1641    1648    1655    1682    1691    1699
POP       1163#   2602    2609    2612    2615
PUSH      1163#   2602    2609    2612    2615
REPORT    1163#
SCOPE     1163#   1455    1461    1466    1470    1475    1486    1496    1512    1537    1553    1561    1569    1578    1586
          1594    1641    1648    1655    1682    1691    1699
SETPRI    1163#
SETTRA    2614#
SETUP     1163#   1348
SKIP      1163#   1391    1394    1397    1400    1403    1406    1425    1436    1438    1440    1453    1520    1539    1679
          1718    2245    2460    2462
SLASH     1163#
SPACE     1163#
STARS     1163#   1220    1222    1223    1446    1455    1461    1466    1470    1475    1486    1496    1526    1537    1553
          1561    1569    1578    1586    1594    1641    1648    1655    1682    1691    1699    2508    2600    2602    2604
          2605    2606    2608    2609    2611    2612    2614    2615
SWRSU     1163#   1348#
TOUT      1305#   2610
TRMTRP    2614#
TSTBM     1193#   1490    1499    1727    1921    2364
TYPBIN    1163#
TYPDEC    1163#   2508
TYPNAM    1163#
TYPNUM    1163#
TYPOCS    1163#   1334    1428    1720    1735    1852    1856    1864    1933    1938    2088    2175    2178    2278
TYPOCT    1163#   2606    2610
TYPTXT    1163#   2610
$CAL.      748#   2610
$DMAST     914#
$DMDT     1048#
$MMAST     772#
$$CMRE    1223#
$$CMTM    1223#
$$ESCA    1163#
$$NEWT    1163#   1446    1455    1461    1466    1470    1475    1486    1496    1526    1537    1553    1561    1569    1578
```

```
         1586    1594    1641    1648    1655    1682    1691    1699
$$SET    2614#
$$SETM   1348#
$$SKIP   1163#   1539    1679
.EQUAT   1157#   1163
.HEADE   1157#   1162
.KMADR     55#   1305
.KSIS     184#   1350
.LOADL    459#   2610
.LPAIN    209#   2610
.PUTCS    418#   2610
.RESET    329#   2610
.SETUP   1159#   1224
.SWRHI   1159#   1164
.SWRLO   1164#
.UTK      699#   2610
.$ACT1   1160#   1220
.$APTB   1160#   1223#
.$APTH   1160#   1222
.$APTY   1160#   2609
.$CATC   1157#   1215
.$CMTA   1157#   1223
.$EOP    1157#   2508
.$ERRO   1158#   2605
.$ERRT   1159#   2606
.$INLP    652#   2610
.$MMAC    141#
.$OUTL    610#   2610
.$PARM   1158#
.$POWE   1158#   2615
.$RAND   1160#
.$RDOC   1160#   2602
.$READ   1158#   2600
.$SAVE   1158#
.$SCOP   1158#   2604
.$SPAC   1159#
.$SWDO   1159#
.$TLKW    511#   2610
.$TOUT   1305#   2610
.$TRAP   1159#   2614
.$TYPD   1160#   2612
.$TYPE   1159#   2608
.$TYPO   1158#   2611


. ABS.  062300     000     CON    RW     ABS    GBL    D
        000000     001     CON    RW     REL    LCL    I


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CRLPKC,CRLPKC/CRF=CRLPAB.MAC,CRLPKC.P11
 RUN-TIME: 24 12 1 SECONDS
 RUN-TIME RATIO: 139/39=3.5
 CORE USED:  36K  (71 PAGES)
```

```
     1                                         ;THIS FILE IS THE SAME AS "CRLPX0.P11" EXCEPT IT IS LOADED INTO 65000
     2                                         ;IT IS ALSO THE SAME AS "DRLPX2.P11" EXCEPT NAME CHANGE "CRLPX2.P11"
     3
     4
     5                                                 .LIST   MC,BIN,BEX,MEB
     6                                                 .NLIST  MD,CND,ME
     7
     8          177777                                 ADDRESS=-1
     9                                         ;       MACRO DEFFINITIONS FOR M8200 AND M8204 MICRO-PROCESSOR
    10                                         ;       INSTRUCTION SET.
    11                                         ;       TO BE USED WITH RSX MACRO-11 ASSEMBLER
    12                                         ;
    13                                         ;       26-MAY-1976
   452  000000'                                        $BEGIN
   453  000000                                         $LOC    42000
   454                                                 .GLOBL  DRLPX2
   455                                                 .ENABL  GBL
   456
   457                                                 ;*
   458                                                 ;*MICRO CODE FOR KMC-11
```

