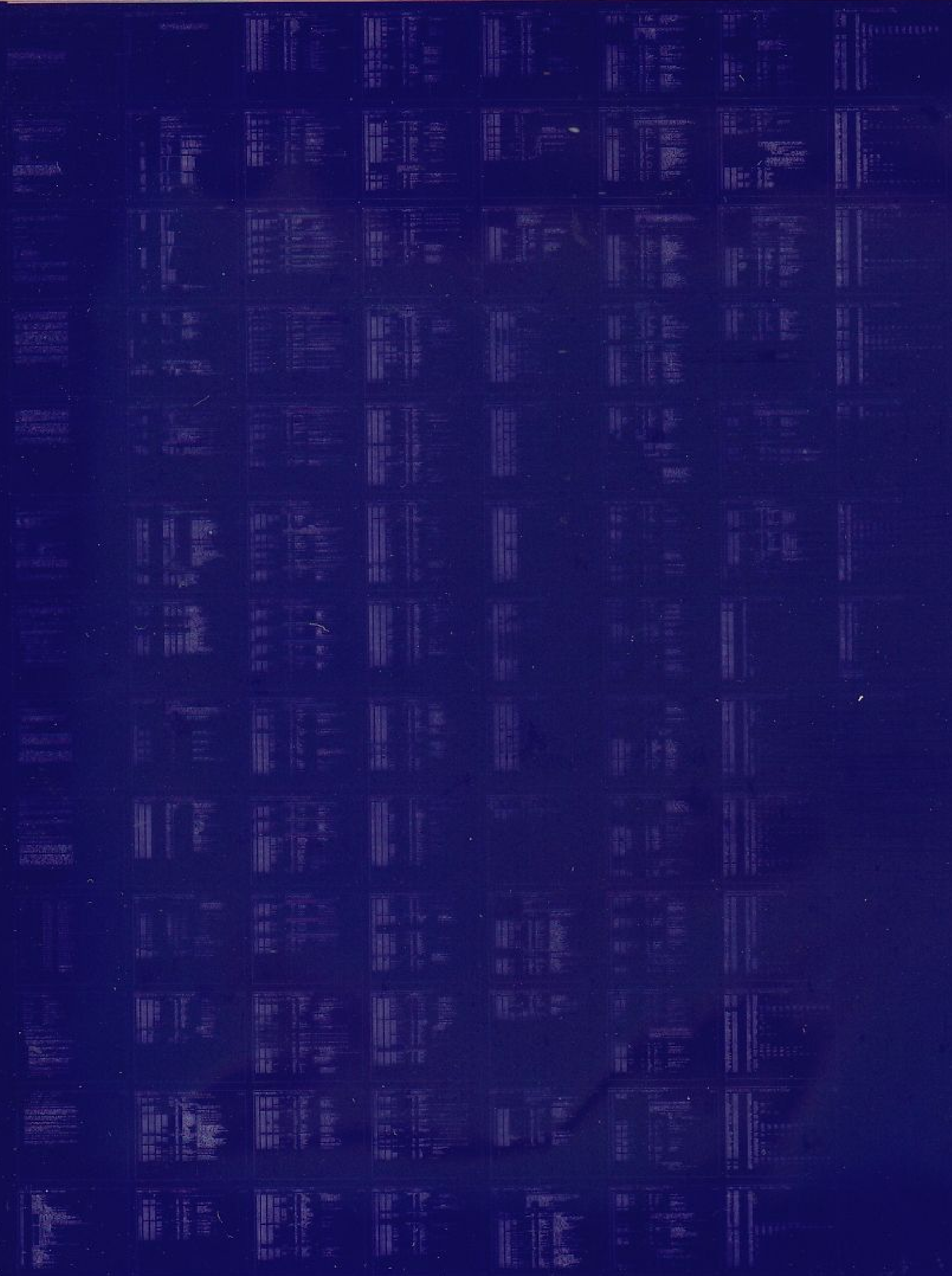
LPA11,AD11K

LPA/AD11-K DIAG TEST
CRLPKB0

AH-B050B-MC

COPYRIGHT 76–80
FICHE 1 OF 1

JAN 1980
digital
MADE IN USA



IDENTIFICATION

Product Code:	AC-B049B-MC
Product Name:	CRLPKB0 LPA/AD11-K DIAG TEST
Date:	JAN 1979
Revised:	JULY 1979
Maintainer:	Diagnostic Group

Copyright (C) 1976, 1977, 1979
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-DRLPA'' 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
*************************************************************
```

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 tne A/D start bit. It also checks that the A/D done flag sets at end of conversion and clears when the converted value is read. It checks the interrupt logic and the correct setting of the error flag.

9.2 Calibration Test

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

9.3 Differential Linearity

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

9.4 Settling Test

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

9.5 Noise Test

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

9.6 Analog Tests

These 11 subtests check the channels and their output.

10. LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

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

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

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN THE USER RUNS DRLPA HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY ''LOOK'' ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

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

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

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:

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

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

PRODUCT CODE:           MAINDEC-11-DZADL-B

PRODUCT NAME:           AD11-K PERFORMANCE TEST

DATE:                   DECEMBER 1976

MAINTAINER:             DIANOSTIC GROUP

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

PRODUCT CODE:           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.
```

```
                    .REM    [

                                  LPA.MAC

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

                                  GOOD LUCK !

                    [
                    .GLOBL  DRLPX2
```

```
 2935                                   .TITLE  LPA-AD11K TEST MD-11-CRLPKB
  (1)                                   ;*COPYRIGHT (C) 1979
  (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)                                   ;*
 2936                                   .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
  2937                                .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>
  2938          170400                ABASE=  170400
  2939          140340                AVECT1= 140340
  2940          000300                APRIOR= 300
  2941
  2946
  2953
  2958
  2965
  2970
  2976
  2982
  2987
  2988                                .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  001714                JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
  2989 000204   000137  002404                JMP     @#BEG2          ;RESTART ADDRESS
  2990 000210   000137  001722                JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
  2991 000214   000137  020550                JMP     @#$UTK                  ;STARTING ADDRESS FOR USER LINK
```

```
 2993                                   .SBTTL  ACT11 HOOKS
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;HOOKS REQUIRED BY ACT11
  (1)             000220                        $SVPC=.                         ;SAVE PC
  (1)             000046                        .=46
  (1)    000046   012100                        $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                        .=$SVPC                         ;; RESTORE PC
 2994             001000                .=1000
 2995                                   .SBTTL  APT PARAMETER BLOCK
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  (2)                                   ;;*****************************************************************
  (1)             001000                        .$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)
```

```
                                   .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
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


 001256                         $ERRTB:



                                ;ITEM   1
 001256  014257                         EM1                     ;STATUS REG. ERROR
 001260  014417                         DH1                     ;ERRPC STREG EXPECTED ACTUAL
 001262  014602                         DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
 001264  014642                         DF1


                                ;ITEM   2
 001266  014305                         EM2                     ;FAILED TO INTERRUPT
 001270  014540                         DH3                     ;ERRPC STREG ACTUAL
 001272  014632                         DT3                     ;$ERRPC, STREG,  $BDDAT
 001274  014642                         DF1

                                ;ITEM   3
 001276  014335                         EM3                     ;UNEXPECTED INTERRUPT
 001300  014540                         DH3                     ;ERRPC STREG
 001302  014632                         DT3                     ;$ERRPC, STREG
 001304  014642                         DF1

                                ;ITEM   4
 001306  014366                         EM4                     ;ERROR ON A/D CHANNEL
 001310  014455                         DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
 001312  014614                         DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT
 001314  014642                         DF1
```

```
3036                            .SBTTL          MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
3037  001316  170400            STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
3038  001320  170402            ADBUFF: ABASE+2                 ;ADDRESS OF A/D BUFFER
3039  001322  000300            BASEBR: APRIOR                  ;INTERRUPT PRIORITY LEVEL
3040  001324  140342            VECTR1: AVECT1+2
3041  001326  000040            VADR:   40                      ;INCREMENT FOR BUS ADDRESS
3042  001330  000040            VVCT:   40                      ;INCREMENT FOR VECTOR ADDRESS
3043  001332  000000            BASECH: 0                       ;BASE CHANNEL
3044  001334  000060            KBVECT: 60
3045  001336  000000            WIDE:   0                       ;NO. OF WIDE STATES
3046  001340  000000            NARROW: 0                       ;NO. OF NARROW STATES
3047  001342  000000            FIRST:  0
3048  001344  000000            SKIPST: 0                       ;NO. OF SKIPPED STATES
3049  001346  000000            TEMP:   0                       ;WORK AREA
3050  001350  000000            CH1:    0                       ;FIRST CHANNEL
3051  001352  000000            CH2:    0                       ;SECOND CHANNEL
3052  001354  000000            NBEXT:  0                       ;NO. OF AD11K'S TO BE TESTED
3053  001356  000000            NMBEXT: 0                       ;NO. OF AD11K'S TO BE TESTED
3054  001360  000000            DUMMY:  0                       ;DUMMY CHANNEL
3055  001362  000000            CHANL:  0                       ;CHANNEL VALUE
3056  001364  000000            TADDR:  0                       ;TEST ADDRESS
3057  001366  000000            RNA:    0                       ;RANDOM
3058  001370  000000            RNB:    0                       ;NUMBER
3059  001372  000000            RNC:    0                       ;VALUES
3060  001374  000000            RMS:    0                       ;RMS NOISE VALUE
3061  001376  000000            PEAK:   0                       ;PEAK NOISE VALUE
3062  001400  000000            FLAG:   0                       ;VT55 FLAG
3063  001402  000000            SPREAD: 0                       ;DEVIATION FROM THE NOMINAL
3064  001404  000000            DAC:    0                       ;SAR VALUE
3065  001406  000000            DELAY:  0                       ;TIME DELAY COUNTER
3066  001410  000000            EDGE:   0                       ;EDGE VALUE
3067  001412  000000            BITPNT: 0
3068  001414  000000            MIN:    0                       ;MIN VALUE
3069  001416  000000            WFTEST: 0                       ;OPTION TEST AREA FLAG
3070  001420  000000            MAX:    0                       ;MAX VALUE
3071  001422  000000            PERCNT: 0                       ;PERCENT FOR SAR ROUTINE
3072  001424  000000            OUT:    0
3073  001426  000000            MYTEMP: 0
3074  001430  000000            EDINT:  0
3075  001432  000000            $TEMP1: 0
3076  001434  000000            $TEMP2: 0
```

```
3078
3079                                            ;
 (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)
 (1)  001436                            LPCI:
 (1)  001436  170460                    KMADO:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.
 (1)
 (1)  001440                            LPMR:
 (1)  001440  170461                    KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
 (1)  001442                            LPCO:
 (1)  001442  170462                    KMAD2:  .WORD   170460+2                ;>PATCH         <;
 (1)  001444                            LPSO:
 (1)  001444  170463                    KMAD3:  .WORD   170460+3                ;>THIS AREA     <
 (1)  001446                            LPADL:
 (1)  001446  170464                    KMAD4:  .WORD   170460+4                ;
 (1)  001450                            LPADH:
 (1)  001450  170465                    KMAD5:  .WORD   170460+5                ;>DO NOT        <
 (1)  001452                            LPMS1:
 (1)  001452  170466                    KMAD6:  .WORD   170460+6                ;>PATCH         <
 (1)  001454                            LPMS2:
 (1)  001454  170467                    KMAD7:  .WORD   170460+7                ;>THIS AREA     <
 (1)
 (1)  001456  000340                    VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
 (1)  001460  000344                    VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2
 (1)
 (1)  001462  000004                    VERSN:  .WORD   4               ;CURRENT VERSION NUMBER OF MICROCODE.
 (1)
 (1)  001464  000000                    .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
 (1)  001466  000020                            .BLKW   16.             ;/BY INIT.
 (1)
3080
3081  001526                            UNEXP:
 (1)  001526  012737  001542  001162            MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
3082  001534  005237  001103                    INC     $ERFLG
3083  001540  104003                            ERROR   3
3084  001542  005037  001162            1$:     CLR     $ESCAPE         ;RETURN ESCAPE TO NORMAL
3085  001546  000002                            RTI                     ;UNEXPECTED INTERRUPT
```

```
                                      .SBTTL          CONTROL A AND C DECODERS
      001550  010046                  ISERV:  MOV     R0,-(SP)                          ;SAVE R0
      001552  017700  177370                  MOV     @$TKB,R0                          ;GET CHARACTER
      001556  042700  177600                  BIC     #177600,R0
      001562  120027  000003                  CMPB    R0,#3                             ;IS IT ^C?
      001566  001010                          BNE     1$
      001570  104401  012250                  TYPE    ,CMSG                             ;ECHO CHARACTER
      001574  012706  001100                  MOV     #STACK,SP
      001600  004737  011362                  JSR     PC,RST                            ;RESET & SET INTRPT. EN.
      001604  000137  002404                  JMP     BEG2
      001610  120027  000001          1$:     CMPB    R0,#1                             ;IS IT ^A?
      001614  001010                          BNE     2$
      001616  104401  012243                  TYPE    ,AMSG                             ;ECHO CHARACTER
      001622  012706  001100                  MOV     #STACK,SP
      001626  004737  011362                  JSR     PC,RST                            ;RESET & SET INTRPT. EN.
      001632  000177  177526                  JMP     @TADDR                            ;RETURN TO TEST
      001636  120027  000007          2$:     CMPB    R0,#7                             ;IS IT ^G?
      001642  001021                          BNE     NONE
      001644  023727  001140  177570          CMP     SWR,#177570                       ;HARDWARE SWREG?
      001652  001415                          BEQ     NONE
      001654  104401  012255                  TYPE    ,GMSG                             ;ECHO CHARACTER
      001660  017746  177254                  MOV     @SWR,-(SP)                    ;;SAVE @SWR FOR TYPEOUT
 (1)                                                                                ;;TYPE SWREG
 (1)  001664  104403                          TYPOS                                 ;;GO TYPE--OCTAL ASCII
 (1)  001666     006                          .BYTE   6                             ;;TYPE 6 DIGITS
 (1)  001667     001                          .BYTE   1                             ;;TYPE LEADING ZEROS
      001670  104401  012435                  TYPE    ,SLASH
      001674  104407                          RDOCT                                     ;READ NEW VALUE
      001676  012677  177236                  MOV     (SP)+,@SWR                        ;LOAD NEW SWREG VALUE
      001702  012600                  POPR0:  MOV     (SP)+,R0
      001704  000002                  RETURN: RTI
      001706  104401  012241          NONE:   TYPE    ,QUEST                            ;TYPE "?"
      001712  000773                          BR      POPR0
```

```
                                        .SBTTL          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
       001714  005037  001416           BEGIN:  CLR     WFTEST
       001720  000403                           BR      RBEG
       001722  012737  000001  001416   BEGIN2: MOV     #1,WFTEST
       001730                           RBEG:   ;RESET
                                        .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                   ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001730  012706  001100                   MOV     #$CMTAG,R6         ;;FIRST LOCATION TO BE CLEARED
  (1)  001734  005026                           CLR     (R6)+              ;;CLEAR MEMORY LOCATION
  (1)  001736  022706  001140                   CMP     #SWR,R6 ;;DONE?
  (1)  001742  001374                           BNE     .-6                ;;LOOP BACK IF NO
  (1)  001744  012706  001100                   MOV     #STACK,SP          ;;SETUP THE STACK POINTER
  (1)                                   ;;INITIALIZE A FEW VECTORS
  (1)  001750  012737  015240  000020           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001756  012737  000340  000022           MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001764  012737  015516  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001772  012737  000340  000032           MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  002000  012737  021302  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  002006  012737  000340  000036           MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  002014  012737  021356  000024           MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  002022  012737  000340  000026           MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  002030  013737  012054  012046           MOV     $ENDCT,$EOPCT      ;;SETUP END-OF-PROGRAM COUNTER
  (1)  002036  005037  001160                   CLR     $TIMES             ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  002042  005037  001162                   CLR     $ESCAPE            ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  002046  112737  000001  001115           MOVB    #1,$ERMAX          ;;ALLOW ONE ERROR PER TEST
  (1)  002054  012737  002054  001106           MOV     #.,$LPADR          ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  002062  012737  002062  001110           MOV     #.,$LPERR          ;;SETUP THE ERROR LOOP ADDRESS
  (2)                                   ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                   ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  002070  013746  000004                   MOV     @#ERRVEC,-(SP)   ;;SAVE ERROR VECTOR
  (2)  002074  012737  002130  000004           MOV     #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
  (2)  002102  012737  177570  001140           MOV     #DSWR,SWR          ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  002110  012737  177570  001142           MOV     #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
  (2)  002116  022777  177777  177014           CMP     #-1,@SWR           ;;TRY TO REFERENCE HARDWARE SWR
  (2)  002124  001012                           BNE     66$                ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                      ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  002126  000403                           BR      65$                ;;BRANCH IF NO TIMEOUT
  (2)  002130  012716  002136           64$:    MOV     #65$,(SP)          ;;SET UP FOR TRAP RETURN
  (2)  002134  000002                           RTI
  (2)  002136  012737  000176  001140   65$:    MOV     #SWREG,SWR         ;;POINT TO SOFTWARE SWR
  (2)  002144  012737  000174  001142           MOV     #DISPREG,DISPLAY
  (2)  002152  012637  000004           66$:    MOV     (SP)+,@#ERRVEC   ;;RESTORE ERROR VECTOR
  (1)
  (2)  002156  005037  001202                   CLR     $PASS              ;;CLEAR PASS COUNT
  (2)  002162  132737  000200  001215           BITB    #APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
  (2)  002170  001403                           BEQ     67$                ;;YES,USE NON-APT SWITCH
  (2)  002172  012737  001216  001140           MOV     #$SWREG,SWR        ;;NO,USE APT SWITCH REGISTER
  (2)  002200                           67$:
```

```
 3124                                            ;
  (1)                                            ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
  (1)                                            ;
  (1)
  (1)   002200  010046                           MOV     R0,-(SP)
  (1)   002202  010146                           MOV     R1,-(SP)
  (1)   002204  013700  001436                   MOV     KMAD0,R0        ;GET KMC-11 ADDRESS.
  (1)   002210  012701  001440                   MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.
  (1)
  (1)   002214  005200                   68$:    INC     R0              ;UPDATE ADDR.
  (1)   002216  010021                           MOV     R0,(1)+         ;WRITE ADDR.
  (1)   002220  020127  001456                   CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
  (1)   002224  001373                           BNE     68$             ;NO - DO NEXT ADDR.
  (1)   002226  005037  001464                   CLR     .DVLS           ;CLR ADDR. LIST.
  (1)   002232  012601                           MOV     (SP)+,R1
  (1)   002234  012600                           MOV     (SP)+,R0
 3125   002236  005037  001400                   CLR     FLAG            ;CLEAR VT55 FLAG
 3126   002242  005737  000042                   TST     @#42                    ;IS IT CHAINED?
 3127   002246  001033                           BNE     REST1
 3128                                    .SBTTL       DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
 3129   002250  042777  000100  176666           BIC     #100,@$TKS
 3130   002256  104401  013675                   TYPE    ,C0             ;TYPE ASCIZ STRING
 3131   002262  004737  002656                   JSR     PC,VTFLG        ;GET A CHARACTER
 3132   002266  020027  000033                   CMP     R0,#33
 3133   002272  001017                           BNE     NOVT55          ;NO VT55 PRESENT
 3134   002274  004737  002656                   JSR     PC,VTFLG        ;GET A CHARACTER
 3135   002300  020027  000057                   CMP     R0,#57
 3136   002304  001012                           BNE     NOVT55          ;NO VT55 PRESENT
 3137   002306  004737  002656                   JSR     PC,VTFLG        ;GET A CHARACTER
 3138   002312  020027  000103                   CMP     R0,#103
 3139   002316  001403                           BEQ     VT55            ;VT55 IS PRESENT
 3140   002320  020027  000105                   CMP     R0,#105
 3141   002324  001002                           BNE     NOVT55
 3142   002326  005237  001400           VT55:   INC     FLAG
```

```
                                          ;          DIALOGUE TO DETERMINE WHICH TEST TO RUN
 002332  104401  014040                   NOVT55: TYPE     ,HEAD1
 002336  004737  005376                   REST1:  JSR      PC,FIXONE                     ;INITIALIZE ADDRESSES
 002342  013700  001334                           MOV      KBVECT,R0
 002346  012720  001550                           MOV      #ISERV,(R0)+
 002352  012710  000340                           MOV      #340,(R0)
 002356  012737  062341  001366                   MOV      #62341,RNA                    ;RANDOM NO, VARIABLES
 002364  012737  142315  001370                   MOV      #142315,RNB
 002372  012737  127623  001372                   MOV      #127623,RNC
 002400  004737  011650                           JSR      PC,WFADJ                      ;STANDARD OR OPTION TEST TOLERANCES?
 002404  012706  001100                   BEG2:   MOV      #STACK,SP                     ;RESET STACK IN CASE RESTARTED
 002410  005737  000042                           TST      @#42                          ;IS IT CHAINED?
 002414  001402                                   BEQ      1$
 002416  000137  005114                           JMP      BEGL                          ;GO TO LOGIC TESTS
 002422  104401  013503                   1$:     TYPE     ,MSG71
 002426  104406                           TRYAG:  RDLIN
 002430  052777  000100  176506                   BIS      #100,@$TKS
 002436  005037  177776                           CLR      PSW
 002442  012600                                   MOV      (SP)+,R0                      ;READ ANSWER
 002444  142710  000040                           BICB     #40,(R0)
 002450  121027  000101                           CMPB     (R0),#'A                      ;IS IT A?
 002454  001002                                   BNE      1$                  ;;NO, TRY C
 002456  000137  005156                           JMP      BEGINA                        ;GO TO AUTO TEST
 002462  121027  000103                   1$:     CMPB     (R0),#'C                      ;IS IT C?
 002466  001002                                   BNE      2$                  ;;NO, TRY L
 002470  000137  004656                           JMP      BEGINC                        ;GO TO CALIBRATION TEST
 002474  121027  000114                   2$:     CMPB     (R0),#'L                      ;IS IT L?
 002500  001002                                   BNE      3$                  ;;NO, TRY N
 002502  000137  005114                           JMP      BEGL                          ;GO TO LOGIC TESTS
 002506  121027  000116                   3$:     CMPB     (R0),#'N                      ;IS IT N?
 002512  001002                                   BNE      4$                  ;;NO, TRY S
 002514  000137  005540                           JMP      BEGINN                        ;GO TO NOISE TEST
 002520  121027  000123                   4$:     CMPB     (R0),#'S                      ;IS IT S?
 002524  001002                                   BNE      5$                  ;;NO, TRY W
 002526  000137  005610                           JMP      BEGINS                        ;GO TO SETTLE TEST
 002532  121027  000127                   5$:     CMPB     (R0),#'W                      ;IS IT W?
 002536  001002                                   BNE      6$                  ;;NO,TRY AGAIN
 002540  000137  005250                           JMP      BEGINW                        ;GO TO WRAPAROUND TEST
 002544  104401  012241                   6$:     TYPE     ,QUEST
 002550  000726                                   BR       TRYAG                         ;WAIT FOR CHARACTER
```

```



                                        ;SIZE AND REPORT THE NUMBER OF AD11K DETECTED

      002552  013737  001250  001126    TESTAD: MOV     $BASE,$BDDAT     ;SETUP TO TEST FOR AD11K'S
      002560  005037  001464                    CLR     .DVLS
      002564  005037  001466                    CLR     .DVLS+2
      002570  005037  001354                    CLR     NBEXT            ;CLEAR AD11K COUNTER
      002574                            1$:                              ;ADDRESS AD11K

 (1)                                    ;*      MOV     $GDDAT,@$BDDAT   ;/ PUT DATA FROM $GDDAT TO DEVICE REG $BDDAT
      002604  005737  017454                    TST     $AERR            ;DEVICE EXSIST? =0,YES
      002610  001006                            BNE     2$               ;=1,NO.

      002612  005237  001354                    INC     NBEXT            ;INCREMENT AD11K COUNTER
      002616  063737  001326  001126            ADD     VADR,$BDDAT      ;GET NEXT AD11K
      002624  000763                            BR      1$               ;;TRY NEXT AD11K
      002626                            2$:
      002626  013746  001354                    MOV     NBEXT,-(SP)      ;;SAVE NBEXT FOR TYPEOUT
 (1)                                                                     ;;TYPE NUMBER OF AD11K'S
 (1)  002632  104403                            TYPOS                    ;;GO TYPE--OCTAL ASCII
 (1)  002634     002                            .BYTE   2                ;;TYPE 2 DIGIT(S)
 (1)  002635     000                            .BYTE   0                ;;SUPPRESS LEADING ZEROS
      002636  104401  013043                    TYPE    ,MSG50
      002642  005337  001354                    DEC     NBEXT            ;ADJUST AD11K COUNT
      002646  013737  001354  001356            MOV     NBEXT,NMBEXT     ;KEEP COUNT OF NUMBER
      002654  000207                            RTS     PC

      002656  005000                    VTFLG:  CLR     R0               ;TEST FOR PRESENCE
      002660  105777  176260            1$:     TSTB    @$TKS            ;OF VT55
      002664  100404                            BMI     2$               ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
      002666  005300                            DEC     R0
      002670  001373                            BNE     1$               ;;
      002672  005726                            TST     (SP)+            ;POP A WORD OFF STACK
      002674  000616                            BR      NOVT55           ;;NO VT55 PRESENT
      002676  017700  176244            2$:     MOV     @$TKB,R0
      002702  042700  177600                    BIC     #177600,R0               ;TEST VT55 CODE
      002706  000207                            RTS     PC
```

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

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

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

                                    ;;*****************************************************************
                                    ;*TEST 4         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
                                    ;;*****************************************************************
003016  000004                      TST4:   SCOPE
003020  012737  000020  001124              MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
003026  004737  003400                      JSR     PC,TESTIT
003032  104001                              ERROR   1               ;FAILED TO LOAD + READ EXT. START ENABLE
                                    ;;*****************************************************************
                                    ;*TEST 5         LOAD AND READ BACK ERROR FLAG BIT15
                                    ;;*****************************************************************
003034  000004                      TST5:   SCOPE
003036  012737  100000  001124              MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
003044  004737  003400                      JSR     PC,TESTIT
003050  104001                              ERROR   1               ;FAILED TO LOAD + READ ERROR FLAG
```

```
                                    ;;*****************************************************************
                                    ;*TEST 6         TEST  DONE FLAG  SETS AND BIT0 CLEARS ON END OF CONV.
                                    ;;*****************************************************************
003052  000004                      TST6:   SCOPE
003054  012700  001000                      MOV     #BIT9,R0        ;STALL TIME COUNTER


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

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


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

                                    ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003144  004737  003410                      JSR     PC,TEST
003150  104001                              ERROR   1               ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR

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


                                    ;;*****************************************************************
                                    ;*TEST 7         TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
                                    ;;*****************************************************************
003166  000004                      TST7:   SCOPE
003170  012737  000001  001426              MOV     #BIT0,MYTEMP

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


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


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

```
                                ;;***************************************************************
                                ;*TEST 10        TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
                                ;;***************************************************************
003252  000004                  TST10:  SCOPE
003254  012737  000010  001160          MOV     #10,$TIMES         ;;DO 10 ITERATIONS
003262  012737  000001  001426          MOV     #BIT0,MYTEMP

                                ;*      MOV     MYTEMP,@STREG      ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003300                          1$:

                                ;*      MOV     @STREG,MYTEMP      ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
003310  105737  001426                  TSTB    MYTEMP
003314  100371                          BPL     1$
003316  012737  100200  001124  2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
003324  012737  000001  001426          MOV     #BIT0,MYTEMP

                                ;*      MOV     MYTEMP,@STREG      ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
003342  012700  001000                  MOV     #BIT9,R0           ;WAIT FOR 2ND
003346  005300                  3$:     DEC     R0                 ;CONVERSION TO END
003350  001376                          BNE     3$
003352  004737  003410          4$:     JSR     PC,TEST
003356  104001                          ERROR   1                  ;ERROR FLAG NOT SET WHEN 2ND
                                                                   ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST

                                ;*      MOV     @ADBUFF,MYTEMP     ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
003370  013700  001426                  MOV     MYTEMP,R0          ;/PUT CONVERTED VALUE IN R0.

003374  000004                          SCOPE
003376  000207                          RTS     PC                 ;RETURN TO TEST SECTION



                                ;;SUBROUTINE FOR LOGIC TESTS;;
003400                          TESTIT:

                                ;*      MOV     $GDDAT,@STREG      ;/ PUT DATA FROM $GDDAT TO DEVICE REG STREG
003410                          TEST:

                                ;*      MOV     @STREG,$BDDAT      ;/READ DEVICE REG STREG,PUT DATA IN $BDDAT.
003420  023737  001124  001126          CMP     $GDDAT,$BDDAT      ;COMPARE RESULTS
003426  001002                          BNE     RETERR             ;;ERROR RETURN
003430  062716  000002                  ADD     #2,(SP)            ;BUMP RETURN ADDRESS TO GET AROUND ERROR
003434  000207                  RETERR: RTS     PC
```

```
 3297                                        .SBTTL          WRAPAROUND TEST SECTION
 3298  003436                                WRAP:
 3299                                        ;;*****************************************************************
  (3)                                        ;*TEST 11        TEST CH14 GROUND
  (3)                                        ;;*****************************************************************
  (2)  003436  000240                        TST11:  NOP
  (1)  003440  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 3300  003446  012737  000011  001102                MOV     #$TN-1,$TSTNM
 3301  003454  012737  003470  001110                MOV     #1$,$LPERR
 3302  003462  012737  003470  001106                MOV     #1$,$LPADR
 3303  003470  004537  011072                1$:     JSR     R5,CONVRT               ;DO 8 CONVERSIONS
 3304  003474  000014                                14
 3305  003476  004537  011314                        JSR     R5,COMPAR               ;COMPARE RESULTS
 3306  003502  004000                                4000                    ;NOMINAL
 3307  003504  011726                                V50                     ;TOLERANCE
 3308  003506  104004                                ERROR   4       ;ERROR-CH14 NOT GROUND-AD11K MUST BE IN SINGLE-ENDED
 3309                                        ;CONFIGURATION,G5036 WRAPAROUND MODULE MUST BE PRESENT,CHECK CONNECTION A-VV,VV-A
 3310                                        ;;*****************************************************************
  (3)                                        ;*TEST 12        TEST CONVERSION FROM EXT. START
  (3)                                        ;;*****************************************************************
  (2)  003510  000004                        TST12:  SCOPE
  (1)  003512  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 3311  003520  005737  001332                        TST     BASECH                  ;TESTING AN AM?
 3312  003524  001044                                BNE     TST13           ;;YES, GOTO NEXT TEST
 3313  003526  012737  000020  001426                MOV     #BIT4,MYTEMP
 3314
  (1)                                        ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
 3315  003544  012700  001000                        MOV     #BIT9,R0                ;TIME DELAY COUNTER
 3316  003550  012737  000220  001124                MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPECTED
 3317  003556  012737  000200  001426                MOV     #200,MYTEMP
 3318
  (1)                                        ;*      MOV     MYTEMP,@ADBUFF  ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
 3319                                                                                ;WRAPAROUND MODULE PRESENT
 3320  003574  005300                        1$:     DEC     R0
 3321  003576  001376                                BNE     1$
 3322  003600  004737  003410                        JSR     PC,TEST
 3323  003604  104001                                ERROR   1                       ;FAILED TO DO CONVERSION FROM EXT. START
 3324
  (2)                                        ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
  (1)  003616  013700  001426                        MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
 3325  003622  005037  001426                        CLR     MYTEMP
  (2)
  (2)                                        ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
 3326                                        ;;*****************************************************************
  (3)                                        ;*TEST 13        TEST CH0 GROUND
  (3)                                        ;;*****************************************************************
  (2)  003636  000004                        TST13:  SCOPE
  (1)  003640  012737  000010  001160                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 3327  003646  004537  011072                        JSR     R5,CONVRT               ;CONVERT 8 TIMES
 3328  003652  000000                                0
 3329  003654  004537  011314                        JSR     R5,COMPAR               ;COMPARE RESULTS
 3330  003660  004000                                4000                            ;NOMINAL
 3331  003662  011720                                V1                              ;TOLERANCE
 3332  003664  104004                                ERROR   4                       ;ERROR ON A/D CHANNEL
```

```
                                                      ;;*******************************************************************
 (3)                                                  ;*TEST 14          TEST CH1 GROUND
 (3)                                                  ;;*******************************************************************
 (2)  003666  000004                                  TST14:  SCOPE
 (1)  003670  012737  000010  001160                          MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      003676  004537  011072                                  JSR     R5,CONVRT                    ;CONVERT 8 TIMES
      003702  000001                                          1                                    ;CHANNEL 1
      003704  004537  011314                                  JSR     R5,COMPAR                    ;COMPARE RESULTS
      003710  004000                                          4000                                 ;NOMINAL
      003712  011724                                          V10                                  ;TOLERANCE
      003714  104004                                          ERROR   4                            ;ERROR ON A/D CHANNEL


                                                      ;;*******************************************************************
 (3)                                                  ;*TEST 15          TEST CH2 +1 VOLT
 (3)                                                  ;;*******************************************************************
 (2)  003716  000004                                  TST15:  SCOPE
 (1)  003720  012737  000010  001160                          MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      003726  004537  011072                                  JSR     R5,CONVRT                    ;CONVERT 8 TIMES
      003732  000002                                          2                                    ;CHANNEL 2
      003734  004537  011314                                  JSR     R5,COMPAR                    ;COMPARE RESULTS
      003740  004632                                          4632                                 ;NOMINAL
      003742  011726                                          V50                                  ;TOLERANCE
      003744  104004                                          ERROR   4                            ;ERROR ON A/D CHANNEL
                                                                                                   ;AD11K MUST BE SET UP FOR +OR- 5V OR +OR- 5.12V


                                                      ;;*******************************************************************
 (3)                                                  ;*TEST 16          TEST CH3 +2.5 VOLTS
 (3)                                                  ;;*******************************************************************
 (2)  003746  000004                                  TST16:  SCOPE
 (1)  003750  012737  000010  001160                          MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      003756  004537  011072                                  JSR     R5,CONVRT                    ;CONVERT 8 TIMES
      003762  000003                                          3                                    ;CHANNEL 3
      003764  004537  011314                                  JSR     R5,COMPAR                    ;COMPARE RESULTS
      003770  006000                                          6000                                 ;NOMINAL
      003772  011734                                          V240                                 ;TOLERANCE
      003774  104004                                          ERROR   4                            ;ERROR ON A/D CHANNEL


                                                      ;;*******************************************************************
 (3)                                                  ;*TEST 17          TEST CH4 -2.5 VOLTS
 (3)                                                  ;;*******************************************************************
 (2)  003776  000004                                  TST17:  SCOPE
 (1)  004000  012737  000010  001160                          MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      004006  004537  011072                                  JSR     R5,CONVRT                    ;CONVERT 8 TIMES
      004012  000004                                          4                                    ;CHANNEL 4
      004014  004537  011314                                  JSR     R5,COMPAR                    ;COMPARE RESULTS
      004020  002000                                          2000                                 ;NOMINAL
      004022  011734                                          V240                                 ;TOLERANCE
      004024  104004                                          ERROR   4
```

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

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

                                            ;*      MOV     MYTEMP,@ADBUFF   ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
004126  004737  004646                              JSR     PC,DAWAIT                  ;DELAY FOR DAC SETTLING
004132  004537  011072                              JSR     R5,CONVRT                  ;CONVERT IT
004136  000012                                      12
004140  005737  001420                              TST     MAX
004144  001010                                      BNE     2$
004146  023727  001346  004000                      CMP     TEMP,#4000
004154  002404                                      BLT     2$
004156  005237  001420                              INC     MAX
004162  010237  001414                              MOV     R2,MIN
004166  020227  000200                      2$:     CMP     R2,#200
004172  001003                                      BNE     3$
004174  013737  001346  004266                      MOV     TEMP,4$
004202  013703  001346                      3$:     MOV     TEMP,R3                    ;SAVE VALUE
004206  160437  001346                              SUB     R4,TEMP                    ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
004212  010304                                      MOV     R3,R4                      ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
004214  004537  011314                              JSR     R5,COMPAR                  ;COMPARE RESULTS
004220  000006                                      6                                  ;WITH 15 MILLIVOLTS(1 DAC LSB)
004222  011736                                      V5
004224  104004                                      ERROR   4
004226  005202                                      INC     R2
004230  020227  000400                              CMP     R2,#400                    ;DONE?
004234  001326                                      BNE     1$                         ;NO-DO NEXT VERNIER DAC VALUE
004236  004737  020426                              JSR     PC,$RESET
004242  052777  000100  174674                      BIS     #100,@$TKS
004250  004737  004646                              JSR     PC,DAWAIT                  ;LET DAC SETTLE
004254  004537  011072                              JSR     R5,CONVRT                  ;CONVERT IT
004260  000012                                      12
004262  004537  011314                              JSR     R5,COMPAR                  ;COMPARE RESULTS
004266  000000                              4$:     0
004270  011722                                      V2
004272  104004                                      ERROR   4
```

```
                                              ;;*****************************************************************
 (3)                                          ;*TEST 21        TEST CH13 +2.5 VOLTS
 (3)                                          ;;*****************************************************************
 (2)  004274  000004                          TST21:  SCOPE
 (1)  004276  012737  000010  001160                  MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      004304  004537  011072                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
      004310  000013                                  13
      004312  004537  011314                          JSR     R5,COMPAR                 ;COMPARE RESULTS
      004316  006000                                  6000                              ;NOMINAL
      004320  011730                                  V144                              ;TOLERANCE
      004322  104004                                  ERROR   4
                                              ;;*****************************************************************
 (3)                                          ;*TEST 22        TEST CH17 +4V
 (3)                                          ;;*****************************************************************
 (2)  004324  000004                          TST22:  SCOPE
 (1)  004326  012737  000010  001160                  MOV     #10,$TIMES        ;;DO 10 ITERATIONS
      004334  004537  011072                          JSR     R5,CONVRT                 ;CONVERT 8 TIMES
      004340  000017                                  17                                ;CHANNEL 17
      004342  004537  011314                          JSR     R5,COMPAR                 ;COMPARE RESULTS
      004346  007146                                  7146                              ;NOMINAL
      004350  011734                                  V240                              ;TOLERANCE
      004352  104004                                  ERROR   4                         ;ERROR ON A/D CHANNEL
                                              ;;*****************************************************************
 (3)                                          ;*TEST 23        OFFSET ON CH0
 (3)                                          ;;*****************************************************************
 (2)  004354  000004                          TST23:  SCOPE
 (1)  004356  012737  000001  001160                  MOV     #1,$TIMES         ;;DO 1 ITERATION
      004364  013737  001332  001362                  MOV     BASECH,CHANL              ;LOAD CHANNEL
      004372  013737  001332  001360                  MOV     BASECH,DUMMY              ;LOAD DUMMY
      004400  012737  004001  001410                  MOV     #4001,EDGE
      004406  004537  006452                          JSR     R5,SARSUB
      004412  000062                                  50.
      004414  013737  001404  001346                  MOV     DAC,TEMP
      004422  004537  006452                          JSR     R5,SARSUB
      004426  000062                                  50.
      004430  063737  001404  001346                  ADD     DAC,TEMP
      004436  162737  000062  001346                  SUB     #62,TEMP
      004444  013700  001414                          MOV     MIN,R0
      004450  006300                                  ASL     R0
      004452  160037  001346                          SUB     R0,TEMP
      004456  104401  013707                          TYPE    ,MOFSET           ;TYPE ASCIZ STRING
      004462  013702  001346                          MOV     TEMP,R2
      004466  004737  011504                          JSR     PC,DECTYP
      004472  104401  013722                          TYPE    ,MLSB             ;TYPE ASCIZ STRING
      004476  004537  011314                          JSR     R5,COMPAR         ;IS RESULT WITHIN LIMITS?
      004502  000000                                  0
      004504  011740                                  V50D
      004506  000401                                  BR      OFFERR            ;NO-ERROR
      004510  000403                                  BR      OFFOK             ;YES-OK
      004512  104401  012511                  OFFERR: TYPE    ,ERMSG
      004516  000402                                  BR      TST24             ;;GO TO NEXT TEST
      004520  104401  012500                  OFFOK:  TYPE    ,OKMSG
```

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

                                         ;;*************************************************************
                                         ;;*TEST 25        SETTLE TEST ON 8 EDGES
                                         ;;*************************************************************
004564  000004                           TST25:  SCOPE
004566  012737  000001  001160                   MOV     #1,$TIMES          ;;DO 1 ITERATION
004574  004537  006122                           JSR     R5,SET8                    ;SETTLE-POSITIVE DIRECTION
004600  000015                                   15
004602  000016                                   16
004604  012737  000116  001346                   MOV     #116,TEMP
004612  004537  006122                           JSR     R5,SET8                    ;SETTLE-NEGATIVE DIRECTION
004616  000016                                   16
004620  000015                                   15
                                         ;;*************************************************************
                                         ;;*TEST 26        DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
                                         ;;*************************************************************
004622  000004                           TST26:  SCOPE
004624  012737  000001  001160                   MOV     #1,$TIMES          ;;DO 1 ITERATION
004632  005737  001202                           TST     $PASS                      ;FIRST TIME-SKIP DIFLIN
004636  001402                                   BEQ     LEND
004640  004737  006750                           JSR     PC,DIFLIN
004644  000207                           LEND:   RTS     PC                         ;RETURN TO TEST SECTION

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

```
                                        .SBTTL          CALIBRATION TEST
004656  012737  004656  001364  BEGINC: MOV     #BEGINC,TADDR                   ;TEST ADDRESS IN TADDR
004664  005037  001426                  CLR     MYTEMP
                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
004700  104401  013617                  TYPE    ,HEAD5                          ;TYPE OUT HEADING
004704  005037  177776                  CLR     PSW
004710  017700  174224          1$:     MOV     @SWR,R0                         ;READ CHANNEL FROM SWITCH REG.
004714  042700  177700                  BIC     #177700,R0                      ;ISOLATE MUX BITS
004720  032777  020000  174212          BIT     #BIT13,@SWR                     ;IS BIT 13 SET?
004726  001005                          BNE     2$              ;;YES,SKIP TYPEOUT
004730  104401  012323                  TYPE    ,CH
004734  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;TYPE CHANNEL
004736  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
004740     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
004741     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
004742                          2$:
004742  000300                          SWAB    R0                              ;SWITCH BYTES
004744  010037  001426                  MOV     R0,MYTEMP
                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
004760  012702  000010                  MOV     #10,R2                          ;TYPEOUT COUNTER
004764                          3$:
                                ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
004774  005237  001426                  INC     MYTEMP
                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
005010                          30$:
                                ;*      MOV     @STREG,MYTEMP   ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
005020  105737  001426                  TSTB    MYTEMP
005024  100371                          BPL     30$
                                ;*      MOV     @ADBUFF,MYTEMP  ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
005036  013700  001426                  MOV     MYTEMP,R0       ;/PUT CONVERTED VALUE IN R0.
005042  032777  020000  174070          BIT     #BIT13,@SWR                     ;IS BIT 13 SET?
005050  001403                          BEQ     4$                              ;NOT SET, TYPE OUT LIST
005052  010077  174064                  MOV     R0,@DISPLAY                             ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
005056  000714                          BR      1$                              ;REPEAT CONVERSION
005060  104401  012326          4$:     TYPE    ,SPACE
005064  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;PRINT OCTAL CONVERTED VALUE
005066  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005070     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
005071     001                          .BYTE   1               ;;TYPE LEADING ZEROS
005072  012701  010000                  MOV     #10000,R1
005076  005301                  5$:     DEC     R1
005100  001376                          BNE     5$
005102  005302                          DEC     R2                              ;DECREMENT THE COUNTER
005104  001327                          BNE     3$                              ;NO CARRIAGE RETURN
005106  104401  001171                  TYPE    ,$CRLF                          ;CARRIAGE RETURN
005112  000676                          BR      1$                              ;REPEAT CONVERSION
```

```
3517                                    .SBTTL          LOGIC TEST SECTION
3518  005114  012737  005114  001364  BEGL:   MOV     #BEGL,TADDR                     ;TEST ADDRESS
3519  005122  005037  001430                  CLR     EDINT
3520  005126  004737  002552                  JSR     PC,TESTAD                       ;NO OF ADDITIONAL AD'S
3521  005132  004737  002710          1$:     JSR     PC,BEGINL                       ;LOGIC TESTS
3522  005136  004737  005322                  JSR     PC,BUMPAD                       ;MORE TO TEST?
3523  005142  000773                          BR      1$                              ;TEST NEXT A/D
3524  005144  012737  005132  012016          MOV     #1$,AGTST                       ;ADDRESS FOR EOP
3525  005152  000137  012020                  JMP     $EOP                            ;TYPE END OF PASS
3526
3527                                    .SBTTL          AUTO TEST
3528  005156  012737  005156  001364  BEGINA: MOV     #BEGINA,TADDR                   ;TEST ADDRESS
3529  005164  005037  001430                  CLR     EDINT
3530  005170  005037  001202                  CLR     $PASS                           ;CLEAR PASS COUNTER
3531  005174  004737  002552                  JSR     PC,TESTAD                       ;NO. OF AD'S TO BE TESTED
3532  005200  004737  002710          1$:     JSR     PC,BEGINL                       ;LOGIC TESTS
3533  005204  104401  013001                  TYPE    ,MEND                           ;TYPE END OF LOGIC TEST
3534  005210  013746  001316                  MOV     STREG,-(SP)                     ;SAVE STREG FOR TYPEOUT
3535  005214  104403                          TYPOS                                   ;TYPE OCTAL NUMBER
3536  005216     006                          .BYTE   6                               ;TYPE 6 DIGITS
3537  005217     001                          .BYTE   1                               ;TYPE LEADING ZEROS
3538  005220  104401  001171                  TYPE    ,$CRLF                          ;TYPE A CR,LF
3539  005224  004737  003436                  JSR     PC,WRAP
3540  005230  004737  005322                  JSR     PC,BUMPAD                       ;TEST NEXT A/D
3541  005234  000761                          BR      1$                              ;TEST NEXT AD
3542  005236  012737  005200  012016          MOV     #1$,AGTST                       ;ADDRESS FOR EOP
3543  005244  000137  012020                  JMP     $EOP                            ;TYPE END OF PASS
3544
3545                                    .SBTTL          WRAPAROUND TEST
3546  005250  012737  005250  001364  BEGINW: MOV     #BEGINW,TADDR                   ;TEST ADDRESS
3547  005256  005037  001430                  CLR     EDINT
3548  005262  005037  001202                  CLR     $PASS                           ;CLEAR PASS COUNT
3549  005266  004737  002552                  JSR     PC,TESTAD                       ;NO. OF AD'S TO BE TESTED
3550  005272  004737  003436          1$:     JSR     PC,WRAP                         ;WRAPAROUND TESTS
3551  005276  005037  001430                  CLR     EDINT
3552  005302  004737  005322                  JSR     PC,BUMPAD                       ;MORE A/D'S TO BE TESTED?
3553  005306  000771                          BR      1$                              ;YES-GO TEST NEXT AD11K
3554  005310  012737  005272  012016          MOV     #1$,AGTST
3555  005316  000137  012020                  JMP     $EOP                            ;INCREMENTS $PASS
```

```
                                          ;        DETERMINE IF MORE AD11K'S TO BE TESTED
005322  005737  001354           BUMPAD: TST     NBEXT                      ;ADDITIONAL AD'S?
005326  001421                           BEQ     FIXADR                     ;NO-INITIALIZE ADDRESSES
005330  063737  001326  001316           ADD     VADR,STREG                 ;SET UP NEW ST. REG.
005336  063737  001326  001320           ADD     VADR,ADBUFF                ;SET UP NEW BUFFER ADDRESS
005344  063737  001330  001456           ADD     VVCT,VECTOR                ;SET UP NEW VECTOR
005352  063737  001330  001324           ADD     VVCT,VECTR1
005360  005077  173740                   CLR     @VECTR1
005364  005337  001354                   DEC     NBEXT                      ;ONE LESS AD11K
005370  000441                           BR      BYPASS
005372  062716  000002           FIXADR: ADD     #2,(SP)
005376  013737  001250  001316   FIXONE: MOV     $BASE,STREG                ;RELOAD INITIAL ADDRESSES
005404  013737  001250  001320           MOV     $BASE,ADBUFF
005412  062737  000002  001320           ADD     #2,ADBUFF
005420  013737  001244  001456           MOV     $VECT1,VECTOR
005426  042737  170000  001456           BIC     #170000,VECTOR
005434  113737  001245  001322           MOVB    $VECT1+1,BASEBR
005442  105037  001323                   CLRB    BASEBR+1                   ;CLEAR HIGH BYTE
005446  013737  001456  001324           MOV     VECTOR,VECTR1
005454  062737  000002  001324           ADD     #2,VECTR1
005462  005077  173636                   CLR     @VECTR1
005466  013737  001356  001354           MOV     NMBEXT,NBEXT               ;RESET COUNTER
                                 ;;LOAD .+2 AND HALT TRAP CATCH;;
005474  012700  000216           BYPASS: MOV     #216,R0                    ;FILL .+2
005500  012701  000214                   MOV     #214,R1                    ;LOAD HALT
005504  020137  001334           1$:     CMP     R1,KBVECT
005510  001410                           BEQ     2$
005512  010021                           MOV     R0,(R1)+
005514  005021                           CLR     (R1)+
005516  010100                           MOV     R1,R0
005520  005720                           TST     (R0)+
005522  020027  001002                   CMP     R0,#1002
005526  001366                           BNE     1$
005530  000207                           RTS     PC                         ;TEST NEXT A/D
005532  022021                   2$:     CMP     (R0)+,(R1)+
005534  022021                           CMP     (R0)+,(R1)+
005536  000762                           BR      1$



                                 ;        NOISE TEST, 1 EDGE
005540  012737  005540  001364   BEGINN: MOV     #BEGINN,TADDR              ;TEST ADDRESS IN TADDR
005546  104401  012132                   TYPE    ,NOIMSG                    ;ASK FOR CHANNEL
005552  104401  013636                   TYPE    ,ASKCH
005556  017737  173356  001350   1$:     MOV     @SWR,CH1                   ;LOAD CHANNEL
005564  042737  177700  001350           BIC     #177700,CH1
005572  012737  000200  001346           MOV     #200,TEMP                  ;LOAD DAC VALUE
005600  004537  010400                   JSR     R5,NOITST                  ;GO TO NOISE SUBROUTINE
005604  001350                           CH1
005606  000763                           BR      1$
```