```
                                      ;*THIS CODE WILL BE DOWN LOADED INTO BOTH
                                      ;*KMC-11'S. THE CODE RUNS ASYNCRONOUS TO THE PDP-11 CODE
                                      ;*WE SYNC THROUGH COMMANDS PASSED VIA THE OUT*/IBUS* REGS.
                                      ;*


      042000                 DRLPX2:  ;JUMP TABLE USED FOR COMMANDS
      042000                          BR      STARTU          ;GOTO START
(2)   042000  100407                  .WORD   .$$$.
      042002                          BR      CMNOP           ;NOP=1
(2)   042002  100420                  .WORD   .$$$.
      042004                          BR      RDSILO          ;=2 READ SILO PUT IN BSEL4
(2)   042004  100430                  .WORD   .$$$.
      042006                          BR      WRSILO          ;=3 READ BSEL4 PUT IN SILO.
(2)   042006  100432                  .WORD   .$$$.
      042010                          BR      RDCMND          ;=4 READ FAST PATH PUT IN BSEL4
(2)   042010  100434                  .WORD   .$$$.
      042012                          BR      WRCMND          ;=5 READ BSEL4, PUT IN FAST PATH.
(2)   042012  100436                  .WORD   .$$$.
      042014                          BR      SAMP            ;=6 TAKE AN A/D SAMPLE
(2)   042014  100440                  .WORD   .$$$.

                                      ;START OF U CODED

                                      .
      042016                 STARTU:
      042016                          MOVE    # 0,BREG
(3)   042016  000400                  .WORD   .$$$.
      042020                          MOVE    BREG,OUT1 <0>   ;CLEAR UNIBUS CSRS
(3)   042020  061220                  .WORD   .$$$.
      042022                          MOVE    BREG,OUT1 <2>
(3)   042022  061222                  .WORD   .$$$.
      042024                          MOVE    BREG,OUT1 <3>
(3)   042024  061223                  .WORD   .$$$.
      042026                          MOVE    BREG,OUT1 <4>
(3)   042026  061224                  .WORD   .$$$.
      042030                          MOVE    BREG,OUT1 <5>
(3)   042030  061225                  .WORD   .$$$.
      042032                          MOVE    BREG,OUT1 <6>
(3)   042032  061226                  .WORD   .$$$.
      042034                          MOVE    BREG,OUT1 <7>
(3)   042034  061227                  .WORD   .$$$.
      042036                          MOVE    BREG,SPAD <6>
(3)   042036  063226                  .WORD   .$$$.

      042040                 CMNOP:   MOVE    INPO <12>,OUT1 <0> ;READ STATUS
(3)   042040  021240                  .WORD   .$$$.
      042042                          MOVE    # 377,BREG
(3)   042042  000777                  .WORD   .$$$.
      042044                          MOVE    BREG,OUT1 <2>   ;INDICATE READY FOR COMMAND.
(3)   042044  061222                  .WORD   .$$$.

      042046                 LOOP:    MOVE    INPO <12>,OUT1 <0> ;READ STATUS
(3)   042046  021240                  .WORD   .$$$.