```
 3607                                         ;               INTERCHANNEL SETTLING TEST, 1 EDGE
 3608  005610  012737  005610  001364  BEGINS: MOV    #BEGINS,TADDR               ;TEST ADDRESS IN TADDR
 3609  005616  104401  012152                  TYPE   ,SETMSG                     ;ASK FOR CHANNELS
 3610  005622  104407                          RDOCT
 3611  005624  012637  001350                  MOV    (SP)+,CH1
 3612  005630  104401  012437                  TYPE   ,TOMSG
 3613  005634  104407                          RDOCT
 3614  005636  012637  001352                  MOV    (SP)+,CH2
 3615  005642  012737  000200  001346  BK3:    MOV    #200,TEMP                   ;LOAD DAC
 3616  005650  013737  001352  001362          MOV    CH2,CHANL
 3617  005656  004737  006226                  JSR    PC,GETEDG                   ;GET EDGE VALUES
 3618  005662  005002                          CLR    R2
 3619  005664  004737  006060                  JSR    PC,SET1A                    ;SCALING = .02 LSB
 3620  005670  004737  006060                  JSR    PC,SET1A                    ;MAKE IT .01 LSB
 3621  005674  100001                          BPL    POSR2
 3622  005676  005402                          NEG    R2
 3623  005700  010204                  POSR2:  MOV    R2,R4
 3624  005702  012737  000001  006450          MOV    #1,EDGFLG
 3625  005710  004737  005716                  JSR    PC,TYPSET
 3626  005714  000752                          BR     BK3
 3627  005716  004737  011504          TYPSET: JSR    PC,DECTYP
 3628  005722  104401  012333                  TYPE   ,LSB
 3629  005726  013746  001352                  MOV    CH2,-(SP)           ;;SAVE CH2 FOR TYPEOUT
  (1)                                                                      ;;TYPE CH
  (1)  005732  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
  (1)  005734     002                          .BYTE  2                    ;;TYPE 2 DIGIT(S)
  (1)  005735     000                          .BYTE  0                    ;;SUPPRESS LEADING ZEROS
 3630  005736  104401  013730                  TYPE   ,MAT                 ;TYPE ASCIZ STRING
 3631  005742  004737  006406                  JSR    PC,TYPEDG
 3632  005746  104401  012346                  TYPE   ,SETCH
 3633  005752  013746  001350                  MOV    CH1,-(SP)           ;;SAVE CH1 FOR TYPEOUT
  (1)                                                                      ;;TYPE CH
  (1)  005756  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
  (1)  005760     002                          .BYTE  2                    ;;TYPE 2 DIGIT(S)
  (1)  005761     000                          .BYTE  0                    ;;SUPPRESS LEADING ZEROS
 3634  005762  104401  012370                  TYPE   ,ATMSG
 3635  005766  013737  001350  006024          MOV    CH1,1$
 3636  005774  163737  001332  006024          SUB    BASECH,1$
 3637  006002  012737  000200  001426          MOV    #200,MYTEMP
 3638
  (1)                                  ;*      MOV    MYTEMP,@ADBUFF      ;/ PUT DATA FROM MYTEMP TO DEVICE REG ADBUFF
 3639  006020  004537  011072                  JSR    R5,CONVRT
 3640  006024  000000                  1$:     0
 3641  006026  013746  001346                  MOV    TEMP,-(SP)          ;;SAVE TEMP FOR TYPEOUT
  (1)                                                                      ;;TYPE VALUE
  (1)  006032  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
  (1)  006034     004                          .BYTE  4                    ;;TYPE 4 DIGIT(S)
  (1)  006035     001                          .BYTE  1                    ;;TYPE LEADING ZEROS
 3642  006036  020437  011746                  CMP    R4,VSET
 3643  006042  003003                          BGT    ERR
 3644  006044  104401  012500                  TYPE   ,OKMSG
 3645  006050  000207                          RTS    PC
```

```
006052  104401  012511          ERR:    TYPE    ,ERMSG
006056  000207                          RTS     PC



                                ;;SUBROUTINE FOR SETTLING TESTS;;
006060  013737  001352  001360  SET1A:  MOV     CH2,DUMMY                       ;LOAD DUMMY
006066  004537  006452                  JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 50%
006072  000062                          50.
006074  063702  001404                  ADD     DAC,R2          ;ADD RESULT TO R2
006100  013737  001350  001360          MOV     CH1,DUMMY                       ;CHANGE DUMMY VALUE
006106  004537  006452                  JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 50%
006112  000062                          50.
006114  163702  001404                  SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
006120  000207                          RTS     PC              ;RETURN

006122  012537  001350          SET8:   MOV     (R5)+,CH1                       ;GET FIRST CHANNEL
006126  012537  001352                  MOV     (R5)+,CH2                       ;GET SECOND CHANNEL
006132  063737  001332  001350          ADD     BASECH,CH1
006140  063737  001332  001352          ADD     BASECH,CH2
006146  004737  006226                  JSR     PC,GETEDG                       ;GET EDGE VALUES
006152  005002                          CLR     R2
006154  012703  000010                  MOV     #10,R3          ;SET UP COUNTER
006160  004737  006060          SETAA:  JSR     PC,SET1A                        ;GET SETTLE VALUES
006164  005237  001410                  INC     EDGE
006170  005303                          DEC     R3
006172  001372                          BNE     SETAA           ;REPEAT 8 TIMES
006174  162737  000010  001410          SUB     #10,EDGE
006202  005702                          TST     R2
006204  100001                          BPL     R2POS
006206  005402                          NEG     R2
006210  010204                  R2POS:  MOV     R2,R4
006212  012737  000010  006450          MOV     #8.,EDGFLG
006220  004737  005716                  JSR     PC,TYPSET                       ;TYPE OUT RESULTS
006224  000205                          RTS     R5                              ;RETURN


                                ;SUBROUTINE TO GET EDGE VALUE
                                ;CALL=JSR   PC,GETEDG
                                ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                ;RESULT IN EDGE, USES R0
006226                          GETEDG:

                                ;*      MOV     TEMP,@ADBUFF    ;/ PUT DATA FROM TEMP TO DEVICE REG ADBUFF
006236  113700  001362                  MOVB    CHANL,R0                        ;GET CHANNEL
006242  000300                          SWAB    R0              ;SET UP A.D STATUS REG.
006244  010037  001426                  MOV     R0,MYTEMP

                                ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
006260  012700  000100                  MOV     #100,R0         ;DAC SETTLING DELAY
006264  005300                  1$:     DEC     R0
006266  001376                          BNE     1$
006270  005037  001410                  CLR     EDGE
006274  012700  000010                  MOV     #10,R0
006300                          CONV:
  (2)
  (2)                                         ;*     MOV     @STREG,MYTEMP     ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
  (1)  006310  005237  001426                        INC     MYTEMP
  (2)
  (2)                                         ;*     MOV     MYTEMP,@STREG     ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
 3698  006324                                 30$:
  (2)
  (2)                                         ;*     MOV     @STREG,MYTEMP     ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
  (1)  006334  105737  001426                        TSTB    MYTEMP
 3699  006340  100371                                BPL     30$
 3700
  (2)
  (2)                                         ;*     MOV     @ADBUFF,MYTEMP    ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
  (1)  006352  063737  001426  001410                ADD     MYTEMP,EDGE
 3701  006360  005300                                DEC     R0
 3702  006362  001346                                BNE     CONV
 3703  006364  006237  001410                        ASR     EDGE
 3704  006370  006237  001410                        ASR     EDGE
 3705  006374  006237  001410                        ASR     EDGE
 3706  006400  005537  001410                        ADC     EDGE
 3707  006404  000207                                RTS     PC
 3708                                         ;;SUBROUTINE TO TYPE EDGE VALUES;;
 3709  006406  013703  001410                 TYPEDG: MOV    EDGE,R3
 3710  006412  010346                                MOV     R3,-(SP)          ;;SAVE R3 FOR TYPEOUT
  (1)                                                                          ;;TYPE OCTAL VALUE OF EDGE
  (1)  006414  104403                                TYPOS                     ;;GO TYPE--OCTAL ASCII
  (1)  006416     004                                .BYTE    4                ;;TYPE 4 DIGIT(S)
  (1)  006417     001                                .BYTE    1                ;;TYPE LEADING ZEROS
 3711  006420  023727  006450  000001                CMP     EDGFLG,#1
 3712  006426  001407                                BEQ     RET
 3713  006430  062703  000007                        ADD     #7,R3
 3714  006434  104401  013700                        TYPE    ,C1               ;TYPE ASCIZ STRING
 3715  006440  010346                                MOV     R3,-(SP)          ;;SAVE R3 FOR TYPEOUT
  (1)                                                                          ;;TYPE EDGE VALUE
  (1)  006442  104403                                TYPOS                     ;;GO TYPE--OCTAL ASCII
  (1)  006444     004                                .BYTE    4                ;;TYPE 4 DIGIT(S)
  (1)  006445     001                                .BYTE    1                ;;TYPE LEADING ZEROS
 3716  006446  000207                         RET:   RTS     PC
 3717  006450  000000                         EDGFLG: 0
```

```
                                              ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
                                              ;CALL=JSR    R5,SARSUB
                                              ;    XXX;XXX=PERCENT
                                              ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
006452  012537  001422                        SARSUB: MOV     (R5)+,PERCNT                 ;GET PERCENT
006456  006337  001422                                ASL     PERCNT
006462  006337  001422                                ASL     PERCNT
006466  012737  000620  006746                        MOV     #400.,CNNO                   ;NO OF SAMPLES FOR SHORT PASS.
006474  032777  004000  172436                        BIT     #BIT11,@SWR                  ;USER WANT SHORT PASS?
006502  001010                                        BNE     SAR1
006504  000407                                        BR      SAR1                         ;ALWAYS USE SHORT SAMPLE COUNT.
006506  012737  003100  006746                        MOV     #1600.,CNNO
006514  006337  001422                                ASL     PERCNT                       ;RESCALE PERCENT FOR 1600.
006520  006337  001422                                ASL     PERCNT                       ;POINTS PER BURST
006524  012737  000200  001412        SAR1:           MOV     #200,BITPNT                  ;INITIALIZE BIT POINTER AT MSB
006532  005037  001404                                CLR     DAC                          ;INITIALIZE DAC VALUE
006536  004537  020744                                JSR     R5,$PUTS
006542  001316                                        .WORD   STREG
006544  005000                        TRY:            CLR     R0
006546  063737  001412  001404                        ADD     BITPNT,DAC                   ;TRY BIT

                                      ;*              MOV     DAC,@ADBUFF          ;/ PUT DATA FROM DAC TO DEVICE REG ADBUFF
006564  012737  000100  001406                        MOV     #100,DELAY
006572  005337  001406                1$:             DEC     DELAY                        ;STALL TIME
006576  001375                                        BNE     1$
006600  013701  006746                                MOV     CNNO,R1              ;SET UP FOR 1600. OR 400. CONVERSIONS
006604  113737  001362  001435                        MOVB    CHANL,$TEMP2+1
006612  052737  000001  001434                        BIS     #1,$TEMP2
006620  113737  001360  001433                        MOVB    DUMMY,$TEMP1+1
006626  052737  000001  001432                        BIS     #1,$TEMP1
006634                                NXTCVT:
006634  013777  001432  172604        $T6MP:          MOV     $TEMP1,@KMAD4
006642  112777  000006  172572                        MOVB    #6,@KMAD2
006650  122777  000377  172564        10$:            CMPB    #377,@KMAD2
006656  001374                                        BNE     10$
006660  013777  001434  172560                        MOV     $TEMP2,@KMAD4
006666  112777  000006  172546                        MOVB    #6,@KMAD2
006674  122777  000377  172540        20$:            CMPB    #377,@KMAD2
006702  001374                                        BNE     20$
006704  027737  172536  001410                        CMP     @KMAD4,EDGE
006712  002001                                        BGE     2$
006714  005200                                        INC     R0                           ;COUNT RESULTS .LT. EDGE
006716  005301                        2$:             DEC     R1
006720  001345                                        BNE     NXTCVT
006722  020037  001422                                CMP     R0,PERCNT
006726  003003                                        BGT     SHIFT
006730  163737  001412  001404                        SUB     BITPNT,DAC                   ;TAKE THE BIT OUT
006736  006237  001412                SHIFT:          ASR     BITPNT
006742  001300                                        BNE     TRY
006744  000205                                        RTS     R5

006746  000000                        CNNO:           .WORD   0
```

```
                                         ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
006750  104401  013124          DIFLIN: TYPE    ,MSG20
006754  005037  001424                  CLR     OUT
006760  012700  042300                  MOV     #BUFFER,R0
006764  012701  010000                  MOV     #4096.,R1       ;4096 WORDS FOR HISTOGRAM
006770  005020                  CLEAR1: CLR     (R0)+           ;CLEAR BUFFER AREA
006772  005301                          DEC     R1
006774  001375                          BNE     CLEAR1
006776  012700  021540                  MOV     #DIST,R0        ;DISTRIBUTION BUFFER POINTER
007002  012701  000310                  MOV     #200.,R1        ;200. WORDS FOR DISTRIBUTION
007006  005003                          CLR     R3
007010  005037  001424                  CLR     OUT
007014  005037  001336                  CLR     WIDE
007020  005037  001340                  CLR     NARROW
007024  005037  001342                  CLR     FIRST
007030  005037  001344                  CLR     SKIPST
007034  005020                  CLEAR2: CLR     (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
007036  005301                          DEC     R1
007040  001375                          BNE     CLEAR2
007042  012700  000011          CHANNL: MOV     #11,R0          ;CHANNEL 11
007046  063700  001332                  ADD     BASECH,R0
007052  000300                          SWAB    R0                              ;LOAD MUX BITS
007054  004537  020744                  JSR     R5,$PUTS
007060  001316                          .WORD   STREG
007062  010037  001426                  MOV     R0,MYTEMP
 (2)
 (2)                            ;*      MOV     MYTEMP,@STREG   ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
007076  010037  001432                  MOV     R0,$TEMP1
007102  052737  000001  001432          BIS     #1,$TEMP1
007110  012700  001440                  MOV     #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
007114  012777  001704  172334          MOV     #RETURN,@VECTOR
007122  012701  007776          AGAIN:  MOV     #4094.,R1
007126  004737  011010          NEXT:   JSR     PC,RANDY        ;GET RANDOM NUMBER
007132  013702  001366                  MOV     RNA,R2
007136  042702  177760                  BIC     #177760,R2      ;MASK IT TO 4 BITS ONLY
007142  001402                          BEQ     CONVR
007144  005302                  DELAY3: DEC     R2              ;STALL
007146  001376                          BNE     DELAY3          ;TIME
007150                          CONVR:
007150  013777  001432  172270  $TBF4:  MOV     $TEMP1,@KMAD4
007156  112777  000006  172256          MOVB    #6,@KMAD2
007164  122777  000377  172250  31$:    CMPB    #377,@KMAD2
007172  001374                          BNE     31$
007174  017702  172246                  MOV     @KMAD4,R2
007200  001413                          BEQ     DELAY1          ;IGNORE IF =0
007202  020227  007777                  CMP     R2,#7777        ;IGNORE IF =7777
007206  001413                          BEQ     DELAY2
007210  006302                          ASL     R2
007212  005262  042300                  INC     BUFFER(R2)      ;MAKE HISTOGRAM
007216  100013                          BPL     OKAY
007220  012762  077777  042300          MOV     #077777,BUFFER(R2)      ;PREVENT OVERFLOW
007226  000407                          BR      OKAY
007230  020227  007777          DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
007234  001400                          BEQ     DELAY2          ;WITH DUMMY INSTR.
007236  005201                  DELAY2: INC     R1
007240  005263  001346                  INC     TEMP(R3)
```

```
     007244  100403                            BMI     NOTOK
     007246  005301                    OKAY:   DEC     R1
     007250  001326                            BNE     NEXT
     007252  000403                            BR      AROUND
     007254  005037  001346            NOTOK:  CLR     TEMP
     007260  000772                            BR      OKAY
     007262  005300                    AROUND: DEC     R0
     007264  001316                            BNE     AGAIN
                                       ;DATA COLLECTION HAS NOW BEEN COMPLETED - WORK ON THE DATA COLLECTED
     007266  012700  007776                    MOV     #4094.,R0
     007272  012701  042302                    MOV     #BUFFER+2,R1
     007276  012102                    READ:   MOV     (R1)+,R2            ;GET STATE WIDTH
     007300  006202                            ASR     R2                  ;1 LSB = 800.
     007302  006202                            ASR     R2
     007304  006202                            ASR     R2
     007306  005502                            ADC     R2                  ;1 LSB = 100.
     007310  020227  000310                    CMP     R2,#200.            ;OUT OF RANGE?
     007314  002403                            BLT     INRNGE
     007316  005237  001424                    INC     OUT                 ;YES - INCREMENT COUNTER
     007322  000423                            BR      TYPBAD
     007324  006302                    INRNGE: ASL     R2
     007326  005262  021540                    INC     DIST(R2)            ;MAKE STATE WIDTH DISTRIBUTION
     007332  006202                            ASR     R2
     007334  020227  000062                    CMP     R2,#50.             ;IS IT 1/2 LSB?
     007340  002007                            BGE     NOTNAR
     007342  005237  001340                    INC     NARROW
     007346  005702                            TST     R2                  ;IS IT A SKIPPED STATE?
     007350  001002                            BNE     31$
     007352  005237  001344                    INC     SKIPST
     007356  000405                    31$:    BR      TYPBAD
     007360  020227  000226            NOTNAR: CMP     R2,#150.            ;IS IT 1.5 LSB?
     007364  003426                            BLE     LAST
     007366  005237  001336                    INC     WIDE
     007372  005737  001342            TYPBAD: TST     FIRST
     007376  001004                            BNE     60$
     007400  005237  001342                    INC     FIRST
     007404  104401  012303                    TYPE    ,STATE
     007410  010103                    60$:    MOV     R1,R3
     007412  162703  042302                    SUB     #BUFFER+2,R3
     007416  006203                            ASR     R3
     007420  010346                            MOV     R3,-(SP)            ;;SAVE R3 FOR TYPEOUT
  (1)                                                                      ;;TYPE STATE
  (1)  007422  104403                          TYPOS                       ;;GO TYPE--OCTAL ASCII
  (1)  007424     004                          .BYTE   4                   ;;TYPE 4 DIGIT(S)
  (1)  007425     001                          .BYTE   1                   ;;TYPE LEADING ZEROS
     007426  104401  012277                    TYPE    ,DASH
     007432  004737  011504                    JSR     PC,DECTYP
     007436  104401  012270                    TYPE    ,LSBMSG
     007442  005300                    LAST:   DEC     R0
     007444  001314                            BNE     READ
     007446  112737  000177  014576            MOVB    #177,DECPNT
     007454  013702  001344                    MOV     SKIPST,R2                     ;GET NO. OF SKIPPED STATES
     007460  004737  011504                    JSR     PC,DECTYP                     ;TYPE IT
     007464  104401  012526                    TYPE    ,SKPMSG             ;TYPE MESSAGE
     007470  005737  001344                    TST     SKIPST
     007474  001403                            BEQ     1$
```

```
007476  104401  012511                    TYPE     ,ERMSG             ;TYPE 'ERROR''
007502  000402                            BR       NAR
007504  104401  012500            1$:     TYPE     ,OKMSG             ;TYPE #OK#
007510  013702  001340            NAR:    MOV      NARROW,R2                  ;GET NO. OF NARROW STATES
007514  004737  011504                    JSR      PC,DECTYP                  ;TYPE IT
007520  104401  012550                    TYPE     ,NARMSG            ;TYPE MESSAGE
007524  013702  001336                    MOV      WIDE,R2
007530  063702  001424                    ADD      OUT,R2
007534  004737  011504                    JSR      PC,DECTYP                  ;TYPE NO. OF WIDE STATES
007540  104401  012607                    TYPE     ,WIDMSG            ;TYPE MESSAGE
007544  013702  001424                    MOV      OUT,R2
007550  004737  011504                    JSR      PC,DECTYP                  ;TYPE NO. OF STATES OUTSIDE 2 LSB
007554  104401  012646                    TYPE     ,OUTMSG            ;TYPE MESSAGE
007560  005737  001424                    TST      OUT
007564  001403                            BEQ      11$
007566  104401  012511                    TYPE     ,ERMSG             ;TYPE 'ERROR''
007572  000402                            BR       HALF
007574  104401  012500            11$:    TYPE     ,OKMSG             ;TYPE 'OK''
007600  013702  001340            HALF:   MOV      NARROW,R2
007604  063702  001336                    ADD      WIDE,R2
007610  063702  001424                    ADD      OUT,R2
007614  010200                            MOV      R2,R0
007616  004737  011504                    JSR      PC,DECTYP                          ;TYPE NO. OF STATES OUTSIDE LIMITS
007622  112737  000056  014576            MOVB     #56,DECPNT
007630  104401  012701                    TYPE     ,HAFMSG
007634  020027  000051                    CMP      R0,#41.            ;COMPARE IT TO NOMINAL
007640  003403                            BLE      21$
007642  104401  012511                    TYPE     ,ERMSG             ;TYPE 'ERROR''
007646  000402                            BR       SWDIST
007650  104401  012500            21$:    TYPE     ,OKMSG             ;TYPE 'OK''
007654  005737  001400            SWDIST: TST      FLAG               ;VT55?
007660  001426                            BEQ      RELACC
007662  004737  010342                    JSR      PC,DELCLR          ;WAIT AWHILE, THEN CLEAR VT55
007666  104401  013156                    TYPE     ,MSG16
007672  104401  013757                    TYPE     ,BUFF1             ;TYPE BUFF1-PRINT GRID
007676  012700  021540                    MOV      #DIST,R0           ;POINTER TO STATE WIDTH DISTRIBUTION
007702  012701  000310                    MOV      #200.,R1           ;GO 200. TIMES UP TO 2 LSB
007706  012002                    NXTY1:  MOV      (R0)+,R2
007710  004737  011402                    JSR      PC,LOADY
007714  005002                            CLR      R2
007716  004737  011402                    JSR      PC,LOADY
007722  005301                            DEC      R1
007724  001370                            BNE      NXTY1
007726  104401  013702                    TYPE     ,C2                ;TYPE ASCIZ STRING
007732  004737  010342                    JSR      PC,DELCLR
```

```
                                    ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

007736  005001                      RELACC: CLR     R1                      ;RUNNING ERROR = 0
007740  005003                              CLR     R3                      ;MAXIMUM ERROR = 0
007742  104401  013551                      TYPE    ,MSG21
007746  012700  042302                      MOV     #BUFFER+2,R0
007752  011002                      NXTSTA: MOV     (R0),R2                 ;STATE WIDTH = R2
007754  162702  001440                      SUB     #800.,R2                ;STATE WIDTH ERROR IN R2
007760  060201                              ADD     R2,R1                   ;UPDATE RUNNING ERROR
007762  010120                              MOV     R1,(R0)+                ;SAVE IN BUFFER
007764  010104                              MOV     R1,R4                   ;SAVE IN R4 ALSO
007766  100001                              BPL     PLUS                    ;IS IT POSITIVE?
007770  005404                              NEG     R4                      ;NO - MAKE IT POSITIVE
007772  020403                      PLUS:   CMP     R4,R3                   ;CHECK AGAINST PREVIOUS MAX. ERROR
007774  003405                              BLE     NOTNEW                  ;NOT A NEW MAXIMUM
007776  010403                              MOV     R4,R3                   ;UPDATE MAXIMUM IN R3
010000  010005                              MOV     R0,R5
010002  162705  042302                      SUB     #BUFFER+2,R5
010006  006205                              ASR     R5                      ;R5=EDGE VALUE AT MAX. RELACC
010010  020027  062276              NOTNEW: CMP     R0,#BUFFER+8190.        ;DONE?
010014  001356                              BNE     NXTSTA                  ;NO - REPEAT
010016  006203                              ASR     R3                      ;RESCALE FROM 1 LSB = 800. SCALING
010020  006203                              ASR     R3                      ;TO 1 LSB = 100. SCALING
010022  006203                              ASR     R3
010024  005503                              ADC     R3
010026  010302                              MOV     R3,R2
010030  004737  011504                      JSR     PC,DECTYP
010034  104401  013576                      TYPE    ,LINEA
010040  010546                              MOV     R5,-(SP)                ;;SAVE R5 FOR TYPEOUT
                                                                            ;;TYPE VALUE
010042  104403                              TYPOS                           ;;GO TYPE--OCTAL ASCII
010044     004                              .BYTE   4                       ;;TYPE 4 DIGIT(S)
010045     001                              .BYTE   1                       ;;TYPE LEADING ZEROS
010046  104401  012435                      TYPE    ,SLASH                          ;PRINT '/'
010052  005205                              INC     R5
010054  010546                              MOV     R5,-(SP)                ;;SAVE R5 FOR TYPEOUT
                                                                            ;;TYPE VALUE
010056  104403                              TYPOS                           ;;GO TYPE--OCTAL ASCII
010060     004                              .BYTE   4                       ;;TYPE 4 DIGIT(S)
010061     001                              .BYTE   1                       ;;TYPE LEADING ZEROS
010062  020337  011750                      CMP     R3,VLIN
010066  003403                              BLE     41$
010070  104401  012511                      TYPE    ,ERMSG
010074  000402                              BR      42$
010076  104401  012500              41$:    TYPE    ,OKMSG
010102  005737  001400              42$:    TST     FLAG                    ;VT55?
010106  001503                              BEQ     L02
010110  012700  042300                      MOV     #BUFFER,R0
010114  012701  010000                      MOV     #4096.,R1
```

```
010120  011002                  GETDAT: MOV     (R0),R2                 ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
010122  006202                          ASR     R2                      ;RESCALE IT TO 1 LSB = 100.
010124  006202                          ASR     R2
010126  006202                          ASR     R2
010130  005502                          ADC     R2
010132  062702  000166                  ADD     #118.,R2                ;AND MOVE IT TO MID-SCREEN
010136  010220                          MOV     R2,(R0)+                ;PUT IT BACK INTO BUFFER
010140  005301                          DEC     R1
010142  001366                          BNE     GETDAT
010144  012700  042300                  MOV     #BUFFER,R0
010150  012704  042300                  MOV     #BUFFER,R4
010154  012705  042302                  MOV     #BUFFER+2,R5
010160  012701  001000                  MOV     #512.,R1
010164  012702  000007          NXT8:   MOV     #7.,R2
010170  012003                          MOV     (R0)+,R3
010172  010337  001414                  MOV     R3,MIN                  ;MINIMUM
010176  010337  001420                  MOV     R3,MAX                  ;MAXIMUM
010202  012003                  NXTCMP: MOV     (R0)+,R3
010204  020337  001414                  CMP     R3,MIN
010210  002002                          BGE     MAXTST
010212  010337  001414                  MOV     R3,MIN                  ;NEW MINIMUM
010216  020337  001420          MAXTST: CMP     R3,MAX
010222  003402                          BLE     TST8
010224  010337  001420                  MOV     R3,MAX                  ;NEW MAXIMUM
010230  005302                  TST8:   DEC     R2
010232  001363                          BNE     NXTCMP
010234  013724  001414                  MOV     MIN,(R4)+
010240  013725  001420                  MOV     MAX,(R5)+
010244  022425                          CMP     (R4)+,(R5)+             ;BUMP EACH ONCE MORE
010246  005301                          DEC     R1
010250  001345                          BNE     NXT8
010252  104401  013064                  TYPE    ,MSG18
010256  104401  014005                  TYPE    ,BUFF2                  ;TYPE BUFF2
010262  012700  042300                  MOV     #BUFFER,R0
010266  004737  010320                  JSR     PC,LOAD
010272  104401  013705                  TYPE    ,C3                     ;TYPE ASCIZ STRING
010276  012700  042302                  MOV     #BUFFER+2,R0
010302  004737  010320                  JSR     PC,LOAD
010306  104401  013702                  TYPE    ,C2                     ;TYPE ASCIZ STRING
010312  004737  010342                  JSR     PC,DELCLR
010316  000207                  LO2:    RTS     PC
010320  012701  001000          LOAD:   MOV     #512.,R1
010324  012002                  LOAD0:  MOV     (R0)+,R2
010326  005720                          TST     (R0)+
010330  004737  011402                  JSR     PC,LOADY
010334  005301                          DEC     R1
010336  001372                          BNE     LOAD0
010340  000207                          RTS     PC
```

```
     010342  005000                             DELCLR: CLR     R0
     010344  012701  000020                             MOV     #20,R1                  ;DELAY BEFORE CLEANING SCREEN
     010350  005300                             1$:     DEC     R0
     010352  001376                                     BNE     1$
     010354  005301                                     DEC     R1
     010356  001374                                     BNE     1$
     010360  032777  010000  170552                     BIT     #BIT12,@SWR             ;TEST FOR HALT FOR DISPLAY
     010366  001401                                     BEQ     2$                      ;;DON'T HALT FOR DISPLAY
     010370  000000                                     HALT
     010372  104401  014025                     2$:     TYPE    ,VTINIT
     010376  000207                                     RTS     PC
                                                ;;NOISE SUBROUTINE;;
     010400  013537  001362                     NOITST: MOV     @(R5)+,CHANL                    ;LOAD CHANNEL
     010404  013737  001362  001360                     MOV     CHANL,DUMMY                     ;LOAD DUMMY CHANNEL
     010412  004737  006226                             JSR     PC,GETEDG                       ;GET EDGE VALUE
     010416  004737  010572                             JSR     PC,NOIA                         ;GET RMS AND PEAK VALUES
     010422  012737  000001  006450                     MOV     #1,EDGFLG
     010430  004737  010436                             JSR     PC,TYPRP                        ;TYPE RMS AND PEAK VALUES
     010434  000205                                     RTS     R5






                                                ;;TYPE RMS AND PEAK VALUES;;
     010436  104401  012375                     TYPRP:  TYPE    ,NOI
     010442  005737  001374                             TST     RMS
     010446  100002                                     BPL     POSRMS
     010450  005037  001374                             CLR     RMS                             ;RMS<0,SET RMS=0
     010454  005737  001376                     POSRMS: TST     PEAK
     010460  100002                                     BPL     POSPEA
     010462  005037  001376                             CLR     PEAK                            ;PEAK<0,SET PEAK=0
     010466  013702  001374                     POSPEA: MOV     RMS,R2
     010472  004737  011504                             JSR     PC,DECTYP
     010476  104401  012750                             TYPE    ,MESR
     010502  013702  001376                             MOV     PEAK,R2
     010506  004737  011504                             JSR     PC,DECTYP
     010512  104401  012763                             TYPE    ,MESP
     010516  004737  006406                             JSR     PC,TYPEDG
     010522  104401  012405                             TYPE    ,CHAN
     010526  013746  001362                             MOV     CHANL,-(SP)             ;;SAVE CHANL FOR TYPEOUT
 (1)                                                                                    ;;TYPE CHANL
 (1) 010532  104403                                     TYPOS                           ;;GO TYPF--OCTAL ASCII
 (1) 010534     002                                     .BYTE   2                       ;;TYPE 2 DIGIT(S)
 (1) 010535     000                                     .BYTE   0                       ;;SUPPRESS LEADING ZEROS
     010536  023737  001374  011742                     CMP     RMS,VNR                 ;WITHIN LIMITS?
     010544  003007                                     BGT     ER
     010546  023737  001376  011744                     CMP     PEAK,VNP                        ;WITHIN LIMITS?
     010554  003003                                     BGT     ER
     010556  104401  012500                             TYPE    ,OKMSG
     010562  000207                                     RTS     PC
     010564  104401  012511                     ER:     TYPE    ,ERMSG
     010570  000207                                     RTS     PC
```

```
                                        ;;SUBROUTINES FOR NOISE TEST;;
010572  005037  001374                  NOIA:   CLR     RMS                             ;CLEAR RMS VLAUE
010576  005037  001376                          CLR     PEAK                            ;CLEAR PEAK VALUE
010602  004537  006452                  NOI1:   JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 16%
010606  000020                                  16.
010610  063737  001404  001374                  ADD     DAC,RMS                 ;ADD RESULT TO RMS
010616  004537  006452                          JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 84%
010622  000124                                  84.
010624  163737  001404  001374                  SUB     DAC,RMS                 ;SUBTRACT RESULT FROM RMS
010632  004537  006452                          JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 1%
010636  000001                                  1
010640  063737  001404  001376                  ADD     DAC,PEAK                        ;ADD RESULT TO PEAK
010646  004537  006452                          JSR     R5,SARSUB                       ;DO SAR ROUTINE AT 99%
010652  000143                                  99.
010654  163737  001404  001376                  SUB     DAC,PEAK                        ;SUBTRACT RESULT FROM PEAK
010662  000207                                  RTS     PC                              ;RETURN

010664  012537  001362                  NOI8:   MOV     (R5)+,CHANL                     ;GET CHANNEL VALUE
010670  063737  001332  001362                  ADD     BASECH,CHANL
010676  013737  001362  001360                  MOV     CHANL,DUMMY                     ;LOAD DUMMY CHANNEL
010704  004737  006226                          JSR     PC,GETEDG                       ;GET EDGE VALUES
010710  005037  001374                          CLR     RMS                     ;CLEAR RMS VALUE
010714  005037  001376                          CLR     PEAK                    ;CLEAR PEAK VALUE
010720  012737  000010  011006                  MOV     #10,10$                 ;SET UP COUNTER
010726  004737  010602                  1$:     JSR     PC,NOI1                 ;GET NOISE VALUES
010732  005237  001410                          INC     EDGE
010736  005337  011006                          DEC     10$
010742  001371                                  BNE     1$                      ;REPEAT 8 TIMES
010744  162737  000010  001410                  SUB     #10,EDGE
010752  006237  001374                          ASR     RMS                     ;SCALE IT TO 1 LSB=100.
010756  005537  001374                          ADC     RMS
010762  006237  001376                          ASR     PEAK
010766  005537  001376                          ADC     PEAK
010772  012737  000010  006450                  MOV     #8.,EDGFLG
011000  004737  010436                          JSR     PC,TYPRP                        ;TYPE RESULTS
011004  000205                                  RTS     R5                      ;RETURN
011006  000000                          10$:    0                               ;COUNTER



                                        ;;RANDOM NUMBER GENERATOR;;
011010  063737  001370  001366          RANDY:  ADD     RNB,RNA
011016  063737  001372  001366                  ADD     RNC,RNA
011024  005537  001366                          ADC     RNA
011030  063737  001366  001370                  ADD     RNA,RNB
011036  063737  001372  001370                  ADD     RNC,RNB
011044  005537  001370                          ADC     RNB
011050  063737  001366  001372                  ADD     RNA,RNC
011056  063737  001370  001372                  ADD     RNB,RNC
011064  005537  001372                          ADC     RNC
011070  000207                                  RTS     PC
```

```
                                      ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
     011072  012500                   CONVRT: MOV    (R5)+,R0                      ;GET CHANNEL VALUE
     011074  063700  001332                   ADD    BASECH,R0
     011100  010037  001362                   MOV    R0,CHANL
     011104  000300                           SWAB   R0
     011106  005037  001346                   CLR    TEMP

 (1)                                  ;*      MOV    @ADBUFF,MYTEMP   ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
     011122  010037  001426                   MOV    R0,MYTEMP
 (2)
 (2)                                  ;*      MOV    MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
     011136  012700  010000                   MOV    #10000,R0
     011142  005300                   2$:     DEC    R0
     011144  001376                           BNE    2$
     011146  012777  001704  170302           MOV    #RETURN,@VECTOR               ;LOAD VECTOR
     011154  012700  000010                   MOV    #10,R0           ;SET UP COUNTER
     011160                           1$:
 (1)
 (1)                                  ;*      MOV    @STREG,MYTEMP    ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
     011170  052737  000001  001426           BIS    #1,MYTEMP

 (1)                                  ;*      MOV    MYTEMP,@STREG    ;/ PUT DATA FROM MYTEMP TO DEVICE REG STREG
     011206  005001                           CLR    R1
     011210  105201                   10$:    INCB   R1
     011212  001007                           BNE    11$
     011214  012737  000200  001124           MOV    #BIT7,$GDDAT                  ;EXPECT DONE TO SET BY NOW
     011222  013737  001426  001126           MOV    MYTEMP,$BDDAT

     011230  104001                           ERROR  1                             ;DONE FAILED TO SET ON A/D

     011232                           11$:

 (2)                                  ;*      MOV    @STREG,MYTEMP    ;/READ DEVICE REG STREG,PUT DATA IN MYTEMP.
 (1) 011242  105737  001426                   TSTB   MYTEMP
     011246  100360                           BPL    10$

 (1)                                  ;*      MOV    @ADBUFF,MYTEMP   ;/READ DEVICE REG ADBUFF,PUT DATA IN MYTEMP.
     011260  063737  001426  001346           ADD    MYTEMP,TEMP
                                                                        ;WAIT FOR CONVERSION
                                                                        ;READ BUFFER
     011266  005300                           DEC    R0
     011270  001333                           BNE    1$               ;DO 8 TIMES
     011272  006237  001346                   ASR    TEMP             ;AVERAGE VALUE
     011276  006237  001346                   ASR    TEMP
     011302  006237  001346                   ASR    TEMP
     011306  005537  001346                   ADC    TEMP
     011312  000205                           RTS    R5               ;RETURN
```

```

                                                ;COMPARE $GDDAT AND $BDDAT;;
011314  012537  001124                          COMPAR: MOV     (R5)+,$GDDAT                    ;GET GOOD DATA
011320  013537  001402                                  MOV     @(R5)+,SPREAD                   ;GET SPREAD
011324  013737  001346  001126                          MOV     TEMP,$BDDAT                     ;GET BAD(ACTUAL) DATA
011332  013701  001126                                  MOV     $BDDAT,R1
011336  013700  001124                                  MOV     $GDDAT,R0
011342  160100                                          SUB     R1,R0                   ;GET DIFFERENCE
011344  100001                                          BPL     7$
011346  005400                                          NEG     R0
011350  020037  001402                          7$:     CMP     R0,SPREAD                       ;COMPARE IT TO SPREAD
011354  003001                                          BGT     10$                     ;GO TO ERROR PRINTOUT
011356  005725                                          TST     (R5)+                   ;BUMP RETURN POINTER AROUND ERROR CALL
011360  000205                                  10$:    RTS     R5


                                                ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011362  004737  020426                          RST:    JSR     PC,$RESET
011366  052777  000100  167550                          BIS     #100,@$TKS
011374  005037  177776                                  CLR     PSW
011400  000207                                          RTS     PC




                                                ;SUBROUTINE LOADY;
011402  005702                                  LOADY:  TST     R2                      ;ROUTINE TO LOAD VLAUE INTO R2
011404  100001                                          BPL     PLUSR2                  ;AS A VT55 Y-VALUE
011406  005002                                          CLR     R2
011410  020227  000353                          PLUSR2: CMP     R2,#235.
011414  002402                                          BLT     LESS
011416  012702  000353                                  MOV     #235.,R2
011422  010203                                  LESS:   MOV     R2,R3
011424  042702  177740                                  BIC     #177740,R2
011430  052702  000040                                  BIS     #40,R2
011434  105777  167510                          B10:    TSTB    @$TPS                   ;PRINT CHARACTER
011440  100375                                          BPL     B10
011442  110277  167504                                  MOVB    R2,@$TPB
011446  006203                                          ASR     R3
011450  006203                                          ASR     R3
011452  006203                                          ASR     R3
011454  006203                                          ASR     R3
011456  006203                                          ASR     R3
011460  042703  177770                                  BIC     #177770,R3
011464  052703  000040                                  BIS     #40,R3
011470  105777  167454                          B11:    TSTB    @$TPS                   ;PRINT CHARACTER
011474  100375                                          BPL     B11
011476  110377  167450                                  MOVB    R3,@$TPB
011502  000207                                          RTS     PC


```

```
                                       ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
                                       ;;IN R2 AS X.XX;;
011504  005702                         DECTYP: TST     R2                      ;TEST VALUE TO BE TYPED
011506  100003                                 BPL     POS
011510  104401  012237                         TYPE    ,MINUS                          ;TYPE MINUS SIGN
011514  005402                                 NEG     R2
011516  020227  001747                 POS:    CMP     R2,#999.                        ;>999. REPLACE IT WITH 999.
011522  003402                                 BLE     OKAYD
011524  012702  001747                         MOV     #999.,R2
011530  105037  014600                 OKAYD:  CLRB    ONES                    ;CLEAR ONES
011534  105037  014577                         CLRB    TENS                    ;CLEAR TENS
011540  105037  014575                         CLRB    HUNS                    ;CLEAR HUNS
011544  005702                         TESTR2: TST     R2                              ;CONVERT VALUE TO A DECIMAL VALUE
011546  001424                                 BEQ     TYPOUT
011550  005302                                 DEC     R2
011552  105237  014600                         INCB    ONES
011556  123727  014600  000012                 CMPB    ONES,#10.
011564  001367                                 BNE     TESTR2
011566  105037  014600                         CLRB    ONES
011572  105237  014577                         INCB    TENS
011576  123727  014577  000012                 CMPB    TENS,#10.
011604  001357                                 BNE     TESTR2
011606  105037  014577                         CLRB    TENS
011612  105237  014575                         INCB    HUNS
011616  000752                                 BR      TESTR2
011620  152737  000060  014575         TYPOUT: BISB    #60,HUNS                        ;PREPARE FOR TYPOUT
011626  152737  000060  014577                 BISB    #60,TENS
011634  152737  000060  014600                 BISB    #60,ONES
011642  104401  014575                         TYPE    ,HUNS                   ;TYPE VALUE
011646  000207                                 RTS     PC

011650  012701  011742                 WFADJ:  MOV     #VNR,R1                         ;SUBROUTINE TO SET UP LIMITS
011654  005737  001332                         TST     BASECH                          ;TESTING AN AM11K?
011660  001403                                 BEQ     1$                      ;;
011662  012702  011774                         MOV     #VARLT3,R2                      ;BASECH NOT ZERO, USE AM11K LIMITS
011666  000410                                 BR      3$                      ;;
011670  005737  001416                 1$:     TST     WFTEST
011674  001003                                 BNE     2$
011676  012702  011754                         MOV     #VARLT1,R2                      ;WFTEST=0,USE NORMAL LIMITS
011702  000402                                 BR      3$
011704  012702  011764                 2$:     MOV     #VARLT2,R2                      ;WFTEST=1,USE OPTION AREA LIMITS
011710  012221                         3$:     MOV     (R2)+,(R1)+
011712  005711                                 TST     (R1)
011714  100375                                 BPL     3$
011716  000207                                 RTS     PC
```

```
 4250  011720  000001                          V1:     1                            ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
 4251  011722  000002                          V2:     2
 4252  011724  000010                          V10:    10
 4253  011726  000050                          V50:    50
 4254  011730  000144                          V144:   144
 4255  011732  000115                          V115:   115
 4256  011734  000240                          V240:   240
 4257  011736  000005                          V5:     5
 4258  011740  000062                          V50D:   50.
 4259
 4260  011742  000000                          VNR:    0                            ;RMS NOISE LIMIT
 4261  011744  000000                          VNP:    0                            ;PEAK NOISE LIMIT
 4262  011746  000000                          VSET:   0                            ;INTER-CHANNEL SETTLING LIMIT
 4263  011750  000000                          VLIN:   0                            ;RELATIVE ACCURACY ERROR LIMIT
 4264  011752  100000                                  BIT15
 4265
 4266  011754  000031                          VARLT1: 25.                 ;.25 LSB,NORMAL LIMITS FOR SYSTEM
 4267  011756  000310                                  200.                ;2. LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
 4268  011760  000144                                  100.                ;1 LSB
 4269  011762  000144                                  100.                ;1 LSB
 4270
 4271  011764  000027                          VARLT2: 23.                 ;.23 LSB,  TIGHTER LIMITS FOR OPTION
 4272  011766  000226                                  150.                ;1.5 LSB,  AREA USE ON SPEC TESTS
 4273  011770  000132                                  90.                 ;.9 LSB
 4274  011772  000132                                  90.                 ;.9 LSB
 4275
 4276  011774  000062                          VARLT3: 50.                 ;.5 LSB,  LIMITS FOR AM11K TESTING
 4277  011776  000310                                  200.                ;2. LSB
 4278  012000  000226                                  150.                ;1.5 LSB
 4279  012002  000226                                  150.                ;1.5 LSB
 4280
 4281  012004  052777  000100  167132  AGATST: BIS     #100,@$TKS
 4282  012012  000177  000000                          JMP     @AGTST
 4283  012016  001714                          AGTST:  BEGIN
 4285                                      .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                                      ;;*****************************************************************
  (1)                                      ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                      ;*TYPE ''END PASS''
  (1)                                      ;*IF THERES A MONITOR GO TO IT
  (1)                                      ;*IF THERE ISN'T JUMP TO AGATST
  (1)                                      ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE ''END OF PASS'' LOCATION
  (1)                                      ;*$ENDMG CAN BE CHANGED TO 7.
  (1)
  (1)  012020                              $EOP:
  (2)  012020  000240                              NOP
  (1)  012022  005037  001102                      CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  012026  005037  001160                      CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  012032  005237  001202                      INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  012036  042737  100000  001202              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  012044  005327                              DEC     (PC)+           ;;LOOP?
  (1)  012046  000001                      $EOPCT: .WORD   1
  (1)  012050  003017                              BGT     $DOAGN          ;;YES
  (1)  012052  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  012054  000001                      $ENDCT: .WORD   1
  (1)  012056  012046                              $EOPCT
  (1)  012060  104401  012117                      TYPE    ,$ENDMG         ;;TYPE ''END PASS''
  (1)  012064  104401  012114                      TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
  (1)  012070  013700  000042              $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (1)  012074  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  012076  000005                              RESET                   ;;CLEAR THE WORLD
  (1)  012100  004710                      $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  012102  000240                              NOP                     ;;SAVE ROOM
  (1)  012104  000240                              NOP                     ;;FOR
  (1)  012106  000240                              NOP                     ;;ACT11
  (1)  012110                              $DOAGN:
  (1)  012110  000137                              JMP     @(PC)+          ;;RETURN
  (1)  012112  012004                      $RTNAD: .WORD   AGATST
  (1)  012114     377     377     000      $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)  012117     015  042412  042116      $ENDMG: .ASCIZ  <15><12>/END PASS/
  (1)  012124  050040  051501  000123
 4286
```

```
                                        .SBTTL          ASCII MESSAGES
012132  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST-- /
012140  020105  042524  052123
012146  026455  000040
012152  005015  042523  052124  SETMSG: .ASCIZ  <15><12>/SETTLING TEST-- TYPE DESIRED 'FROM' CHANNEL & CR: /
012160  044514  043516  052040
012166  051505  026524  020055
012174  054524  042520  042040
012202  051505  051111  042105
012210  023440  051106  046517
012216  020047  044103  047101
012224  042516  020114  020046
012232  051103  020072     000
012237     055     000          MINUS:  .BYTE   55,0
012241     077     000          QUEST:  .BYTE   77,0
012243     136     101     040  AMSG:   .BYTE   136,101,40,40,0
012246     040     000
012250     136     103     040  CMSG:   .BYTE   136,103,40,40,0
012253     040     000
012255     136     107     015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012260     012     123     127
012263     122     105     107
012266     072     000
012270  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
012276     000
012277     055  020055     000  DASH:   .ASCIZ  /-- /
012303     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012310  026455  053440  042111
012316  044124  005015     000
012323     103  000110          CH:     .ASCIZ  /CH/
012326  020040  020040     000  SPACE:  .ASCIZ  /    /
012333     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
012340  047117  041440  000110
012346  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
012354  047111  020107  051106
012362  046517  041440  000110
012370  040440  020124     000  ATMSG:  .ASCIZ  / AT /
012375     116  044517  042523  NOI:    .ASCIZ  /NOISE: /
012402  020072     000
012405     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
012412  040510  047116  046105
012420  000040
012422  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
012430  042516  005015     000
012435     057     000          SLASH:  .ASCIZ  #/#
012437     124  050131  020105  TOMSG:  .ASCIZ  /TYPE DESIRED 'TO' CHANNEL & CR: /
012444  042504  044523  042522
012452  020104  052047  023517
012460  041440  040510  047116
012466  046105  023040  041440
012474  035122  000040
012500  020040  020040  045517  OKMSG:  .ASCIZ  /    OK/<15><12>
012506  005015     000
```

```
 4311  012511       040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
       012516    047522  025122  006452
       012524    000012
 4312  012526    051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
       012534    042105  051440  040524
       012542    042524  051450  000051
 4313  012550    047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
       012556    020127  036050  030440
       012564    031057  046040  041123
       012572    020051  052123  052101
       012600    024105  024523  005015
       012606       000
 4314  012607       040  044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
       012614    024040  020076  020061
       012622    027461  020062  051514
       012630    024502  051440  040524
       012636    042524  051450  006451
       012644    000012
 4315  012646    051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
       012654    051450  020051  044527
       012662    042504  020122  044124
       012670    047101  031040  046040
       012676    041123     000
 4316  012701       040  052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
       012706    026505  044527  052104
       012714    024110  024523  047440
       012722    052125  044523  042504
       012730    025440  047440  020122
       012736    020055  027461  020062
       012744    051514  000102
 4317  012750    046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
       012756    051515  020054     000
 4318  012763       040  051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
       012770    042520  045501  040440
       012776    020124     000
 4319  013001       015  042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
       013006    047440  020106  047514
       013014    044507  020103  042524
       013022    052123     123
 4320  013025       040  047117  040440  ONAD:   .ASCIZ  / ON AD11K AT /
       013032    030504  045461  040440
       013040    020124     000
 4321  013043       040  042101  030461  MSG50:  .ASCIZ  / AD11K'S FOUND/<15><12>
       013050    023513  020123  047506
       013056    047125  006504  000012
 4322  013064    005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
       013072    020062  051514  006502
       013100    005012  005012  005012
       013106    005012  005012  005012
 4323  013114    030455  031057  051514          .ASCIZ  \-1/2LSB\
       013122    000102
 4324
 4326                                                      .EVEN
 4327   013124  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
        013132  047105  044524  046101
        013140  046040  047111  040505
        013146  044522  054524  006472
        013154  000012
 4328   013156  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
        013164  020040  020040  020040
        013172  020040  020040  020040
        013200  020040  052123  052101
        013206  026505  044527  052104
        013214  020110  044504  052123
        013222  044522  052502  044524
        013230  047117  005015  005012
 4329   013236  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><
        013244  051440  040524  042524
        013252  005123  005012  005012
        013260  005012  005012  005012
        013266  005012  005012  005012
        013274  005012
 4330   013276  020040  020040  020040          .ASCII  /                                             STATE WIDTH (LSB)/<15>
        013304  020040  020040  020040
        013312  020040  020040  020040
        013320  020040  020040  020040
        013326  020040  020040  020040
        013334  020040  020040  020040
        013342  020040  020040  020040
        013350  020040  020040  020040
        013356  051440  040524  042524
        013364  053440  042111  044124
        013372  024040  051514  024502
        013400  005015
 4331   013402  030040  020040  020040          .ASCIZ  # 0            1/2           1            1 1/2           2#
        013410  020040  020040  020040
        013416  020040  020040  027461
        013424  020062  020040  020040
        013432  020040  020040  020040
        013440  020040  020061  020040
        013446  020040  020040  020040
        013454  020040  030440  030440
        013462  031057  020040  020040
        013470  020040  020040  020040
        013476  020040  031040     000
 4332   013503     015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR DESIRED TEST: /
        013510  020105  042514  052124
        013516  051105  023040  041440
        013524  020122  047506  020122
        013532  042504  044523  042522
        013540  020104  042524  052123
        013546  020072     000
 4333   013551     122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
        013556  053111  020105  041501
        013564  052503  040522  054503
        013572  006472  000012
 4334   013576  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
        013604  054101  046511  046525
        013612  040440  020124     000
 4335   013617     015  041412  046101  HEAD5:  .ASCII  <15><12>/CALIBRATION--/
        013624  041111  040522  044524
        013632  047117  026455
 4336   013636  051440  052105  041440  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
        013644  040510  047116  046105
        013652  044440  020116  053523
        013660  020122  047514  020127
        013666  054502  042524  005015
        013674     000
 4337   013675     033  000132          C0:     .ASCIZ  <33><132>
 4338   013700  000055                  C1:     .ASCIZ  <55>
 4339   013702  031033     000          C2:     .ASCIZ  <33><62>
 4340   013705     112     000          C3:     .ASCIZ  <112>
 4341   013707     015  047412  043106  MOFSET: .ASCIZ  <15><12>/OFFSET =/
        013714  042523  020124  000075
 4342   013722  046040  041123  000040  MLSB:   .ASCIZ  / LSB /
 4343   013730  040440  020124     000  MAT:    .ASCIZ  / AT /
 4344   013735     015  020012  047105  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
        013742  042524  044522  043516
        013750  052040  051505  020124
        013756     000
 4345   013757     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
        013762     061     111     062
        013765     114     041     060
        013770     045     063     051
        013773     066     055     071
        013776     061     074     110
        014001     041     040     112
        014004     000
 4346   014005     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
        014010     047     111     061
        014013     104     050     065
        014016     044     062     110
        014021     040     040     102
        014024     000
 4347   014025     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0
        014030     112     033     061
        014033     101     040     033
        014036     062     000
 4348   014040  005015  046412  026504  HEAD1:  .ASCII  <15><12><12>#MD-11-CRLPK-B    AD11K/LPA-11 DIAGNOSTIC#<15><12>
        014046  030461  041455  046122
        014054  045520  041055  020040
        014062  020040  042101  030461
        014070  027513  050114  026501
        014076  030461  042040  040511
        014104  047107  051517  044524
        014112  006503     012
 4349   014115     012  035101  040440          .ASCII  <12>/A: AUTO TEST/
        014122  052125  020117  042524
        014130  052123
 4350   014132  005015  035103  041440          .ASCII  <15><12>/C: CALIBRATION/
        014140  046101  041111  040522
        014146  044524  047117
 4351   014152  005015  035114  046040          .ASCII  <15><12>/L: LOGIC TEST/
        014160  043517  041511  052040
        014166  051505     124
 4352   014171     015  047012  020072          .ASCII  <15><12>/N: NOISE TEST/
        014176  047516  051511  020105
        014204  042524  052123
 4353   014210  005015  035123  051440          .ASCII  <15><12>/S: SETTLE TEST/
        014216  052105  046124  020105
        014224  042524  052123
 4354   014230  005015  035127  053440          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
        014236  040522  040520  047522
        014244  047125  020104  042524
        014252  052123  005015     000
 4355   014257     015  051412  040524  EM1:    .ASCIZ  <15><12>/STATUS REG. ERROR/<15><12>
        014264  052524  020123  042522
        014272  027107  042440  051122
        014300  051117  005015     000
 4356   014305     015  043012  044501  EM2:    .ASCIZ  <15><12>/FAILED TO INTERRUPT/<15><12>
        014312  042514  020104  047524
        014320  044440  052116  051105
        014326  052522  052120  005015
        014334     000
 4357   014335     015  052412  042516  EM3:    .ASCIZ  <15><12>/UNEXPECTED INTERRUPT/<15><12>
        014342  050130  041505  042524
        014350  020104  047111  042524
        014356  051122  050125  006524
        014364  000012
 4358   014366  005015  051105  047522  EM4:    .ASCIZ  <15><12>#ERROR ON A/D CHANNEL#<15><12>
        014374  020122  047117  040440
        014402  042057  041440  040510
        014410  047116  046105  005015
        014416     000
 4359   014417     105  051122  041520  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/<15><12>
        014424  051440  051124  043505
        014432  042440  050130  041505
        014440  042524  020104  041501
        014446  052524  046101  005015
        014454     000
 4360   014455     105  051122  041520  DH2:    .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
        014462  020040  052123  042522
        014470  020107  020040  044103
        014476  047101  042516  020114
        014504  047040  046517  047111
        014512  046101  020040  047524
        014520  042514  040522  041516
        014526  020105  040440  052103
        014534  040525  000114
 4361   014540  051105  050122  020103  DH3:    .ASCIZ  /ERRPC      STREG    ACTUAL/<15><12>
        014546  020040  020040  051440
        014554  051124  043505  020040
        014562  020040  041501  052524
        014570  046101  005015     000
 4363  014575     000                         HUNS:    .BYTE   0
 4364  014576     056                         DECPNT:  .BYTE   56
 4365  014577     000                         TENS:    .BYTE   0
 4366  014600     000      000                ONES:    .BYTE   0,0
 4367                                         .EVEN
 4368
 4369  014602  001116  001316  001124  DT1:     $ERRPC, STREG, $GDDAT, $BDDAT,0
       014610  001126  000000
 4370  014614  001116  001316  001362  DT2:     $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
       014622  001124  001402  001126
       014630  000000
 4371  014632  001116  001316  001126  DT3:     $ERRPC,STREG,$BDDAT,0
       014640  000000
 4372
 4373  014642  000000                  DF1:     0
 4374
 4375
```