      042050                          MOVE    INP1 <2>,SPAD <0> ;READ COMMAND REG.
```

```
   (3)  042050  123040                        .WORD   .$$$.
  496   042052                                BZ      LOOP             ;NO COMMAND THEN LOOP
   (2)  042052  101423                        .WORD   .$$$.
  497
  498   042054                                MOVE    INP1 <2>,SPAD <0> ;RE-READ COMMAND.
   (3)  042054  123040                        .WORD   .$$$.
  499
  500   042056                                BR      SPAD <0>         ;BR BASED ON CMND.
   (2)  042056  160600                        .WORD   .$$$.
  501                                                                  ;NO-USER PROTECTION OFFERED.
  502                                                                  ;IF YOU ENTER WRONG CODE -
  503                                                                  ;YOU LOSE.
  504
  505                                         ;
  506                                         ;ROUTINE TO READ THE SILO, PUT IN
  507                                         ;*BUS REG 4
  508                                         ;CMD=2
  509                                         ;
  510   042060                        RDSILO: MOVE    INP0 <10>,OUT1 <4> ;READ SILO.
   (3)  042060  021204                        .WORD   .$$$.
  511                                                                  ;WRITE *BUS
  512   042062                                BR      CMNOP            ;RETURN.
   (2)  042062  100420                        .WORD   .$$$.
  513
  514                                         ;
  515                                         ;ROUTINE TO WRITE SILO, READ DATA FROM
  516                                         ;*BUS REG 4
  517                                         ;CMD=3
  518                                         ;
  519
  520   042064                        WRSILO: MOVE    INP1 <4>,OUT0 <10> ;READ DATA IN *BUS
   (3)  042064  122110                        .WORD   .$$$.
  521                                                                  ;WRITE SILO.
  522   042066                                BR      CMNOP
   (2)  042066  100420                        .WORD   .$$$.
  523
  524                                         ;
  525                                         ;ROUTINE TO READ FAST PATH (CMND) REG.
  526                                         ;PUT IN *BUS REG 4
  527                                         ;CMD=4
  528                                         ;
  529
  530   042070                        RDCMND: MOVE    INP0 <11>,OUT1 <4> ;READ FAST PATH
   (3)  042070  021224                        .WORD   .$$$.
  531                                                                  ;WRITE *BUS.
  532   042072                                BR      CMNOP            ;RETURN
   (2)  042072  100420                        .WORD   .$$$.
  533
  534                                         ;
  535                                         ;ROUTINE TO WRITE FAST PATH (CMND) REG.
  536                                         ;TAKE DATA FROM *BUS REG 4.
  537                                         ;CMD=5
  538                                         ;
  539
  540   042074                        WRCMND: MOVE    INP1 <4>,OUT0 <11> ;READ DATA IN *BUS
   (3)  042074  122111                        .WORD   .$$$.
  541                                                                     ;WRITE INTO FAST PATH.
  542  042076                                BR      CMNOP                ;RETURN.
  (2)  042076  100420                        .WORD   .$$$.
  543
  544
  545                                        ;
  546                                        ;THIS ROUTINE TAKES AN A/D SAMPLE.
  547                                        ;CALL= CMND 6 IN BSEL2
  548                                        ;THESE REGS. MUST BE SET UP IN ADVANCE.
  549                                        ;BSEL 3 MUST CONTAIN READ CODE FOR A/D BUFFER.
  550                                        ;BSEL 4,5 MUST CONTAIN  A/D CSR SETTING.
  551                                        ;BSEL 6 MUST CONTAIN WRITE CODE FOR A/D CSR
  552                                        ;BSEL 7 MUST CONTAIN READ CODE FOR A/D CSR
  553                                        ; BSEL 3,6,7 WILL REMAIN  UNEFFECTED.
  554                                        ; BSEL 4,5 WILL CONTAIN A/D SAMPLE.
  555                                        ;BSEL2 WILL CONTAIN CODE 377 WHEN DONE.
  556
  566
  567  042100                                WTMC    SAMP
  (4)  042100  020640                        .WORD   .$$$.
  (3)  042102  103040                        .WORD   .$$$.
  568  042104                                MOVE    INP1 <6>,OUTO <11>      ;SEND A/D WRITE CODE.
  (3)  042104  122151                        .WORD   .$$$.
  569  042106                                WTMC    SAMP1
  (4)  042106  020640                        .WORD   .$$$.
  (3)  042110  103043                        .WORD   .$$$.
  570  042112                                MOVE    INP1 <4>,OUTO <11>      ;SEND LOW BYTE CSR INFO.
  (3)  042112  122111                        .WORD   .$$$.
  571  042114                                WTMC    SAMP2
  (4)  042114  020640                        .WORD   .$$$.
  (3)  042116  103046                        .WORD   .$$$.
  572  042120                                MOVE    INP1 <5>,OUTO <11>      ;SEND HIGH BYTE CSR INFO.
  (3)  042120  122131                        .WORD   .$$$.
  573  042122                                WTMC    SLOOP
  (4)  042122  020640                        .WORD   .$$$.
  (3)  042124  103051                        .WORD   .$$$.
  574  042126                                MOVE    INP1 <7>,OUTO <11>      ;SEND READ CODE TO GET A/D CSR.
  (3)  042126  122171                        .WORD   .$$$.
  575  042130                                WTMC    SAMP3
  (4)  042130  020640                        .WORD   .$$$.
  (3)  042132  103054                        .WORD   .$$$.
  576  042134                                WTMM    SLOOP1
  (4)  042134  020640                        .WORD   .$$$.
  (4)  042136  061620                        .WORD   .$$$.
  (3)  042140  103056                        .WORD   .$$$.
  577  042142                                MOVE    INPO <11>,BREG
  (3)  042142  020620                        .WORD   .$$$.
  578  042144                                MOVE    BREG,SPAD <0>
  (3)  042144  063220                        .WORD   .$$$.
  579  042146                                WTMM    SLOOP2
  (4)  042146  020640                        .WORD   .$$$.
  (4)  042150  061620                        .WORD   .$$$.
  (3)  042152  103063                        .WORD   .$$$.
  580  042154                                MOVE    INPO <11>,BREG
  (3)  042154  020620                        .WORD   .$$$.
  581  042156                                BB7     CMNOP                   ;ABORT IF A/D BIT 15=1
```