```
                                        .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)  014644  011646                    $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
 (1)  014646  016666  000004  000002            MOV     4(SP),2(SP)             ;;SAVE THE PS
 (1)  014654  105777  164264            1$:     TSTB    @$TKS                   ;;WAIT FOR
 (1)  014660  100375                            BPL     1$                      ;;A CHARACTER
 (1)  014662  117766  164260  000004            MOVB    @$TKB,4(SP)             ;;READ THE TTY
 (1)  014670  042766  177600  000004            BIC     #^C<177>,4(SP)          ;;GET RID OF JUNK IF ANY
 (1)  014676  026627  000004  000023            CMP     4(SP),#23               ;;IS IT A CONTROL-S?
 (1)  014704  001013                            BNE     3$                      ;;BRANCH IF NO
 (1)  014706  105777  164232            2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
 (1)  014712  100375                            BPL     2$                      ;;LOOP UNTIL ITS THERE
 (1)  014714  117746  164226                    MOVB    @$TKB,-(SP)             ;;GET CHARACTER
 (1)  014720  042716  177600                    BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
 (1)  014724  022627  000021                    CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
 (1)  014730  001366                            BNE     2$                      ;;IF NOT DISCARD IT
 (1)  014732  000750                            BR      1$                      ;;YES, RESUME
 (1)  014734  026627  000004  000140    3$:     CMP     4(SP),#140              ;;IS IT UPPER CASE?
 (1)  014742  002407                            BLT     4$                      ;;BRANCH IF YES
 (1)  014744  026627  000004  000175            CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
 (1)  014752  003003                            BGT     4$                      ;;BRANCH IF YES
 (1)  014754  042766  000040  000004            BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
 (1)  014762  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)  014764  010346                    $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
 (1)  014766  012703  015072            1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
 (1)  014772  022703  015102            2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
 (1)  014776  101405                            BLOS    4$                      ;;BR IF YES
 (1)  015000  104405                            RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
 (1)  015002  112613                            MOVB    (SP)+,(R3)              ;;GET CHARACTER
 (1)  015004  122713  000177            10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
 (1)  015010  001003                            BNE     3$                      ;;SKIP IF NOT
 (1)  015012  104401  001170            4$:     TYPE    ,$QUES                  ;;TYPE A '?'
 (1)  015016  000763                            BR      1$                      ;;CLEAR THE BUFFER AND LOOP
 (1)  015020  111337  015070            3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
 (1)  015024  104401  015070                    TYPE    ,9$
```

```
    (1)  015030  122723  000015                   CMPB    #15,(R3)+           ;;CHECK FOR RETURN
    (1)  015034  001356                           BNE     2$                  ;;LOOP IF NOT RETURN
    (1)  015036  105063  177777                   CLRB    -1(R3)              ;;CLEAR RETURN (THE 15)
    (1)  015042  104401  001172                   TYPE    ,$LF                ;;TYPE A LINE FEED
    (1)  015046  012603                           MOV     (SP)+,R3            ;;RESTORE R3
    (1)  015050  011646                           MOV     (SP),-(SP)          ;;ADJUST THE STACK AND PUT ADDRESS OF THE
    (1)  015052  016666  000004  000002           MOV     4(SP),2(SP)         ;;       FIRST ASCII CHARACTER ON IT
    (1)  015060  012766  015072  000004           MOV     #$TTYIN,4(SP)
    (1)  015066  000002                           RTI                         ;;RETURN
    (1)  015070     000                  9$:      .BYTE   0                   ;;STORAGE FOR ASCII CHAR. TO TYPE
    (1)  015071     000                           .BYTE   0                   ;;TERMINATOR
    (1)  015072  000010                  $TTYIN:  .BLKB   8.                  ;;RESERVE 8 BYTES FOR TTY INPUT
    (1)  015102  052536  005015     000  $CNTLU:  .ASCIZ  /^U/<15><12>        ;;CONTROL 'U'
    (1)  015107     136  006507  000012  $CNTLG:  .ASCIZ  /^G/<15><12>        ;;CONTROL 'G'
    (1)  015114  005015  053523  020122  $MSWR:   .ASCIZ  <15><12>/SWR = /
    (1)  015122  020075     000
    (1)  015125     040  047040  053505  $MNEW:   .ASCIZ  /  NEW = /
    (1)  015132  036440  000040
```

```
 4379                                       .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) 015136 011646                         $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
  (1) 015140 016666 000004 000002                   MOV     4(SP),2(SP)             ;;INPUT NUMBER
  (3) 015146 010046                                 MOV     R0,-(SP)                ;;PUSH R0 ON STACK
  (3) 015150 010146                                 MOV     R1,-(SP)                ;;PUSH R1 ON STACK
  (3) 015152 010246                                 MOV     R2,-(SP)                ;;PUSH R2 ON STACK
  (1) 015154 104406                         1$:     RDLIN                           ;;READ AN ASCIZ LINE
  (1) 015156 012600                                 MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
  (1) 015160 005001                                 CLR     R1                      ;;CLEAR DATA WORD
  (1) 015162 005002                                 CLR     R2
  (1) 015164 112046                         2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
  (1) 015166 001412                                 BEQ     3$                      ;;IF ZERO GET OUT
  (1) 015170 006301                                 ASL     R1                      ;;*2
  (1) 015172 006102                                 ROL     R2
  (1) 015174 006301                                 ASL     R1                      ;;*4
  (1) 015176 006102                                 ROL     R2
  (1) 015200 006301                                 ASL     R1                      ;;*8
  (1) 015202 006102                                 ROL     R2
  (1) 015204 042716 177770                          BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
  (1) 015210 062601                                 ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
  (1) 015212 000764                                 BR      2$                      ;;LOOP
  (1) 015214 005726                         3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
  (1) 015216 010166 000012                          MOV     R1,12(SP)               ;;SAVE THE RESULT
  (1) 015222 010237 015236                          MOV     R2,$HIOCT
  (3) 015226 012602                                 MOV     (SP)+,R2                ;;POP STACK INTO R2
  (3) 015230 012601                                 MOV     (SP)+,R1                ;;POP STACK INTO R1
  (3) 015232 012600                                 MOV     (SP)+,R0                ;;POP STACK INTO R0
  (1) 015234 000002                                 RTI                             ;;RETURN
  (1) 015236 000000                         $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
 4381                                   .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                                   ;;*****************************************************************
  (1)                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                   ;*AND LOAD THE TEST NUMBER($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)  015240                           $SCOPE:
  (1)  015240  032777  040000  163672   1$:     BIT     #BIT14,@SWR        ;;LOOP ON PRESENT TEST?
  (1)  015246  001114                           BNE     $OVER              ;;YES IF SW14=1
  (1)                                   ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  015250  000416                   $XTSTR: BR      6$                 ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
  (1)                                                                      ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
  (1)  015252  013746  000004                   MOV     @#ERRVEC,-(SP)     ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  015256  012737  015276  000004           MOV     #5$,@#ERRVEC       ;;SET FOR TIMEOUT
  (1)  015264  005737  177060                   TST     @#177060           ;;TIME OUT ON XOR?
  (1)  015270  012637  000004                   MOV     (SP)+,@#ERRVEC     ;;RESTORE THE ERROR VECTOR
  (1)  015274  000463                           BR      $SVLAD             ;;GO TO THE NEXT TEST
  (1)  015276  022626                   5$:     CMP     (SP)+,(SP)+        ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  015300  012637  000004                   MOV     (SP)+,@#ERRVEC     ;;RESTORE THE ERROR VECTOR
  (1)  015304  000423                           BR      7$                 ;;LOOP ON THE PRESENT TEST
  (1)  015306                           6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  015306  032777  000400  163624           BIT     #BIT08,@SWR        ;;LOOP ON SPEC. TEST?
  (1)  015314  001404                           BEQ     2$                 ;;BR IF NO
  (1)  015316  127737  163616  001102           CMPB    @SWR,$TSTNM        ;;ON THE RIGHT TEST?    SWR<7:0>
  (1)  015324  001465                           BEQ     $OVER              ;;BR IF YES
  (1)  015326  105737  001103           2$:     TSTB    $ERFLG             ;;HAS AN ERROR OCCURRED?
  (1)  015332  001421                           BEQ     3$                 ;;BR IF NO
  (1)  015334  123737  001115  001103           CMPB    $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  015342  101015                           BHI     3$                 ;;BR IF NO
  (1)  015344  032777  001000  163566           BIT     #BIT09,@SWR        ;;LOOP ON ERROR?
  (1)  015352  001404                           BEQ     4$                 ;;BR IF NO
  (1)  015354  013737  001110  001106   7$:     MOV     $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
  (1)  015362  000446                           BR      $OVER
  (1)  015364  105037  001103           4$:     CLRB    $ERFLG             ;;ZERO THE ERROR FLAG
  (1)  015370  005037  001160                   CLR     $TIMES             ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
  (1)  015374  000415                           BR      1$                 ;;ESCAPE TO THE NEXT TEST
  (1)  015376  032777  004000  163534   3$:     BIT     #BIT11,@SWR        ;;INHIBIT ITERATIONS?
  (1)  015404  001011                           BNE     1$                 ;;BR IF YES
  (1)  015406  005737  001202                   TST     $PASS              ;;IF FIRST PASS OF PROGRAM
  (1)  015412  001406                           BEQ     1$                 ;;        INHIBIT ITERATIONS
  (1)  015414  005237  001104                   INC     $ICNT              ;;INCREMENT ITERATION COUNT
  (1)  015420  023737  001160  001104           CMP     $TIMES,$ICNT       ;;CHECK THE NUMBER OF ITERATIONS MADE
  (1)  015426  002024                           BGE     $OVER              ;;BR IF MORE ITERATION REQUIRED
  (1)  015430  012737  000001  001104   1$:     MOV     #1,$ICNT           ;;REINITIALIZE THE ITERATION COUNTER
  (1)  015436  013737  015514  001160           MOV     $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
  (1)  015444  105237  001102           $SVLAD: INCB    $TSTNM             ;;COUNT TEST NUMBERS
  (1)  015450  113737  001102  001200           MOVB    $TSTNM,$TESTN      ;;SET TEST NUMBER IN APT MAILBOX
  (1)  015456  011637  001106                   MOV     (SP),$LPADR        ;;SAVE SCOPE LOOP ADDRESS
  (1)  015462  011637  001110                   MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
  (1)  015466  005037  001162                   CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
  (1)  015472  112737  000001  001115           MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
  (1)  015500  013777  001102  163434  $OVER:   MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
  (1)  015506  013716  001106                   MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
  (1)  015512  000002                           RTI                       ;;FIXES PS
  (1)  015514  003720                  $MXCNT:  2000.                     ;;MAX. NUMBER OF ITERATIONS
 4382                                  .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)  015516                          $ERROR:
  (1)  015516  105237  001103          7$:      INCB    $ERFLG            ;;SET THE ERROR FLAG
  (1)  015522  001775                           BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
  (1)  015524  013777  001102  163410           MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
  (1)  015532  032777  002000  163400           BIT     #BIT10,@SWR       ;;BELL ON ERROR?
  (1)  015540  001402                           BEQ     1$                ;;NO - SKIP
  (1)  015542  104401  001164                   TYPE    ,$BELL            ;;RING BELL
  (1)  015546  005237  001112          1$:      INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
  (1)  015552  011637  001116                   MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
  (1)  015556  162737  000002  001116           SUB     #2,$ERRPC
  (1)  015564  117737  163326  001114           MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
  (1)  015572  032777  020000  163340           BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
  (1)  015600  001004                           BNE     20$               ;;SKIP TYPEOUTS
  (1)  015602  004737  015712                   JSR     PC,$ERRTYP        ;;GO TO USER ERROR ROUTINE
  (1)  015606  104401  001171                   TYPE    ,$CRLF
  (1)  015612                          20$:
  (1)  015612  122737  000001  001214           CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
  (1)  015620  001007                           BNE     2$                ;;NO,SKIP APT ERROR REPORT
  (1)  015622  113737  001114  015634           MOVB    $ITEMB,21$        ;;SET ITEM NUMBER AS ERROR NUMBER
  (1)  015630  004737  016346                   JSR     PC,$ATY4          ;;REPORT FATAL ERROR TO APT
  (1)  015634     000                  21$:     .BYTE   0
  (1)  015635     000                           .BYTE   0
  (1)  015636  000777                  22$:     BR      22$               ;;APT ERROR LOOP
  (1)  015640  005777  163274          2$:      TST     @SWR              ;;HALT ON ERROR
  (1)  015644  100001                           BPL     3$                ;;SKIP IF CONTINUE
  (1)  015646  000000                           HALT                      ;;HALT ON ERROR!
  (1)  015650  032777  001000  163262  3$:      BIT     #BIT09,@SWR       ;;LOOP ON ERROR SWITCH SET?
  (1)  015656  001402                           BEQ     4$                ;;BR IF NO
  (1)  015660  013716  001110                   MOV     $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
  (1)  015664  005737  001162          4$:      TST     $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
  (1)  015670  001402                           BEQ     5$                ;;BR IF NONE
  (1)  015672  013716  001162                   MOV     $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
  (1)  015676                          5$:
  (1)  015676  022737  012100  000042           CMP     #$ENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
  (1)  015704  001001                           BNE     6$                ;;BRANCH IF NO
 (1)  015706  000000                  HALT                          ;;YES
 (1)  015710                  6$:
 (1)  015710  000002                  RTI                           ;;RETURN
                         .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 (1)
 (2)                      ;;*******************************************************************
 (1)                      ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
 (1)                      ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE'' ($ERRTB),
 (1)                      ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 (1)
 (1)  015712              $ERRTYP:
 (1)  015712  104401  001171          TYPE    ,$CRLF                ;;"CARRIAGE RETURN'' & 'LINE FEED''
 (1)  015716  010046                  MOV     R0,-(SP)              ;;SAVE R0
 (1)  015720  005000                  CLR     R0                    ;;PICKUP THE ITEM INDEX
 (1)  015722  153700  001114          BISB    @#$ITEMB,R0
 (1)  015726  001004                  BNE     1$                    ;;IF ITEM NUMBER IS ZERO, JUST
 (1)                                                                ;;TYPE THE PC OF THE ERROR
 (2)  015730  013746  001116          MOV     $ERRPC,-(SP)          ;;SAVE $ERRPC FOR TYPEOUT
 (2)                                                                ;;ERROR ADDRESS
 (2)  015734  104402                  TYPOC                         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  015736  000426                  BR      6$                    ;;GET OUT
 (1)  015740  005300          1$:     DEC     R0                    ;;ADJUST THE INDEX SO THAT IT WILL
 (1)  015742  006300                  ASL     R0                    ;;       WORK FOR THE ERROR TABLE
 (1)  015744  006300                  ASL     R0
 (1)  015746  006300                  ASL     R0
 (1)  015750  062700  001256          ADD     #$ERRTB,R0            ;;FORM TABLE POINTER
 (1)  015754  012037  015764          MOV     (R0)+,2$              ;;PICKUP 'ERROR MESSAGE'' POINTER
 (1)  015760  001404                  BEQ     3$                    ;;SKIP TYPEOUT IF NO POINTER
 (1)  015762  104401                  TYPE                          ;;TYPE THE 'ERROR MESSAGE''
 (1)  015764  000000          2$:     .WORD   0                     ;;'ERROR MESSAGE'' POINTER GOES HERE
 (1)  015766  104401  001171          TYPE    ,$CRLF                ;;'CARRIAGE RETURN'' & 'LINE FEED''
 (1)  015772  012037  016002  3$:     MOV     (R0)+,4$              ;;PICKUP 'DATA HEADER'' POINTER
 (1)  015776  001404                  BEQ     5$                    ;;SKIP TYPEOUT IF 0
 (1)  016000  104401                  TYPE                          ;;TYPE THE 'DATA HEADER''
 (1)  016002  000000          4$:     .WORD   0                     ;;'DATA HEADER'' POINTER GOES HERE
 (1)  016004  104401  001171          TYPE    ,$CRLF                ;;'CARRIAGE RETURN'' & 'LINE FEED''
 (1)  016010  011000          5$:     MOV     (R0),R0               ;;PICKUP 'DATA TABLE'' POINTER
 (1)  016012  001004                  BNE     7$                    ;;GO TYPE THE DATA
 (1)  016014  012600          6$:     MOV     (SP)+,R0              ;;RESTORE R0
 (1)  016016  104401  001171          TYPE    ,$CRLF                ;;'CARRIAGE RETURN'' & 'LINE FEED''
 (1)  016022  000207                  RTS     PC                    ;;RETURN
 (1)  016024                  7$:
 (2)  016024  013046                  MOV     @(R0)+,-(SP)          ;;SAVE @(R0)+ FOR TYPEOUT
 (2)  016026  104402                  TYPOC                         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  016030  005710                  TST     (R0)                  ;;IS THERE ANOTHER NUMBER?
 (1)  016032  001770                  BEQ     6$                    ;;BR IF NO
 (1)  016034  104401  016042          TYPE    ,8$                   ;;TYPE TWO(2) SPACES
 (1)  016040  000771                  BR      7$                    ;;LOOP
 (1)  016042  020040     000  8$:     .ASCIZ  /  /                  ;;TWO(2) SPACES
 (1)          016046                  .EVEN
```

```
                                          .SBTTL  TYPE ROUTINE
 (1)
 (2)                                      ;;*******************************************************************
 (1)                                      ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 (1)                                      ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 (1)                                      ;;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 (1)                                      ;;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 (1)                                      ;;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 (1)                                      ;;*
 (1)                                      ;;*CALL:
 (1)                                      ;;*1) USING A TRAP INSTRUCTION
 (1)                                      ;;*      TYPE     ,MESADR            ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 (1)                                      ;;*OR
 (1)                                      ;;*      TYPE
 (1)                                      ;;*      MESADR
 (1)                                      ;;*
 (1)
 (1) 016046  105737  001157               $TYPE:  TSTB    $TPFLG            ;;IS THERE A TERMINAL?
 (1) 016052  100002                               BPL     1$                ;;BR IF YES
 (1) 016054  000000                               HALT                      ;;HALT HERE IF NO TERMINAL
 (1) 016056  000430                               BR      3$                ;;LEAVE
 (1) 016060  010046                       1$:     MOV     R0,-(SP)          ;;SAVE R0
 (1) 016062  017600  000002                       MOV     @2(SP),R0         ;;GET ADDRESS OF ASCIZ STRING
 (1) 016066  122737  000001  001214               CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
 (1) 016074  001011                               BNE     62$               ;;NO,GO CHECK FOR APT CONSOLE
 (1) 016076  132737  000100  001215               BITB    #APTSPOOL,$ENVM   ;;SPOOL MESSAGE TO APT
 (1) 016104  001405                               BEQ     62$               ;;NO,GO CHECK FOR CONSOLE
 (1) 016106  010037  016116                       MOV     R0,61$            ;;SETUP MESSAGE ADDRESS FOR APT
 (1) 016112  004737  016336                       JSR     PC,$ATY3          ;;SPOOL MESSAGE TO APT
 (1) 016116  000000                       61$:    .WORD   0                 ;;MESSAGE ADDRESS
 (1) 016120  132737  000040  001215       62$:    BITB    #APTCSUP,$ENVM    ;;APT CONSOLE SUPPRESSED
 (1) 016126  001003                               BNE     60$               ;;YES,SKIP TYPE OUT
 (1) 016130  112046                       2$:     MOVB    (R0)+,-(SP)       ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 (1) 016132  001005                               BNE     4$                ;;BR IF IT ISN'T THE TERMINATOR
 (1) 016134  005726                               TST     (SP)+             ;;IF TERMINATOR POP IT OFF THE STACK
 (1) 016136  012600                       60$:    MOV     (SP)+,R0          ;;RESTORE R0
 (1) 016140  062716  000002               3$:     ADD     #2,(SP)           ;;ADJUST RETURN PC
 (1) 016144  000002                               RTI                       ;;RETURN
 (1) 016146  122716  000011               4$:     CMPB    #HT,(SP)          ;;BRANCH IF <HT>
 (1) 016152  001430                               BEQ     8$
 (1) 016154  122716  000200                       CMPB    #CRLF,(SP)        ;;BRANCH IF NOT <CRLF>
 (1) 016160  001006                               BNE     5$
 (1) 016162  005726                               TST     (SP)+             ;;POP  <CR><LF> EQUIV
 (1) 016164  104401                               TYPE                      ;;TYPE A CR AND LF
 (1) 016166  001171                               $CRLF
 (1) 016170  105037  016324                       CLRB    $CHARCNT          ;;CLEAR CHARACTER COUNT
 (1) 016174  000755                               BR      2$                ;;GET NEXT CHARACTER
 (1) 016176  004737  016260               5$:     JSR     PC,$TYPEC         ;;GO TYPE THIS CHARACTER
 (1) 016202  123726  001156               6$:     CMPB    $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
 (1) 016206  001350                               BNE     2$                ;;IF NO GO GET NEXT CHAR.
 (1) 016210  013746  001154                       MOV     $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
 (1)                                                                        ;;AND THE NULL CHAR.
 (1) 016214  105366  000001               7$:     DECB    1(SP)             ;;DOES A NULL NEED TO BE TYPED?
 (1) 016220  002770                               BLT     6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
 (1) 016222  004737  016260                       JSR     PC,$TYPEC         ;;GO TYPE A NULL
 (1) 016226  105337  016324                       DECB    $CHARCNT          ;;DO NOT COUNT AS A COUNT
```

```
  (1)   016232  000770                           BR      7$                ;;LOOP
  (1)
  (1)                                  ;HORIZONTAL TAB PROCESSOR
  (1)
  (1)   016234  112716  000040         8$:       MOVB    #' ,(SP)          ;;REPLACE TAB WITH SPACE
  (1)   016240  004737  016260         9$:       JSR     PC,$TYPEC         ;;TYPE A SPACE
  (1)   016244  132737  000007  016324           BITB    #7,$CHARCNT       ;;BRANCH IF NOT AT
  (1)   016252  001372                           BNE     9$                ;;TAB STOP
  (1)   016254  005726                           TST     (SP)+             ;;POP SPACE OFF STACK
  (1)   016256  000724                           BR      2$                ;;GET NEXT CHARACTER
  (1)   016260  105777  162664         $TYPEC:   TSTB    @$TPS             ;;WAIT UNTIL PRINTER IS READY
  (1)   016264  100375                           BPL     $TYPEC
  (1)   016266  116677  000002  162556           MOVB    2(SP),@$TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
  (1)   016274  122766  000015  000002           CMPB    #CR,2(SP)         ;;IS CHARACTER A CARRIAGE RETURN?
  (1)   016302  001003                           BNE     1$                ;;BRANCH IF NO
  (1)   016304  105037  016324                   CLRB    $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
  (1)   016310  000406                           BR      $TYPEX            ;;EXIT
  (1)   016312  122766  000012  000002 1$:       CMPB    #LF,2(SP)         ;;IS CHARACTER A LINE FEED?
  (1)   016320  001402                           BEQ     $TYPEX            ;;BRANCH IF YES
  (1)   016322  105227                           INCB    (PC)+             ;;COUNT THE CHARACTER
  (1)   016324  000000                 $CHARCNT:.WORD    0                 ;;CHARACTER COUNT STORAGE
  (1)   016326  000207                 $TYPEX:   RTS     PC
  (1)
 4386                                  .SBTTL  APT COMMUNICATIONS ROUTINE
  (1)
  (2)                                  ;;*************************************************************
  (1)   016330  112737  000001  016574 $ATY1:    MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
  (1)   016336  112737  000001  016572 $ATY3:    MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
  (1)   016344  000403                           BR      $ATYC
  (1)   016346  112737  000001  016574 $ATY4:    MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
  (1)   016354                         $ATYC:
  (3)   016354  010046                           MOV     R0,-(SP)          ;;PUSH R0 ON STACK
  (3)   016356  010146                           MOV     R1,-(SP)          ;;PUSH R1 ON STACK
  (1)   016360  105737  016572                   TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
  (1)   016364  001450                           BEQ     5$                ;;IF NOT:  BR
  (1)   016366  122737  000001  001214           CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
  (1)   016374  001031                           BNE     3$                ;;IF NOT:  BR
  (1)   016376  132737  000100  001215           BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
  (1)   016404  001425                           BEQ     3$                ;;IF NOT:  BR
  (1)   016406  017600  000004                   MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
  (1)   016412  062766  000002  000004           ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
  (1)   016420  005737  001174         1$:       TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
  (1)   016424  001375                           BNE     1$                ;;IF NOT:  WAIT
  (1)   016426  010037  001210                   MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
  (1)   016432  105720                 2$:       TSTB    (R0)+             ;;FIND END OF MESSAGE
  (1)   016434  001376                           BNE     2$
  (1)   016436  163700  001210                   SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
  (1)   016442  006200                           ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
  (1)   016444  010037  001212                   MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
  (1)   016450  012737  000004  001174           MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
  (1)   016456  000413                           BR      5$
  (1)   016460  017637  000004  016504 3$:       MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
  (1)   016466  062766  000002  000004           ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
  (3)   016474  013746  177776                   MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
  (1)   016500  004737  016046                   JSR     PC,$TYPE          ;;CALL TYPE MACRO
  (1)   016504  000000                 4$:       .WORD   0
```

```
    (1)  016506                            5$:
    (1)  016506  105737  016574            10$:     TSTB    $FFLG               ;;SHOULD REPORT FATAL ERROR?
    (1)  016512  001416                             BEQ     12$                 ;;IF NOT:  BR
    (1)  016514  005737  001214                     TST     $ENV                ;;RUNNING UNDER APT?
    (1)  016520  001413                             BEQ     12$                 ;;IF NOT:  BR
    (1)  016522  005737  001174            11$:     TST     $MSGTYPE            ;;FINISHED LAST MESSAGE?
    (1)  016526  001375                             BNE     11$                 ;;IF NOT:  WAIT
    (1)  016530  017637  000004  001176             MOV     @4(SP),$FATAL       ;;GET ERROR #
    (1)  016536  062766  000002  000004             ADD     #2,4(SP)                    ;;BUMP RETURN ADDR.
    (1)  016544  005237  001174                     INC     $MSGTYPE            ;;TELL APT TO TAKE ERROR
    (1)  016550  105037  016574            12$:     CLRB    $FFLG               ;;CLEAR FATAL FLAG
    (1)  016554  105037  016573                     CLRB    $LFLG               ;;CLEAR LOG FLAG
    (1)  016560  105037  016572                     CLRB    $MFLG               ;;CLEAR MESSAGE FLAG
    (3)  016564  012601                             MOV     (SP)+,R1            ;;POP STACK INTO R1
    (3)  016566  012600                             MOV     (SP)+,R0            ;;POP STACK INTO R0
    (1)  016570  000207                             RTS     PC                  ;;RETURN
    (1)  016572     000                    $MFLG:   .BYTE   0                   ;;MESSG. FLAG
    (1)  016573     000                    $LFLG:   .BYTE   0                   ;;LOG FLAG
    (1)  016574     000                    $FFLG:   .BYTE   0                   ;;FATAL FLAG
    (1)          016576                             .EVEN
    (1)          000200                    APTSIZE=200
    (1)          000001                    APTENV=001
    (1)          000100                    APTSPOOL=100
    (1)          000040                    APTCSUP=040
 4387
    (2)                                             ;*
    (2)                                             ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
    (2)                                             ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
    (2)                                             ;*NEXT WE WILL INIT BOTH UPROCESSORS
    (2)                                             ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
    (2)                                             ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
    (2)                                             ;*
    (2)                                             ;*      CALL=   JSR     R5,$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)  016576                            $LPAI:
    (2)  016576  013746  000004                     MOV     4,-(SP)
    (2)
    (2)  016602  000413                             BR      31$             ;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)  016604  012737  016630  000004                 MOV     #30$,4
    (2)  016612  005237  170000                         INC     170000
    (3)  016616  104401  016624                         TYPE    ,65$            ;;TYPE ASCIZ STRING
    (3)  016622  000401                                 BR      64$             ;;GET OVER THE ASCIZ
    (3)                                         ;;65$:  .ASCIZ  <7>##
    (3)  016626                                 64$:
    (2)  016626  000401                                 BR      31$
    (2)  016630  022626                         30$:    CMP     (SP)+,(SP)+
    (2)  016632  012637  000004                 31$:    MOV     (SP)+,4         ;ALL THIS JUNK MUST BE REMOVED!!
    (2)  016636  005037  017454                         CLR     $AERR
    (2)  016642  004537  017456                         JSR     R5,$LOAD        ;LOAD MICRO-CODE.
    (2)  016646  000000G                                .WORD   DRLPX2          ;FILE 'DRLPX2.OBJ''
    (2)
    (2)  016650  052777  040000  162560                 BIS     #BIT14,@KMAD0   ;ISSUE KMC+DMC INIT.
    (2)
    (2)  016656                                 1$:
    (2)                                                                         ;''HANGS'' HERE THEN KMC-11 ERROR.
    (2)  016656  010146                                 MOV     R1,-(SP)
    (2)  016660  005001                                 CLR     R1
    (2)  016662  005201                         2$:     INC     R1              ;STALL FOR DMC-UP
    (2)  016664  001376                                 BNE     2$
    (2)  016666  012777  104000  162542                 MOV     #BIT15!BIT11,@KMAD0     ;SET RUN, AND ENABLE ARBITRATION.
    (2)  016674  105201                         25$:    INCB    R1
    (2)  016676  001376                                 BNE     25$
    (2)
    (2)  016700  032777  000040  162530                 BIT     #BIT5,@KMAD0    ;SLAVE READY? (READING IPBM SR)
    (2)  016706  001401                                 BEQ     3$
    (2)                                                                         ;FATAL LPA-11 ERROR SLAVE NOT READY.
    (2)  016710  104000                                 ERROR
    (2)
    (2)  016712  012777  000004  162522         3$:     MOV     #4,@KMAD2       ;READ FAST PATH
    (2)  016720                                 4$:
    (3)  016720  004537  020366                         JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  016724  104000                                 ERROR                   ;/TIME-OUT ERROR
    (3)                                                                         ;/WE FAILED TO COMPLETE
    (3)                                                                         ;/CURRENT OPERATION.
    (3)                                                                         ;/CONTINUES IN THIS LOOP
    (3)                                                                         ;/WOULD MAKE US ''HANG'' HERE
    (3)
    (3)  016726  000774                                 BR              4$
    (3)
    (3)                                                                         ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  016730  122777  000377  162504                 CMPB    #377,@KMAD2     ;WAIT TILL KMC DONE COMMAND.
    (2)  016736  001370                                 BNE     4$
    (2)  016740  122777  000377  162500                 CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
    (2)  016746  001001                                 BNE     35$
    (2)  016750  104000                                 ERROR                   ;IPBM ERROR (SLAVE SIDE)
    (2)                                                                         ;YOU MUST RUN IPBM DIAGNOSTIC.
    (2)
    (2)  016752  122777  000004  162466         35$:    CMPB    #4,@KMAD4       ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
    (2)  016760  001543                                 BEQ     5$              ;YES-CONTINUE.
    (2)  016762  005227  177777                         INC     #-1
    (2)  016766  001140                                 BNE     5$
    (2)  016770  005227  177777                         INC     #-1
    (2)  016774  001135                              BNE     5$
    (3)  016776  104401  017004                      TYPE    ,67$               ;;TYPE ASCIZ STRING
    (3)  017002  000440                              BR      66$                ;;GET OVER THE ASCIZ
    (3)                                     ;;67$:   .ASCIZ  <200>'W A R N I N G  THIS PROGRAM WAS DESIGNED TO RUN WITH VERSION 4''
    (3)  017104                             66$:
    (3)  017104  104401  017112                      TYPE    ,69$               ;;TYPE ASCIZ STRING
    (3)  017110  000430                              BR      68$                ;;GET OVER THE ASCIZ
    (3)                                     ;;69$:   .ASCIZ  <200>'MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED.''
    (3)  017172                             68$:
    (3)  017172  104401  017200                      TYPE    ,71$               ;;TYPE ASCIZ STRING
    (3)  017176  000434                              BR      70$                ;;GET OVER THE ASCIZ
    (3)                                     ;;71$:   .ASCIZ  <200>''THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED.''<200><200>
    (3)  017270                             70$:
    (2)
    (2)  017270  112737  177777  017422     5$:      MOVB    #0-1,11$           ;DAC CODE FOR SLAVE.
    (2)  017276  012501                              MOV     (5)+,R1            ;GET NEXT DEVICE ADDR.
    (2)  017300  021127  000000             6$:      CMP     (R1),#0            ;TERM REACHED?
    (2)  017304  001444                              BEQ     10$
    (2)  017306  105237  017422                      INCB    11$
    (2)  017312  113777  017422  162126              MOVB    11$,@KMAD4         ;FIFO DATA
    (2)  017320  004737  017424                      JSR     PC,20$             ;ISSUE SEND
    (2)  017324  112177  162116                      MOVB    (R1)+,@KMAD4       ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
    (2)  017330  004737  017424                      JSR     PC,20$             ;ISSUE SEND
    (2)  017334  112177  162106                      MOVB    (R1)+,@KMAD4       ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
    (2)  017340  004737  017424                      JSR     PC,20$
    (2)
    (2)  017344  032777  000002  162064     7$:      BIT     #BIT1,@KMAD0       ;WAIT FOR FIFO DATA
    (2)  017352  001374                              BNE     7$                 ;=1 NO DATA. =0 DATA.
    (2)  017354  112777  000002  162060              MOVB    #2,@KMAD2          ;READ FIFO.
    (2)
    (2)  017362                             8$:
    (3)  017362  004537  020366                      JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  017366  104000                              ERROR                      ;/TIME-OUT ERROR
    (3)                                                                         ;/WE FAILED TO COMPLETE
    (3)                                                                         ;/CURRENT OPERATION.
    (3)                                                                         ;/CONTINUES IN THIS LOOP
    (3)                                                                         ;/WOULD MAKE US ''HANG'' HERE
    (3)
    (3)  017370  000774                              BR              8$
    (3)
    (3)                                                                         ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  017372  122777  000377  162042              CMPB    #377,@KMAD2        ;WAIT FOR READ.
    (2)  017400  001370                              BNE     8$
    (2)  017402  105777  162040                      TSTB    @KMAD4             ;WAS A ZERO RETURNED?
    (2)  017406  001734                              BEQ     6$                 ;YES GET NEXT ADDR.
    (2)                                                                         ;SLAVE WILL RETURN CODE 0 IF
    (2)  017410  005237  017454                      INC     $AERR              ;DEV PRESENT.    ELSE
    (2)                                                                         ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
    (2)  017414  005041                              CLR     -(1)               ;GET RID OF REFERENCE TO BAD ADDR.
    (2)  017416  012601                     10$:     MOV     (SP)+,R1
    (2)  017420  000205                              RTS     R5                 ;RETURN ALL ADDR. CHECKED.
    (2)
    (2)  017422  000000                     11$:     .WORD   0                  ;HOLDS DAC CODE PLUS OFFSET
    (2)                                                                         ;TO SLAVES ADDR. TABLE.
    (2)
```

```
 (2)  017424  112777  000003  162010  20$:    MOVB    #3,@KMAD2             ;ISSUE FIFO WRITE
 (2)  017432                          21$:
 (3)  017432  004537  020366                  JSR     R5, $TOUT             ;-TOUT-CHECK FOR TIMEOUT
 (3)
 (3)  017436  104000                          ERROR                         ;/TIME-OUT ERROR
 (3)                                                                        ;/WE FAILED TO COMPLETE
 (3)                                                                        ;/CURRENT OPERATION.
 (3)                                                                        ;/CONTINUES IN THIS LOOP
 (3)                                                                        ;/WOULD MAKE US 'HANG'' HERE
 (3)
 (3)  017440  000774                          BR              21$
 (3)
 (3)                                                                        ;/RETURNS HERE-FROM-TIMED OUT.
 (2)  017442  122777  000377  161772          CMPB    #377,@KMAD2           ;KMC CODE WILL RETURN A '377''
 (2)  017450  001370                          BNE     21$                   ;WHEN DONE COMMAND.
 (2)  017452  000207                          RTS     PC
 (2)
 (2)  017454  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)  017456  010446                  $LOAD:  MOV     R4,-(SP)              ;SAVE R4.
 (2)  017460  010046                          MOV     R0,-(SP)              ;SAVE R0.
 (2)  017462  012500                  1$:     MOV     (5)+,R0               ;GET PROG. ADDR.
 (2)  017464  005077  161746                  CLR     @KMAD0                ;CLEAR CSR
 (2)  017470  005077  161752                  CLR     @KMAD4                ;CLEAR CRAM ADDR.
 (2)  017474  052777  002000  161734  2$:     BIS     #2000,@KMAD0          ;SELECT CRAM.
 (2)  017502  012077  161744                  MOV     (0)+,@KMAD6           ;WRITE DATA.
 (2)  017506  052777  020000  161722          BIS     #20000,@KMAD0         ;SET CRAM WRITE
 (2)  017514  005077  161716                  CLR     @KMAD0                ;DISABLE CRAM.
 (2)  017520  005277  161722                  INC     @KMAD4                ;UPDATE CRAM ADDR.
 (2)  017524  021027  177777                  CMP     (0),#-1               ;ALL DONE?
 (2)  017530  001361                          BNE     2$                    ;NO LOOP.
 (2)  017532  005077  161710                  CLR     @KMAD4                ;CLEAR CRAM ADDR.
 (2)  017536  016500  177776                  MOV     -2(5),R0              ;GET MICRO CODE ADDR.
 (2)
 (2)  017542  052777  002000  161666  3$:     BIS     #2000,@KMAD0          ;SELECT CRAM
 (2)  017550  022077  161676                  CMP     (R0)+,@KMAD6          ;DATA OK?
 (2)  017554  001013                          BNE     5$                    ;NO - REPORT AN ERROR.
 (2)  017556  021027  177777                  CMP     (0),#-1               ;ALL DONE?
 (2)  017562  001405                          BEQ     4$                    ;YES - EXIT
 (2)  017564  005077  161646                  CLR     @KMAD0                ;NO - DESELECT CRAM.
 (2)  017570  005277  161652                  INC     @KMAD4                ;UPDATE CRAM ADDR.
 (2)  017574  000762                          BR      3$
 (2)
 (2)  017576  012600                  4$:     MOV     (SP)+,R0              ;RESTORE R0
 (2)  017600  012604                          MOV     (SP)+,R4              ;RESTORE R4
 (2)  017602  000205                          RTS     R5                    ;EXIT
 (2)
 (2)  017604                          5$:                                   ;COME HERE ON LOAD ERROR
```

```
(2)  017604  005745                          TST     -(5)
(2)  017606  105204                          INCB    R4                  ;UPDATE ERROR COUNTER.
(2)  017610  100324                          BPL     1$                  ;IF NOT TOO MANY, TRY AGAIN.
(2)  017612  000000                          HALT                        ;MICRO CODE LOAD ERROR.
(2)                                                                      ;KMC-11 FAULT. YOU COULD TRY
(2)  017614  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)                                          ;
(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)  017616  010046                  $TLKW:  MOV     R0,-(SP)            ;SAVE R0
(2)  017620  012500                          MOV     (5)+,R0             ;GET DEVICE OFFSET
(2)  017622  052700  000340                  BIS     #340,R0             ;ADD WRITE CODE.
(2)  017626  004737  020100                  JSR     PC,$LPW             ;WAIT FOR FAST PATH READY
(2)  017632  010037  017724                  MOV     R0,W1
(2)  017636  010077  161604                  MOV     R0,@KMAD4
(2)  017642  112777  000005  161572          MOVB    #5,@KMAD2           ;ISSUE FAST PATH WRITE
(2)  017650  004737  020100                  JSR     PC,$LPW             ;WAIT FOR RDY
(2)  017654  011537  017726                  MOV     (5),W2
(2)  017660  112577  161562                  MOVB    (5)+,@KMAD4         ;WRITE LOW BYTE DATA.
(2)
(2)  017664  112777  000005  161550          MOVB    #5,@KMAD2           ;FP WRITE
(2)  017672  004737  020100                  JSR     PC,$LPW
(2)  017676  111537  017730                  MOVB    (5),W3
(2)  017702  112577  161540                  MOVB    (5)+,@KMAD4         ;WRITE HIGH BYTE
(2)  017706  112777  000005  161526          MOVB    #5,@KMAD2
(2)  017714  004737  020100                  JSR     PC,$LPW
(2)  017720  012600                          MOV     (SP)+,R0
(2)  017722  000205                          RTS     R5                  ;EXIT DONE.
(2)  017724  000000                  W1:     0
(2)  017726  000000                  W2:     0
(2)  017730  000000                  W3:     0
(2)
(2)                                          ;*
(2)                                          ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
(2)                                          ;*
(2)                                          ;*       CALL =  JSR     R5,$TLKR
(2)                                          ;*               .WORD   0                   ;OFFSET OF DEVICE
(2)                                          ;*               ;RETURNS HERE
(2)                                          ;*DATA IN WORD $DATR
(2)                                          ;*
(2)
(2)  017732  010046                  $TLKR:  MOV     R0,-(SP)            ;SAVE R0
(2)  017734  012500                          MOV     (5)+,R0             ;GET OFFSET
(2)  017736  052700  000300                  BIS     #300,R0             ;ADD READ CODE
(2)  017742  004737  020100                  JSR     PC,$LPW             ;WAIT TILL READY
(2)  017746  110077  161474                  MOVB    R0,@KMAD4
```

```
    (2)  017752  112777  000005  161462            MOVB    #5,@KMAD2          ;ISSUE WRITE FP
    (2)  017760  004737  020100                    JSR     PC,$LPW
    (2)  017764  010037  020074                    MOV     R0,RD1
    (2)  017770                            1$:
    (3)  017770  004537  020366                    JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  017774  104000                            ERROR                      ;/TIME-OUT ERROR
    (3)                                                                       ;/WE FAILED TO COMPLETE
    (3)                                                                       ;/CURRENT OPERATION.
    (3)                                                                       ;/CONTINUES IN THIS LOOP
    (3)                                                                       ;/WOULD MAKE US ''HANG'' HERE
    (3)
    (3)  017776  000774                            BR              1$
    (3)
    (3)                                                                       ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  020000  032777  000040  161430            BIT     #BIT5,@KMAD0       ;FAST PATH GOT DATA?
    (2)  020006  001370                            BNE     1$
    (2)  020010  112777  000004  161424            MOVB    #4,@KMAD2          ;ISSUE FAST PATH READ
    (2)  020016  004737  020100                    JSR     PC,$LPW
    (2)  020022  117737  161420  020076            MOVB    @KMAD4,$DATR       ;GET LOW BYTE
    (2)  020030                            2$:
    (3)  020030  004537  020366                    JSR     R5, $TOUT          ;-TOUT-CHECK FOR TIMEOUT
    (3)
    (3)  020034  104000                            ERROR                      ;/TIME-UUT ERROR
    (3)                                                                       ;/WE FAILED TO COMPLETE
    (3)                                                                       ;/CURRENT OPERATION.
    (3)                                                                       ;/CONTINUES IN THIS LOOP
    (3)                                                                       ;/WOULD MAKE US ''HANG'' HERE
    (3)
    (3)  020036  000774                            BR              2$
    (3)
    (3)                                                                       ;/RETURNS HERE-FROM-TIMED OUT.
    (2)  020040  032777  000040  161370            BIT     #BIT5,@KMAD0       ;FAST PATH READY?
    (2)  020046  001370                            BNE     2$
    (2)  020050  112777  000004  161364            MOVB    #4,@KMAD2          ;ISSUE FAST PATH READ
    (2)  020056  004737  020100                    JSR     PC,$LPW
    (2)  020062  117737  161360  020077            MOVB    @KMAD4,$DATR+1     ;SAVE HIGH BYTE
    (2)  020070  012600                            MOV     (SP)+,R0
    (2)  020072  000205                            RTS     R5
    (2)  020074  000000                    RD1:    0
    (2)  020076  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)  020100  010146                    $LPW:   MOV     R1,-(SP)           ;SAVE R1
    (2)  020102  005001                            CLR     R1
    (2)  020104  122777  000377  161330    1$:     CMPB    #377,@KMAD2        ;FINISHED INSTRUCTION?
    (2)  020112  001403                            BEQ     2$
    (2)  020114  005201                            INC     R1                 ;TIME OUT?
    (2)  020116  001372                             BNE     1$
    (2)  020120  000411                             BR      10$
    (2)
    (2)  020122  032777  000020  161306  2$:        BIT     #BIT4,@KMADO        ;FAST PATH READ?
    (2)  020130  001403                             BEQ     3$
    (2)  020132  005201                             INC     R1                  ;NO - TIME OUT?
    (2)  020134  001372                             BNE     2$
    (2)  020136  000402                             BR      10$                 ;YES - REPORT AN ERROR
    (2)
    (2)  020140  012601                  3$:        MOV     (SP)+,R1            ;RESTORE R1
    (2)  020142  000207                             RTS     PC                  ;EXIT
    (2)
    (2)  020144                          10$:
    (3)  020144  104401  020152                     TYPE    ,65$                ;;TYPE ASCIZ STRING
    (3)  020150  000407                             BR      64$                 ;;GET OVER THE ASCIZ
    (3)                                  ;;65$:     .ASCIZ  <200>#LPA-11 FAULT#
    (3)  020170                          64$:
    (2)
    (2)  020170  000000                  11$:       HALT                        ;LPA-11 FAULT RUN LPA-11
    (2)  020172  000776                             BR      11$                 ;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)  020174  010046                  $OUTLP:    MOV     R0,-(SP)            ;SAVE R0
    (2)  020176  010146                             MOV     R1,-(SP)            ;SAVE R1
    (2)
    (2)  020200  012700  001464                     MOV     #.DVLS,R0           ;PROGRAM DEFINED LIST.
    (2)  020204  005001                             CLR     R1
    (2)  020206  005710                  1$:        TST     (0)                 ;TERMINATOR REACHED?
    (2)  020210  001421                             BEQ     10$                 ;YES NEXT STEP.
    (2)  020212  027520  000000                     CMP     @(5),(0)+           ;MATCH WITH ADDR IN LIST?
    (2)  020216  001402                             BEQ     2$
    (2)  020220  005201                             INC     R1
    (2)  020222  000771                             BR      1$
    (2)
    (2)  020224  010137  020242          2$:        MOV     R1,3$               ;SAVE OFFSET, DEVICE KNOWN.
    (2)  020230  005725                             TST     (5)+
    (2)  020232  013537  020244                     MOV     @(5)+,4$            ;GET DATA TO BE WRITTEN
    (2)  020236  004537  017616                     JSR     R5,$TLKW            ;DO WRITE
    (2)  020242  000000                  3$:        .WORD   0                   ;DEVICE OFFSET
    (2)  020244  000000                  4$:        .WORD   0                   ;DATA TO BE WRITTEN.
    (2)  020246  012601                             MOV     (SP)+,R1
    (2)  020250  012600                             MOV     (SP)+,R0
    (2)  020252  000205                             RTS     R5
    (2)  020254  017520  000000          10$:       MOV     @(5),(0)+           ;SAVE ADDR.
    (2)  020260  005010                          CLR     (0)
    (2)  020262  004537  016576                  JSR     R5,$LPAI
    (2)  020266  001464                          .WORD   .DVLS
    (2)  020270  000755                          BR      2$
    (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)  020272  010046                  $INLP:  MOV     R0,-(SP)        ;SAVE R0
    (2)  020274  010146                          MOV     R1,-(SP)        ;SAVE R1
    (2)
    (2)  020276  012700  001464                  MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
    (2)  020302  005001                          CLR     R1
    (2)  020304  005710                  1$:     TST     (0)             ;EOL REACHED?
    (2)  020306  001420                          BEQ     10$             ;YES - DEFINE NEW ADDR.
    (2)
    (2)  020310  027520  000000                  CMP     @(5),(0)+       ;ADDR. MATCH?
    (2)  020314  001402                          BEQ     2$
    (2)  020316  005201                          INC     R1
    (2)  020320  000771                          BR      1$
    (2)
    (2)  020322  010137  020334          2$:     MOV     R1,3$           ;SAVE LIST OFFSET
    (2)  020326  005725                          TST     (5)+
    (2)  020330  004537  017732                  JSR     R5,$TLKR        ;GO READ DEVICE
    (2)          020334                  $OFS=.
    (2)  020334  000000                  3$:     .WORD   0               ;OFFSET OF DEVICE
    (2)
    (2)  020336  013735  020076                  MOV     $DATR,@(5)+     ;STORE DATA.
    (2)  020342  012601                          MOV     (SP)+,R1        ;RESTORE R1
    (2)  020344  012600                          MOV     (SP)+,R0        ;RESTORE R2
    (2)  020346  000205                          RTS     R5              ;EXIT
    (2)
    (2)  020350  017520  000000          10$:    MOV     @(5),(0)+
    (2)  020354  005010                          CLR     (0)
    (2)  020356  004537  016576                  JSR     R5,$LPAI
    (2)  020362  001464                          .WORD   .DVLS
    (2)  020364  000756                          BR      2$
    (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)  020366  020537  020422          $TOUT:  CMP     R5,$SAD          ;SAME ADDR?
 (2)  020372  001405                          BEQ     1$
 (2)  020374  010537  020422                  MOV     R5,$SAD          ;NO-SAVE THIS ADDR.
 (2)  020400  005037  020424                  CLR     $CNT             ;CLR CNT AT ADDR.
 (2)  020404  000403                          BR      2$
 (2)  020406  005237  020424          1$:     INC     $CNT             ;OVERFLOW?
 (2)  020412  100402                          BMI     3$               ;YES-ERROR RETURN
 (2)  020414  062705  000004          2$:     ADD     #4,R5            ;NO-NON ERROR RETURN
 (2)  020420  000205                  3$:     RTS     R5               ;RETURN.
 (2)
 (2)  020422  000000                  $SAD:   .WORD   0                ;CONTAINS LOOP ADDR.
 (2)  020424  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)  020426  000005                  $RESET: RESET                    ;RESET THE WORLD.
 (3)
 (3)                                  ;*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
 (2)  020440  005737  017454                  TST     $AERR            ;IF NO ERROR,LOOP
 (2)  020444  001004                          BNE     10$              ;THERE WAS AN ERROR.
 (2)  020446  062737  000002  020462          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)  020454  000764                          BR      $RESET
 (2)  020456                          10$:
 (2)  020456  000207                          RTS     PC
 (2)  020460  000000                  1$:     .WORD   0                ;JUNK LOC.
 (2)  020462  160000                  2$:     .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)  020464                          SDELAY:
 (2)  020464  005737  020546                  TST     RTCCSR           ;CLOCK PRESENT?
 (2)  020470  100016                          BPL     10$
```