```
(2)  042156  103420                        .WORD   .$$$.
582  042160                                MOVE    SPAD <0>,BREG
(3)  042160  060600                        .WORD   .$$$.
583  042162                                BB7     LOPE
(2)  042162  103473                        .WORD   .$$$.
584  042164                                BR      SLOOP                   ;IF A/D NOT DONE,EXIT.
(2)  042164  100451                        .WORD   .$$$.
585  042166                        LOPE:   MOVE    INP1 <3>,OUT0 <11>      ;ISSUE READ A/B BUFFER.
(3)  042166  122071                        .WORD   .$$$.
586  042170                                WTMM    SLOOP3
(4)  042170  020640                        .WORD   .$$$.
(4)  042172  061620                        .WORD   .$$$.
(3)  042174  103074                        .WORD   .$$$.
587  042176                                MOVE    INP0 <11>,OUT1 <4>
(3)  042176  021224                        .WORD   .$$$.
588  042200                                WTMM    SLOOP4
(4)  042200  020640                        .WORD   .$$$.
(4)  042202  061620                        .WORD   .$$$.
(3)  042204  103100                        .WORD   .$$$.
589  042206                                MOVE    INP0 <11>,OUT1 <5>
(3)  042206  021225                        .WORD   .$$$.
590  042210                                BR      CMNOP
(2)  042210  100420                        .WORD   .$$$.
591  042212  177777                        .WORD   -1
592          000001                        .END
```

```
ADDRES= 177777          SAMP     042100        .ADDWC= 000020       .DMEM  = 002400       .SELB  = 000220
CLK    = 000020         SAMP1    042106        .AND   = 000260      .DNOP  = 000000       .SIMM  = 000000
CMNOP    042040         SAMP2    042114        .BB0   = 002000      .DOUT0= 002000        .SIN0  = 020000
DRLPX2   042000 G       SAMP3    042130        .BB1   = 002400      .DOUT1= 001000        .SIN1  = 120000
LOOP     042046         SLOOP    042122        .BB4   = 003000      .DSPAD= 003000        .SMEM  = 040000
LOPE     042166         SLOOP1   042134        .BB7   = 003400      .DSPBR= 003400        .SUB   = 000340
MARHLD= 000000          SLOOP2   042146        .BC    = 001000      .DO    = 000400       .SUBWC= 000040
MARINC= 014000          SLOOP3   042170        .BR    = 000400      .FO    = 000020       .SUB2C= 000360
MARLD  = 010000         SLOOP4   042200        .BSBRG= 160000       .INC   = 000060       .SO    = 020000
MARLDX= 004000          STARTU   042016        .BSIMM= 100000       .LORN  = 000240       .XOR   = 000320
PAGE0  = 000000         WRCMND   042074        .BSMEM= 140000       .MINUS= 000360        .$$$.  = 100420
PAGE1  = 001000         WRSILO   042064        .BZ    = 001400      .MO    = 004000       ..LOC  = 042040
PAGE2  = 002000         $$$SER= 000001         .CO    = 000400      .OR    = 000300       .2A    = 000120
PAGE3  = 003000         .      = 042214        .DBR   = 000400      .PLUS  = 000000       .2AWC  = 000140
RDCMND   042070         .ADC   = 000100        .DBRSH= 001400       .SBREG= 060000
RDSILO   042060         .ADD   = 000000        .DEC   = 000160      .SELA  = 000200

. ABS.   042214     000        OVR    RW     ABS    LCL    D
         000000     001        CON    RW     ABS    LCL    I
ABCODE   004000     002        CON    RW     REL    LCL    I


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CRLPX2,CRLPX2=CRLPX2
 RUN-TIME: 3 3 0 SECONDS
 RUN-TIME RATIO: 34/7-4.4
 CORE USED:  36K  (71 PAGES)
```