```
    (2)  020472  012737  000002  020536                  MOV     #2,TIME
    (2)  020500  052777  000115  000040                  BIS     #115,@RTCCSR      ;START CLOCK
    (2)  020506  005037  177776                          CLR     PS
    (2)  020512  005737  020536             1$:          TST     TIME
    (2)  020516  001375                                  BNE     1$
    (2)  020520  005077  000022                          CLR     @RTCCSR           ;STOP CLOCK
    (2)
    (2)  020524  000207                                  RTS PC
    (2)  020526  105237  020536             10$:         INCB    TIME
    (2)  020532  001375                                  BNE     10$
    (2)  020534  000207                                  RTS     PC
    (2)
    (2)  020536  000000                     TIME:        .WORD   0
    (2)
    (2)  020540  005337  020536             CLKINT: DEC     TIME
    (2)  020544  000002                                  RTI
    (2)  020546  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)  020550                             $UTK:
    (2)  020550  005037  001464                          CLR     .DVLS
    (2)  020554                             21$:
    (3)  020554  104401  020562                          TYPE    ,65$              ;;TYPE ASCIZ STRING
    (3)  020560  000405                                  BR      64$               ;;GET OVER THE ASCIZ
    (3)                                     ;;65$:       .ASCIZ  <200>#E OR D?#
    (3)  020574                             64$:
    (2)  020574  105777  160344             1$:          TSTB    @$TKS
    (2)  020600  100375                                  BPL     1$
    (2)  020602  117737  160340  020724                  MOVB    @$TKB,20$         ;GET INPUT
    (2)  020610  104401  020724                          TYPE,   20$               ;ECHO, NEXT MESSAGE.
    (2)  020614  142737  000240  020724                  BICB    #240,20$          ;STRIP PARITY, LC
    (2)  020622  104407                                  RDOCT                     ;GET ADDR.
    (2)  020624  012637  020722                          MOV     (SP)+,14$
    (2)  020630  123727  020724  000104                  CMPB    20$,#'D           ;DEPOSIT?
    (2)  020636  001411                                  BEQ     10$
    (2)
    (2)  020640  004537  020272                          JSR     R5,$INLP          ;GET DATA
    (2)  020644  020722                     2$:          .WORD   14$
    (2)  020646  020660                                  .WORD   5$
    (2)
    (3)  020650  013746  020660                          MOV     5$,-(SP)          ;;SAVE 5$ FOR TYPEOUT
    (3)  020654  104402                                  TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (2)  020656  000736                                  BR      21$               ;LOOP.
    (2)  020660  000000                     5$:          .WORD   0
    (2)
    (2)  020662                             10$:
    (3)  020662  104401  020670                          TYPE    ,67$              ;;TYPE ASCIZ STRING
```

```
   (3)  020666  000404                          BR      66$             ;;GET OVER THE ASCIZ
   (3)                                  ;;67$:  .ASCIZ  <200>#DATA= #
   (3)  020700                          66$:
   (2)  020700  104407                          RDOCT
   (2)  020702  012637  020720                  MOV     (SP)+,13$
   (2)
   (2)  020706  004537  020174          11$:    JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
   (2)  020712  020722                  12$:    .WORD   14$             ;DEVICE ADDR.
   (2)  020714  020720                          .WORD   13$             ;DATA
   (2)  020716  000716                          BR      21$
   (2)
   (2)  020720  000000                  13$:    .WORD   0
   (2)  020722  000000                  14$:    .WORD   0
   (2)  020724  100001  042504  044526  20$:    .ASCIZ  <1><200>#DEVICE ADDR= #
   (2)  020732  042503  040440  042104
   (2)  020740  036522  000040
   (2)                                          .EVEN
   (2)
   (1)
   (1)
   (2)                                          ;
   (2)                                          ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
   (2)                                          ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
   (2)                                          ;TO SET UP THE USER PROGRAM TO LINK TO FILE 'DRLPX2'' FOR
   (2)                                          ;SAMPLE TAKEING PURPOSES.
   (2)                                          ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
   (2)                                          ;       A/D CSR IN BSEL 4,AND 5.
   (2)                                          ;       (2) HE MUST CALL THIS ROUTINE:
   (2)                                          ;               JSR     R5,$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)  020744  012537  020754          $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
   (2)  020750  004537  020272                  JSR     R5,$INLP
   (2)  020754  000000                  1$:     .WORD   0
   (2)  020756  021052                          .WORD   10$
   (2)  020760  113777  020334  160464          MOVB    $OFS,@KMAD6
   (2)  020766  113777  020334  160460          MOVB    $OFS,@KMAD7
   (2)  020774  013737  020754  021014          MOV     1$,2$
   (2)  021002  062737  000002  021014          ADD     #2,2$
   (2)  021010  004537  020272                  JSR     R5,$INLP
   (2)  021014  000000                  2$:     .WORD   0
   (2)  021016  021052                          .WORD   10$
   (2)  021020  113777  020334  160416          MOVB    $OFS,@KMAD3
   (2)  021026  152777  000340  160416          BISB    #340,@KMAD6
   (2)  021034  152777  000300  160412          BISB    #300,@KMAD7
   (2)  021042  152777  000300  160374          BISB    #300,@KMAD3
   (2)  021050  000205                          RTS     R5
```

```
 (2)  021052  000000                    10$:     .WORD   0
 (2)
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 (1)
 (2)                                    ;;*************************************************************
 (1)                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 (1)                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 (1)                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 (1)                                    ;*CALL:
 (1)                                    ;*       MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
 (1)                                    ;*       TYPOS                      ;;CALL FOR TYPEOUT
 (1)                                    ;*       .BYTE   N                  ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 (1)                                    ;*       .BYTE   M                  ;;M=1 OR 0
 (1)                                    ;*                                            ;;1=TYPE LEADING ZEROS
 (1)                                    ;*                                            ;;0=SUPPRESS LEADING ZEROS
 (1)                                    ;*
 (1)                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 (1)                                    ;*$TYPOS OR $TYPOC
 (1)                                    ;*CALL:
 (1)                                    ;*       MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
 (1)                                    ;*       TYPON                      ;;CALL FOR TYPEOUT
 (1)                                    ;*
 (1)                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 (1)                                    ;*CALL:
 (1)                                    ;*       MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
 (1)                                    ;*       TYPOC                      ;;CALL FOR TYPEOUT
 (1)
 (1)  021054  017646  000000            $TYPOS: MOV      @(SP),-(SP)        ;;PICKUP THE MODE
 (1)  021060  116637  000001  021277            MOVB     1(SP),$0FILL       ;;LOAD ZERO FILL SWITCH
 (1)  021066  112637  021301                    MOVB     (SP)+,$OMODE+1     ;;NUMBER OF DIGITS TO TYPE
 (1)  021072  062716  000002                    ADD      #2,(SP)            ;;ADJUST RETURN ADDRESS
 (1)  021076  000406                            BR       $TYPON
 (1)  021100  112737  000001  021277    $TYPOC: MOVB     #1,$0FILL          ;;SET THE ZERO FILL SWITCH
 (1)  021106  112737  000006  021301            MOVB     #6,$OMODE+1        ;;SET FOR SIX(6) DIGITS
 (1)  021114  112737  000005  021276    $TYPON: MOVB     #5,$OCNT           ;;SET THE ITERATION COUNT
 (1)  021122  010346                            MOV      R3,-(SP)           ;;SAVE R3
 (1)  021124  010446                            MOV      R4,-(SP)           ;;SAVE R4
 (1)  021126  010546                            MOV      R5,-(SP)           ;;SAVE R5
 (1)  021130  113704  021301                    MOVB     $OMODE+1,R4        ;;GET THE NUMBER OF DIGITS TO TYPE
 (1)  021134  005404                            NEG      R4
 (1)  021136  062704  000006                    ADD      #6,R4              ;;SUBTRACT IT FOR MAX. ALLOWED
 (1)  021142  110437  021300                    MOVB     R4,$OMODE          ;;SAVE IT FOR USE
 (1)  021146  113704  021277                    MOVB     $0FILL,R4          ;;GET THE ZERO FILL SWITCH
 (1)  021152  016605  000012                    MOV      12(SP),R5          ;;PICKUP THE INPUT NUMBER
 (1)  021156  005003                            CLR      R3                 ;;CLEAR THE OUTPUT WORD
 (1)  021160  006105                    1$:     ROL      R5                 ;;ROTATE MSB INTO "C"
 (1)  021162  000404                            BR       3$                 ;;GO DO MSB
 (1)  021164  006105                    2$:     ROL      R5                 ;;FORM THIS DIGIT
 (1)  021166  006105                            ROL      R5
 (1)  021170  006105                            ROL      R5
 (1)  021172  010503                            MOV      R5,R3
 (1)  021174  006103                    3$:     ROL      R3                 ;;GET LSB OF THIS DIGIT
 (1)  021176  105337  021300                    DECB     $OMODE             ;;TYPE THIS DIGIT?
 (1)  021202  100016                            BPL      7$                 ;;BR IF NO
 (1)  021204  042703  177770                    BIC      #177770,R3         ;;GET RID OF JUNK
 (1)  021210  001002                            BNE      4$                 ;;TEST FOR 0
```

```
    (1)  021212  005704                          TST     R4              ;;SUPPRESS THIS 0?
    (1)  021214  001403                          BEQ     5$              ;;BR IF YES
    (1)  021216  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
    (1)  021220  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
    (1)  021224  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
    (1)  021230  110337  021274                  MOVB    R3,8$           ;;SAVE FOR TYPING
    (1)  021234  104401  021274                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
    (1)  021240  105337  021276          7$:     DECB    $OCNT           ;;COUNT BY 1
    (1)  021244  003347                          BGT     2$              ;;BR IF MORE TO DO
    (1)  021246  002402                          BLT     6$              ;;BR IF DONE
    (1)  021250  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
    (1)  021252  000744                          BR      2$              ;;GO DO THE LAST DIGIT
    (1)  021254  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
    (1)  021256  012604                          MOV     (SP)+,R4        ;;RESTORE R4
    (1)  021260  012603                          MOV     (SP)+,R3        ;;RESTORE R3
    (1)  021262  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
    (1)  021270  012616                          MOV     (SP)+,(SP)
    (1)  021272  000002                          RTI                     ;;RETURN
    (1)  021274     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
    (1)  021275     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
    (1)  021276     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
    (1)  021277     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
    (1)  021300  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                                .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)
 (1)   021302  010046                           $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 (1)   021304  016600  000002                           MOV     2(SP),R0        ;;GET TRAP ADDRESS
 (1)   021310  005740                                   TST     -(R0)           ;;BACKUP BY 2
 (1)   021312  111000                                   MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 (1)   021314  006300                                   ASL     R0              ;;POSITION FOR INDEXING
 (1)   021316  016000  021336                           MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 (1)   021322  000200                                   RTS     R0              ;;GO TO ROUTINE
 (1)
 (1)
 (1)                                            ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
 (1)
 (1)   021324  011646                           $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
 (1)   021326  016666  000004  000002                   MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
 (1)   021334  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)   021336  021324                           $TRPAD: .WORD   $TRAP2
 (3)   021340  016046                                   $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
 (3)   021342  021100                                   $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 (3)   021344  021054                                   $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 (3)   021346  021114                                   $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 (1)
 (1)
 (3)   021350  014644                                   $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
 (3)   021352  014764                                   $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
 (3)   021354  015136                                   $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
 4392                                        .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                        ;;*****************************************************************
  (1)                                        ;POWER DOWN ROUTINE
  (1)  021356  012737  021522  000024        $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  021364  012737  000340  000026                MOV     #340,@#PWRVEC+2  ;;PRIO:7
  (3)  021372  010046                                MOV     R0,-(SP)          ;;PUSH R0 ON STACK
  (3)  021374  010146                                MOV     R1,-(SP)          ;;PUSH R1 ON STACK
  (3)  021376  010246                                MOV     R2,-(SP)          ;;PUSH R2 ON STACK
  (3)  021400  010346                                MOV     R3,-(SP)          ;;PUSH R3 ON STACK
  (3)  021402  010446                                MOV     R4,-(SP)          ;;PUSH R4 ON STACK
  (3)  021404  010546                                MOV     R5,-(SP)          ;;PUSH R5 ON STACK
  (3)  021406  017746  157526                        MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
  (1)  021412  010637  021526                        MOV     SP,$SAVR6         ;;SAVE SP
  (1)  021416  012737  021430  000024                MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  021424  000000                                HALT
  (1)  021426  000776                                BR      .-2               ;;HANG UP
  (1)
  (2)                                        ;;*****************************************************************
  (1)                                        ;POWER UP ROUTINE
  (1)  021430  012737  021522  000024        $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
  (1)  021436  013706  021526                        MOV     $SAVR6,SP         ;;GET SP
  (1)  021442  005037  021526                        CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
  (1)  021446  005237  021526                1$:     INC     $SAVR6            ;;WAIT FOR THE INC
  (1)  021452  001375                                BNE     1$                ;;OF  WORD
  (3)  021454  012677  157460                        MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
  (3)  021460  012605                                MOV     (SP)+,R5          ;;POP STACK INTO R5
  (3)  021462  012604                                MOV     (SP)+,R4          ;;POP STACK INTO R4
  (3)  021464  012603                                MOV     (SP)+,R3          ;;POP STACK INTO R3
  (3)  021466  012602                                MOV     (SP)+,R2          ;;POP STACK INTO R2
  (3)  021470  012601                                MOV     (SP)+,R1          ;;POP STACK INTO R1
  (3)  021472  012600                                MOV     (SP)+,R0          ;;POP STACK INTO R0
  (1)  021474  012737  021356  000024                MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)  021502  012737  000340  000026                MOV     #340,@#PWRVEC+2  ;;PRIO:7
  (1)  021510  104401                                TYPE                      ;REPORT THE POWER FAILURE
  (1)  021512  021530                        $PWRMG: .WORD   $POWER            ;;POWER FAIL MESSAGE POINTER
  (1)  021514  012716                                MOV     (PC)+,(SP)        ;;RESTART AT BEGIN
  (1)  021516  001714                        $PWRAD: .WORD   BEGIN             ;;RESTART ADDRESS
  (1)  021520  000002                                RTI
  (1)  021522  000000                        $ILLUP: HALT                      ;;THE POWER UP SEQUENCE WAS STARTED
  (1)  021524  000776                                BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)  021526  000000                        $SAVR6: 0                         ;;PUT THE SP HERE
  (1)  021530  005015  047520  042527        $POWER: .ASCIZ  <15><12>'POWER'
  (1)  021536  000122
  (1)                                                .EVEN
 4393                                        .EVEN
 4394  021540  000310                        DIST:   .BLKW   200.              ;STATE-WIDTH DISTRIBUTION
 4395
 4396          042000                                .=42000
 4397                                        ;THE MICRO-CODE FOR THIS PROGRAM RESIDES HERE.
 4398          042300                                .=42300
 4399
 4400  042300  010000                        BUFFER: .BLKW   4096.             ;BUFFER AREA
 4401
 4402          000001                        .END
```

```
ABASE = 170400          2938#   2996    3037    3038
ACDW1 = 000000          2996
ACDW2 = 000000          2996
ACPUOP= 000000          2996
ADBUFF  001320          3038#   3257    3265    3283    3318    3324    3369    3381    3502    3561*   3569*   3570*   3638
                        3688    3700    3739    4122    4143
ADDW0 = 000000          2996
ADDW1 = 000000          2996
ADDW10= 000000          2996
ADDW11= 000000          2996
ADDW12= 000000          2996
ADDW13= 000000          2996
ADDW14= 000000          2996
ADDW15= 000000          2996
ADDW2 = 000000          2996
ADDW3 = 000000          2996
ADDW4 = 000000          2996
ADDW5 = 000000          2996
ADDW6 = 000000          2996
ADDW7 = 000000          2996
ADDW8 = 000000          2996
ADDW9 = 000000          2996
ADEVCT= 000000          2996
ADEVM = 000000          2996
AENV  = 000000          2996
AENVM = 000000          2996
AFATAL= 000000          2996
AGAIN   007122          3800#   3832
AGATST  012004          4281#   4285
AGTST   012016          3524*   3542*   3554*   4282    4283#
AMADR1= 000000          2996
AMADR2= 000000          2996
AMADR3= 000000          2996
AMADR4= 000000          2996
AMAMS1= 000000          2996
AMAMS2= 000000          2996
AMAMS3= 000000          2996
AMAMS4= 000000          2996
AMSG    012243          3099    4293#
AMSGAD= 000000          2996
AMSGLG= 000000          2996
AMSGTY= 000000          2996
AMTYP1= 000000          2996
AMTYP2= 000000          2996
AMTYP3= 000000          2996
AMTYP4= 000000          2996
APASS = 000000          2996
APRIOR= 000300          2940#   2996    3039
APTCSU= 000040          4385    4386#
APTENV= 000001          4382    4385    4386#
APTSIZ= 000200          3122    4386#
APTSPO= 000100          4385    4386#
AROUND  007262          3828    3831#
ASKCH   013636          3599    4336#
ASWREG= 000000          2996
ATESTN= 000000          2996
```

```
ATMSG   012370          3634     4303#
AUNIT = 000000          2996
AUSWR = 000000          2996
AVECT1= 140340          2939#    2996     3040     3079
AVECT2= 000000          2996
BASEBR  001322          3039#    3573*    3574*
BASECH  001332          3043#    3311     3429     3430     3636     3665     3666     3791     4083     4118     4236
BEGIN   001714          2988     3118#    4283     4392
BEGINA  005156          3166     3528#
BEGINC  004656          3169     3484#
BEGINL  002710          3218#    3521     3532
BEGINN  005540          3175     3597#
BEGINS  005610          3178     3608#
BEGINW  005250          3181     3546#
BEGIN2  001722          2990     3120#
BEGL    005114          3157     3172     3518#
BEG2    002404          2989     3096     3154#
BITPNT  001412          3067#    3733*    3738     3764     3765*
BIT0  = 000001          2936#    3260     3270     3275
BIT00 = 000001          2936#
BIT01 = 000002          2936#
BIT02 = 000004          2936#
BIT03 = 000010          2936#
BIT04 = 000020          2936#
BIT05 = 000040          2936#
BIT06 = 000100          2936#
BIT07 = 000200          2936#
BIT08 = 000400          2936#    4381
BIT09 = 001000          2936#    4381     4382
BIT1  = 000002          2936#    4387
BIT10 = 002000          2936#    4382
BIT11 = 004000          2936#    3727     4381     4387
BIT12 = 010000          2936#    4021
BIT13 = 020000          2936#    3490     3503     4382
BIT14 = 040000          2936#    3225     4381     4387
BIT15 = 100000          2936#    3244     3254     3274     4264     4387
BIT2  = 000004          2936#
BIT3  = 000010          2936#
BIT4  = 000020          2936#    3240     3313     3316     4387
BIT5  = 000040          2936#    3235     4387
BIT6  = 000100          2936#    3230
BIT7  = 000200          2936#    3251     3274     3316     4135
BIT8  = 000400          2936#    3221
BIT9  = 001000          2936#    3249     3277     3315
BK3     005642          3615#    3626
BPTVEC= 000014          2936#
BUFFER  042300          3774     3817*    3819*    3835     3863     3929     3941     3943     3963     3975     3976     3977     3999
                        4002     4400#
BUFF1   013757          3911     4345#
BUFF2   014005          3998     4346#
BUMPAD  005322          3522     3540     3552     3558#
BYPASS  005474          3566     3580#
B10     011434          4187#    4188
B11     011470          4197#    4198
CH      012323          3492     4299#
CHAN    012405          4054     4305#
```

```
                                                            J  7
LPA-AD11K TEST MD-11-CRLPKB       MACY11 30G(1063)  08-AUG-79  10:19  PAGE 49-2
CRLPKB.P11     08-AUG-79 10:18              CROSS REFERENCE TABLE -- USER SYMBOLS                                  SEQ 0087

CHANL    001362             3055#    3429*    3616*    3689     3744     4027*    4028     4055     4082*    4083*    4084     4119*    4370
CHANNL   007042             3790#
CH1      001350             3050#    3600*    3601*    3604     3611*    3633     3635     3657     3663*    3665*
CH2      001352             3051#    3614*    3616     3629     3653     3664*    3666*
CLEAR1   006770             3776#    3778
CLEAR2   007034             3787#    3789
CLKINT   020540             4387#
CMSG     012250             3093     4294#
CNNO     006746             3726*    3730*    3743     3769#
COMPAR   011314             3305     3329     3337     3345     3354     3362     3374     3397     3409     3417     3424     3446     4156#
CONV     006300             3697#    3702
CONVR    007150             3804     3807#
CONVRT   011072             3303     3327     3335     3343     3352     3360     3371     3383     3407     3415     3422     3639     4117#
CR     = 000015             2936#    4385
CRLF   = 000200             2936#    4385
C0       013675             3130     4337#
C1       013700             3714     4338#
C2       013702             3920     4004     4339#
C3       013705             4001     4340#
DAC      001404             3064#    3434     3437     3656     3660     3734*    3738*    3739     3764*    4070     4073     4076     4079
DASH     012277             3866     4297#
DAWAIT   004646             3370     3382     3406     3478#
DDISP  = 177570             2936#    2996     3122
DECPNT   014576             3871*    3900*    4364#
DECTYP   011504             3444     3627     3867     3873     3881     3885     3888     3899     3950     4048     4051     4206#
DELAY    001406             3065#    3740*    3741*
DELAY1   007230             3813     3821#
DELAY2   007236             3815     3822     3823#
DELAY3   007144             3805#    3806
DELCLR   010342             3909     3921     4005     4015#
DF1      014642             3013     3020     3026     3032     4373#
DH1      014417             3011     4359#
DH2      014455             3030     4360#
DH3      014540             3018     3024     4361#
DIFLIN   006750             3475     3772#
DISPLA   001142             2996#    3122*    3505*    4381*    4382*
DISPRE   000174             2988#    3122
DIST     021540             3779     3846*    3912     4394#
DONE     012422             4306#
DRLPX2=  ****** G             52#    4387
DSWR   = 177570             2936#    2996     3122
DT1      014602             3012     4369#
DT2      014614             3031     4370#
DT3      014632             3019     3025     4371#
DUMMY    001360             3054#    3430*    3653*    3657*    3746     4028*    4084*
EDGE     001410             3066#    3431*    3671*    3674*    3695*    3700*    3703*    3704*    3705*    3706*    3709     3757     4090*
                            4093*
EDGFLG   006450             3624*    3679*    3711     3717#    4031*    4098*
EDINT    001430             3074#    3519*    3529*    3547*    3551*
EMTVEC=  000030             2936#    3122*
EM1      014257             3010     4355#
EM2      014305             3017     4356#
EM3      014335             3023     4357#
EM4      014366             3029     4358#
ER       010564             4057     4059     4062#
ERMSG    012511             3451     3647     3877     3892     3904     3958     4062     4311#
```

```
                                                                 K  7
LPA-AD11K TEST MD-11-CRLPKB      MACY11 30G(1063)  08-AUG-79  10:19  PAGE 49-3
CRLPKB.P11     08-AUG-79 10:18            CROSS REFERENCE TABLE -- USER SYMBOLS                              SEQ 0088

ERR      006052             3643      3647#
ERRVEC=  000004             2936#     3122*     4381*
FIRST    001342             3047#     3785*     3858      3860*
FIXADR   005372             3559      3567#
FIXONE   005376             3146      3568#
FLAG     001400             3062#     3125*     3142*     3907      3961
GETDAT   010120             3966#     3974
GETEDG   006226             3617      3667      3688#     4029      4085
GMSG     012255             3107      4295#
GNS    = ****** U           2988      4387      4390
HAFMSG   012701             3901      4316#
HALF     007600             3893      3895#
HEAD1    014040             3145      4348#
HEAD5    013617             3486      4335#
HT     = 000011             2936#     4385
HUNS     014575             4215*     4227*     4229*     4232      4363#
INRNGE   007324             3842      3845#
IOTVEC=  000020             2936#     3122*
ISERV    001550             3088#     3148
KBVECT   001334             3044#     3147      3582
KMAD0    001436             3079#     3124      4387*
KMAD1    001440             3079#     3124
KMAD2    001442             3079#     3750*     3751      3754*     3755      3809*     3810      4387*
KMAD3    001444             3079#     4387*
KMAD4    001446             3079#     3749*     3753*     3757      3808*     3812      4387*
KMAD5    001450             3079#
KMAD6    001452             3079#     4387*
KMAD7    001454             3079#     3124      4387*
LAST     007442             3856      3869#
LEND     004644             3474      3476#
LESS     011422             4182      4184#
LF     = 000012             2936#     4385
LINEA    013576             3951      4334#
LOAD     010320             4000      4003      4007#
LOADY    011402             3915      3917      4010      4178#
LOAD0    010324             4008#     4012
LO2      010316             3962      4006#
LPADH    001450             3079#
LPADL    001446             3079#
LPCI     001436             3079#
LPCO     001442             3079#
LPMR     001440             3079#
LPMS1    001452             3079#
LPMS2    001454             3079#
LPSO     001444             3079#
LSB      012333             3628      4301#
LSBMSG   012270             3868      4296#
MAT      013730             3630      4343#
MAX      001420             3070#     3378*     3385      3389*     3982*     3987      3989*     3993
MAXTST   010216             3985      3987#
MEND     013001             3533      4319#
MESP     012763             4052      4318#
MESR     012750             4049      4317#
METST    013735             4344#
MIN      001414             3068#     3390*     3439      3981*     3984      3986*     3992
MINUS    012237             4208      4291#
```

```
MLSB     013722           3445      4342#
MOFSET   013707           3442      4341#
MSG16    013156           3910      4328#
MSG18    013064           3997      4322#
MSG20    013124           3772      4327#
MSG21    013551           3928      4333#
MSG50    013043           3202      4321#
MSG71    013503           3158      4332#
MYTEMP   001426           3073#     3250*     3254*     3257      3260*     3261      3263      3265      3270*     3271      3272      3275*     3276
                          3283      3313*     3314      3317*     3318      3324      3325*     3368*     3369      3380*     3381      3485*     3496*
                          3498*     3500      3502      3637*     3638      3691*     3697*     3698      3700      3795*     4122      4123*     4129
                          4130*     4131      4136      4141      4143      4144
NAR      007510           3878      3880#
NARMSG   012550           3882      4313#
NARROW   001340           3046#     3784*     3850*     3880      3895
NBEXT    001354           3052#     3191*     3197*     3201      3203*     3204      3558      3565*     3578*
NEXT     007126           3801#     3827
NMBEXT   001356           3053#     3204*     3578
NOI      012375           4040      4304#
NOIA     010572           4030      4066#
NOIMSG   012132           3598      4289#
NOITST   010400           3603      4027#
NOI1     010602           4068#     4089
NOI8     010664           3457      3459      3461      4082#
NONE     001706           3104      3106      3114#
NOTNAR   007360           3849      3855#
NOTNEW   010010           3938      3943#
NOTOK    007254           3825      3829#
NOVT55   002332           3133      3136      3141      3145#     3213
NXTCMP   010202           3983#     3991
NXTCVT   006634           3748#     3761
NXTSTA   007752           3930#     3944
NXTY1    007706           3914#     3919
NXT8     010164           3979#     3996
OFFERR   004512           3449      3451#
OFFOK    004520           3450      3453#
OKAY     007246           3818      3820      3826#     3830
OKAYD    011530           4211      4213#
OKMSG    012500           3453      3644      3879      3894      3906      3960      4060      4309#
ONAD     013025           4320#
ONES     014600           4213*     4219*     4220      4222*     4231*     4366#
OUT      001424           3072#     3773*     3782*     3843*     3884      3887      3890      3897
OUTMSG   012646           3889      4315#
PEAK     001376           3061#     4044      4046*     4050      4058      4067*     4076*     4079*     4087*     4096*     4097*
PERCNT   001422           3071#     3723*     3724*     3725*     3731*     3732*     3762
PIRQ   = 177772           2936#
PIRQVE=  000240           2936#
PLUS     007772           3935      3937#
PLUSR2   011410           4179      4181#
POPRO    001702           3112#     3115
POS      011516           4207      4210#
POSPEA   010466           4045      4047#
POSRMS   010454           4042      4044#
POSR2    005700           3621      3623#
PRO    = 000000           2936#
PR1    = 000040           2936#
```

```
                                                       M  7
LPA-AD11K TEST MD-11-CRLPKB      MACY11 30G(1063)  08-AUG-79  10:19  PAGE 49-5
CRLPKB.P11     08-AUG-79 10:18          CROSS REFERENCE TABLE -- USER SYMBOLS                          SEQ 0090

PR2    = 000100       2936#
PR3    = 000140       2936#
PR4    = 000200       2936#
PR5    = 000240       2936#
PR6    = 000300       2936#
PR7    = 000340       2936#
PS     = 177776       2936#    4387*
PSW    = 177776       2936#    3161*    3487*    4172*
PWRVEC= 000024        2936#    3122*    4392*
QUEST    012241       3114     3182     4292#
RANDY    011010       3801     4105#
RBEG     001730       3119     3121#
RDCHR  = 104405       4377     4390#
RDLIN  = 104406       3159     4379     4390#
RDOCT  = 104407       3110     3610     3613     4387     4390#
RD1      020074       4387#*
READ     007276       3836#    3870
RELACC   007736       3908     3926#
REST1    002336       3127     3146#
RESVEC= 000010        2936#
RET      006446       3712     3716#
RETERR   003434       3293     3295#
RETURN   001704       3113#    3799     4127
RMS      001374       3060#    4041     4043*    4047     4056     4066*    4070*    4073*    4086*    4094*    4095*
RNA      001366       3057#    3150*    3802     4105*    4106*    4107*    4108     4111
RNB      001370       3058#    3151*    4105     4108*    4109*    4110*    4112
RNC      001372       3059#    3152*    4106     4109     4111*    4112*    4113*
RST      011362       3095     3101     4170#
RTCCSR   020546       4387#*
R2POS    006210       3676     3678#
SARSUB   006452       3432     3435     3654     3658     3723#    4068     4071     4074     4077
SAR1     006524       3728     3729     3733#
SDELAY   020464       4387#
SETAA    006160       3670#    3673
SETCH    012346       3632     4302#
SETMSG   012152       3609     4290#
SET1A    006060       3619     3620     3653#    3670
SET8     006122       3465     3469     3663#
SHIFT    006736       3763     3765#
SKIPST   001344       3048#    3786*    3853*    3872     3875
SKPMSG   012526       3874     4312#
SLASH    012435       3109     3953     4307#
SPACE    012326       3507     4300#
SPREAD   001402       3063#    4157*    4164     4370
STACK  = 001100       2936#    3094     3100     3122     3154
STATE    012303       3861     4298#
STKLMT= 177774        2936#
STREG    001316       3037#    3250     3254     3261     3263     3271     3272     3276     3290     3291     3314     3325     3485
                      3496     3498     3500     3534     3560*    3568*    3691     3697     3698     3736     3794     3795     4123
                      4129     4131     4141     4369     4370     4371
SWDIST   007654       3905     3907#
SWR      001140       2996#    3105     3108     3111*    3122*    3488     3490     3503     3600     3727     4021     4381     4382
                      4392*
SWREG    000176       2988#    3122
SW0    = 000001       2936#
SW00   = 000001       2936#
```

```
SW01   = 000002           2936#
SW02   = 000004           2936#
SW03   = 000010           2936#
SW04   = 000020           2936#
SW05   = 000040           2936#
SW06   = 000100           2936#
SW07   = 000200           2936#
SW08   = 000400           2936#
SW09   = 001000           2936#
SW1    = 000002           2936#
SW10   = 002000           2936#
SW11   = 004000           2936#
SW12   = 010000           2936#
SW13   = 020000           2936#
SW14   = 040000           2936#
SW15   = 100000           2936#
SW2    = 000004           2936#
SW3    = 000010           2936#
SW4    = 000020           2936#
SW5    = 000040           2936#
SW6    = 000100           2936#
SW7    = 000200           2936#
SW8    = 000400           2936#
SW9    = 001000           2936#
TADDR    001364           3056#    3102     3484*    3518*    3528*    3546*    3597*    3608*
TBITVE=  000014           2936#
TEMP     001346           3049#    3373     3387     3393     3394     3395*    3434*    3437*    3438*    3441*    3443     3456*    3468*
                          3602*    3615*    3641     3688     3824*    3829*    4121*    4144*    4149*    4150*    4151*    4152*    4158
TENS     014577           4214*    4223*    4224     4226*    4230*    4365#
TEST     003410           3255     3266     3280     3291#    3322
TESTAD   002552           3188#    3520     3531     3549
TESTIT   003400           3222     3231     3236     3241     3245     3290#
TESTR2   011544           4216#    4221     4225     4228
TIME     020536           4387#*
TKVEC  = 000060           2936#
TOMSG    012437           3612     4308#
TPVEC  = 000064           2936#
TRAPVE=  000034           2936#    3122*
TRTVEC=  000014           2936#
TRY      006544           3737#    3766
TRYAG    002426           3159#    3183
TST1     002710           3219#    3220
TST10    003252           3269#
TST11    003436           3299#
TST12    003510           3310#
TST13    003636           3312     3326#
TST14    003666           3334#
TST15    003716           3342#
TST16    003746           3351#
TST17    003776           3359#
TST2     002754           3228#
TST20    004026           3367#
TST21    004274           3414#
TST22    004324           3421#
TST23    004354           3428#
TST24    004524           3452     3455#
```

```
TST25    004564          3464#
TST26    004622          3472#
TST3     003000          3234#
TST4     003016          3239#
TST5     003034          3243#
TST6     003052          3248#
TST7     003166          3259#
TST8     010230          3988     3990#
TYPBAD   007372          3844     3854     3858#
TYPE   = 104401          3093     3099     3107     3109     3114     3130     3145     3158     3182     3202     3442     3445     3451
                         3453     3486     3492     3507     3514     3533     3538     3598     3599     3609     3612     3628     3630
                         3632     3634     3644     3647     3714     3772     3861     3866     3868     3874     3877     3879     3882
                         3886     3889     3892     3894     3901     3904     3906     3910     3911     3920     3928     3951     3953
                         3958     3960     3997     3998     4001     4004     4024     4040     4049     4052     4054     4060     4062
                         4208     4232     4285     4377     4382     4383     4385     4387     4388     4390#    4392
TYPEDG   006406          3631     3709#    4053
TYPOC  = 104402          4383     4387     4390#
TYPON  = 104404          4390#
TYPOS  = 104403          3108     3201     3493     3508     3535     3629     3633     3641     3710     3715     3865     3952     3955
                         4055     4390#
TYPOUT   011620          4217     4229#
TYPRP    010436          4032     4040#    4099
TYPSET   005716          3625     3627#    3680
UNEXP    001526          3081#    3229
VADR     001326          3041#    3198     3560     3561
VARLT1   011754          4242     4266#
VARLT2   011764          4244     4271#
VARLT3   011774          4238     4276#
VECTOR   001456          3079#    3229*    3562*    3571*    3572*    3575     3799*    4127*
VECTPS   001460          3079#
VECTR1   001324          3040#    3563*    3564*    3575*    3576*    3577*
VERSN    001462          3079#
VLIN     011750          3956     4263#
VNP      011744          4058     4261#
VNR      011742          4056     4235     4260#
VSET     011746          3642     4262#
VTFLG    002656          3131     3134     3137     3207#
VTINIT   014025          4024     4347#
VT55     002326          3139     3142#
VVCT     001330          3042#    3562     3563
V1       011720          3331     4250#
V10      011724          3339     4252#
V115     011732          3376     4255#
V144     011730          3419     4254#
V2       011722          3411     4251#
V240     011734          3356     3364     3426     4256#
V5       011736          3399     4257#
V50      011726          3307     3347     4253#
V50D     011740          3448     4258#
WFADJ    011650          3153     4235#
WFTEST   001416          3069#    3118*    3120*    4240
WIDE     001336          3045#    3783*    3857*    3883     3896
WIDMSG   012607          3886     4314#
WRAP     003436          3298#    3539     3550
W1       017724          4387#*
W2       017726          4387#*
```

```
W3       017730             4387#*
$AERR    017454             3194     4387#*
$APTHD   001000             2995#
$ASTAT=  ****** U           4386
$ATYC    016354             4386#
$ATY1    016330             4386#
$ATY3    016336             4385     4386#
$ATY4    016346             4382     4386#
$AUTOB   001134             2996#
$BASE    001250             2996#    3188     3568     3569
$BDADR   001122             2996#
$BDDAT   001126             2996#    3188*    3193     3198*    3291     3292     4136*    4158*    4159     4369     4370     4371
$BELL    001164             2996#    4382
$CDW1    001254             2996#
$CHARC   016324             4385#*
$CKSWR=  ****** U           4390
$CMTAG   001100             2996#    3122
$CM3   = 000000             2996#
$CNT     020424             4387#*
$CNTLG   015107             4377#
$CNTLU   015102             4377#
$CPUOP   001222             2996#
$CRLF    001171             2996#    3514     3538     4377     4382     4383     4385
$DATR    020076             4387#*
$DEVCT   001204             2996#
$DEVM    001252             2996#
$DOAGN   012110             4285#
$ENDAD   012100             2993     4285#    4382
$ENDCT   012054             3122     4285#
$ENDMG   012117             4285#
$ENULL   012114             4285#
$ENV     001214             2996#    4382     4385     4386
$ENVM    001215             2996#    3122     4385     4386
$EOP     012020             3525     3543     3555     4285#
$EOPCT   012046             3122*    4285#
$ERFLG   001103             2996#    3082*    4381*    4382*
$ERMAX   001115             2996#    3122*    4381*
$ERROR   015516             3122     4382#
$ERRPC   001116             2996#    4369     4370     4371     4382*    4383
$ERRTB   001256             2996#    4383
$ERRTY   015712             4382     4383#
$ERTTL   001112             2996#    4382*
$ESCAP   001162             2996#    3081*    3084*    3122*    4381*    4382
$ETABL   001214             2996#
$ETEND   001256             2995     2996#
$FATAL   001176             2996#    4386*
$FFLG    016574             4386#*
$FILLC   001156             2996#    4385
$FILLS   001155             2996#    4385
$GDADR   001120             2996#
$GDDAT   001124             2996#    3193     3221*    3224*    3225     3230*    3235*    3240*    3244*    3251*    3262*    3274*    3290
                            3292     3316*    4135*    4156*    4160     4369     4370
$GET42   012070             4285#
$GTSWR=  ****** U           4390
$HD    = 000000             2935
$HIBTS   001000             2995#
```

```
$HIOCT  015236            4379#*
$ICNT   001104            2996#   4381*
$ILLUP  021522            4392#
$INLP   020272            3250    3254    3257    3263    3265    3272    3283    3291    3324    3498    3500    3502    3697
                          3698    3700    4122    4129    4141    4143    4387#
$INTAG  001135            2996#
$ITEMB  001114            2996#   4382*   4383
$LF     001172            2996#   4377    4382    4385
$LFLG   016573            4386#*
$LOAD   017456            4387#
$LPADR  001106            2996#   3122*   3219*   3302*   4381*
$LPAI   016576            4387#
$LPERR  001110            2996#   3122*   3220*   3301*   4381*   4382
$LPW    020100            4387#
$MADR1  001226            2996#
$MADR2  001232            2996#
$MADR3  001236            2996#
$MADR4  001242            2996#
$MAIL   001174            2995    2996#   3122    4381    4382    4385
$MAMS1  001224            2996#
$MAMS2  001230            2996#
$MAMS3  001234            2996#
$MAMS4  001240            2996#
$MBADR  001002            2995#
$MFLG   016572            4386#*
$MNEW   015125            4377#
$MSGAD  001210            2996#   4386*
$MSGLG  001212            2996#   4386*
$MSGTY  001174            2996#   4386*
$MSWR   015114            4377#
$MTYP1  001225            2996#
$MTYP2  001231            2996#
$MTYP3  001235            2996#
$MTYP4  001241            2996#
$MXCNT  015514            4381#
$NULL   001154            2996#   4385
$NWTST= 000001            3219#   3228#   3234#   3239#   3243#   3248#   3259#   3269#   3299#   3310#   3326#   3334#   3342#
                          3351#   3359#   3367#   3414#   3421#   3428#   3455#   3464#   3472#
$OCNT   021276            4388#*
$OFS  = 020334            4387#
$OMODE  021300            4388#*
$OUTLP  020174            3193    3250    3254    3261    3271    3276    3290    3314    3318    3325    3369    3381    3485
                          3496    3498    3638    3688    3691    3697    3739    3795    4123    4131    4387#
$OVER   015500            4381#
$PASS   001202            2996#   3122*   3473    3530*   3548*   4285*   4381
$PASTM  001006            2995#
$POWER  021530            4392#
$PUTS   020744            3735    3793    4387#
$PWRAD  021516            4392#
$PWRDN  021356            3122    4392#
$PWRMG  021512            4392#
$PWRUP  021430            4392#
$QUES   001170            2996#   4377    4382    4385
$RDCHR  014644            4377#   4390
$RDDEC= ****** U          4390
$RDLIN  014764            4377#   4390
```

```
$RDOCT  015136           4379#   4390
$RDSZ = 000010           4377#
$RESET  020426           3404    4170    4387#
$RTNAD  012112           4285#
$R2A  = ****** U         4390
$SAD    020422           4387#*
$SAVRE= ****** U         4390
$SAVR6  021526           4392#*
$SCOPE  015240           3122    4381#
$SETUP= 000037           2997#   3122    4285    4377    4381    4382
$STUP = 177777           2997#
$SVLAD  015444           4381#
$SVPC = 000220           2993#
$SWR  = 167400           2928#   2935    2937    2996    3122    3219    3228    3234    3239    3243    3248    3259    3269
                         3299    3310    3326    3334    3342    3351    3359    3367    3414    3421    3428    3455    3464
                         3472    4285    4381    4382    4392
$SWREG  001216           2996#   3122
$SWRMK= 000000           2937    4381
$TBF4   007150           3808#
$TEMP1  001432           3075#   3746*   3747*   3749    3796*   3797*   3808
$TEMP2  001434           3076#   3744*   3745*   3753
$TESTN  001200           2996#   4381*
$TIMES  001160           2996#   3122*   3269*   3299*   3310*   3326*   3334*   3342*   3351*   3359*   3367*   3414*   3421*
                         3428*   3455*   3464*   3472*   4285*   4381*
$TKB    001146           2996#   3089    3214    4377    4387
$TKS    001144           2996#   3129*   3160*   3208    3405*   4171*   4281*   4377    4387
$TLKR   017732           4387#
$TLKW   017616           4387#
$TN   = 000027           2929#   2935    3219#   3228#   3234#   3239#   3243#   3248#   3259#   3269#   3299#   3300    3310#
                         3312    3326#   3334#   3342#   3351#   3359#   3367#   3414#   3421#   3428#   3452    3455#   3464#
                         3472#
$TOUT   020366           4387#
$TPB    001152           2996#   4189*   4199*   4385*
$TPFLG  001157           2996#   4385
$TPS    001150           2996#   4187    4197    4385
$TRAP   021302           3122    4390#
$TRAP2  021324           4390#
$TRP  = 000010           4390#
$TRPAD  021336           4390#
$TSTM   001004           2995#
$TSTNM  001102           2996#   3300*   4285*   4381*   4382
$TTYIN  015072           4377#
$TYPBN= ****** U         4390
$TYPDS= ****** U         4390
$TYPE   016046           4385#   4386    4390
$TYPEC  016260           4385#
$TYPEX  016326           4385#
$TYPOC  021100           4388#   4390
$TYPON  021114           4388#   4390
$TYPOS  021054           4388#   4390
$T6MP   006634           3749#
$UNIT   001206           2996#
$UNITM  001010           2995#
$USWR   001220           2996#
$UTK    020550           2991    4387#
$VECT1  001244           2996#   3571    3573
```

```
                                                                                                  F  8
LPA-AD11K TEST MD-11-CRLPKB      MACY11 30G(1063)  08-AUG-79  10:19  PAGE 49-11
CRLPKB.P11     08-AUG-79 10:18              CROSS REFERENCE TABLE -- USER SYMBOLS                                                      SEQ 0096

$VECT2  001246          2996#
$XTSTR  015250          4381#
$$GET4= 000000          4285#
$OFILL  021277          4388#*
$40CAT= ****** U        4381     4382
.     = 062300          2988#    2993#    2994#    2995#    2996#    3079#    3122     4285     4377#    4381     4382     4383#    4385
                        4386#    4387#    4392     4394#    4396#    4398#    4400#
.DVLS   001464          3079#    3124*    3189*    3190*    4387*
.$ASTA= ****** U        4386
.$X   = 001000          2995#
```

```
                                                              G  8
LPA-AD11K TEST MD-11-CRLPKB      MACY11 30G(1063)  08-AUG-79  10:19  PAGE 50
CRLPKB.P11     08-AUG-79 10:18          CROSS REFERENCE TABLE -- MACRO NAMES                                SEQ 0097

ADDM     2971#   3700
BICM     2959#   3254
CLRM     2954#   3325    3485
CMPM     2977#
COMMEN   2936#
DUMWRN   3001#
ENDCOM   2936#
ERROR    2936#   3083    3223    3232    3237    3242    3246    3256    3267    3281    3308    3323    3332    3340    3348
         3357    3365    3377    3400    3412    3420    3427    4138    4387
ESCAPE   2936#   3081
GETPRI   2936#
GETSWR   2936#
INCRM    2947#   3250    3498    3697
MOVEI     170#   3250    3254    3257    3263    3265    3272    3283    3291    3324    3498    3500    3502    3697    3698
         3700    4122    4129    4141    4143    4387
MOVEM     157#   3193    3250    3254    3261    3271    3276    3290    3314    3318    3325    3369    3381    3485    3496
         3498    3638    3688    3691    3697    3739    3795    4123    4131
MOVEMR   2983#   3496    3691    3795    4123
MOVERO   2942#   3257    3265    3283    3324    3502
MULT     2936#
NEWTST   2936#   3219    3228    3234    3239    3243    3248    3259    3269    3299    3310    3326    3334    3342    3351
         3359    3367    3414    3421    3428    3455    3464    3472
POP      2936#   4379    4386    4392
PUSH     2936#   4379    4386    4392
REPORT   2936#
SCOPE    2936#   3228    3234    3239    3243    3248    3259    3269    3285    3310    3326    3334    3342    3351    3359
         3367    3414    3421    3428    3455    3464    3472
SETPRI   2936#
SETTRA   4390#
SETUP    2936#   3122
SKIP     2936#   3165    3168    3171    3174    3177    3180    3199    3209    3211    3213    3226    3293    3312    3452
         3491    4022    4237    4239
SLASH    2936#
SPACE    2936#
STARS    2936#   2993    2995    2996    3219    3228    3234    3239    3243    3248    3259    3269    3299    3310    3326
         3334    3342    3351    3359    3367    3414    3421    3428    3455    3464    3472    4285    4377    4379    4381
         4382    4383    4385    4386    4388    4390    4392
SWRSU    2936#   3122#
TOUT     3079#   4387
TRMTRP   4390#
TSTBM    2966#   3263    3272    3500    3698    4141
TYPBIN   2936#
TYPDEC   2936#
TYPNAM   2936#
TYPNUM   2936#
TYPOCS   2936#   3108    3201    3493    3508    3629    3633    3641    3710    3715    3865    3952    3955    4055
TYPOCT   2936#   4383    4387
TYPTXT   2936#   4387
$CAL.     747#   4387
$DMAST   1792#
$DMDT    2821#
$MMAST    761#
$$CMRE   2996#
$$CMTM   2996#
$$ESCA   2936#
$$NEWT   2936#   3219    3228    3234    3239    3243    3248    3259    3269    3299    3310    3326    3334    3342    3351
```

```
                                                                                              H  8
LPA-AD11K TEST MD-11-CRLPKB       MACY11 30G(1063)  08-AUG-79  10:19  PAGE 50-1
CRLPKB.P11     08-AUG-79 10:18            CROSS REFERENCE TABLE -- MACRO NAMES                                                    SEQ 0098

          3359     3367     3414     3421     3428     3455     3464     3472
$$SET     4390#
$$SETM    3122#
$$SKIP    2936#    3312     3452
.EQUAT    2930#    2936
.HEADE    2930#    2935
.KMADR      55#    3079
.KSIS      184#    3124
.LOADL     458#    4387
.LPAIN     209#    4387
.PUTCS     417#    4387
.RESET     328#    4387
.SETUP    2932#    2997
.SWRHI    2932#    2937
.SWRLO    2937#
.UTK       698#    4387
.$ACT1    2933#    2993
.$APTB    2933#    2996#
.$APTH    2933#    2995
.$APTY    2933#    4386
.$CATC    2930#    2988
.$CMTA    2930#    2996
.$EOP     2930#    4285
.$ERRO    2931#    4382
.$ERRT    2932#    4383
.$INLP     651#    4387
.$MMAC     141#
.$OUTL     609#    4387
.$PARM    2931#
.$POWE    2931#    4392
.$RAND    2933#
.$RDOC    2933#    4379
.$READ    2931#    4377
.$SAVE    2931#
.$SCOP    2931#    4381
.$SPAC    2932#
.$SWDO    2932#
.$TLKW     510#    4387
.$TOUT    3079#    4387
.$TRAP    2932#    4390
.$TYPD    2933#
.$TYPE    2932#    4385
.$TYPO    2931#    4388


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


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CRLPKB,CRLPKB/CRF=DRLPA.MAC,CRLPKB
 RUN-TIME: 30 17 1 SECONDS
 RUN-TIME RATIO: 163/49=3.3
 CORE USED:  40K  (79 PAGES)
```