AD11-K

PERFORMANCE TEST

MD-11-DZADL-B

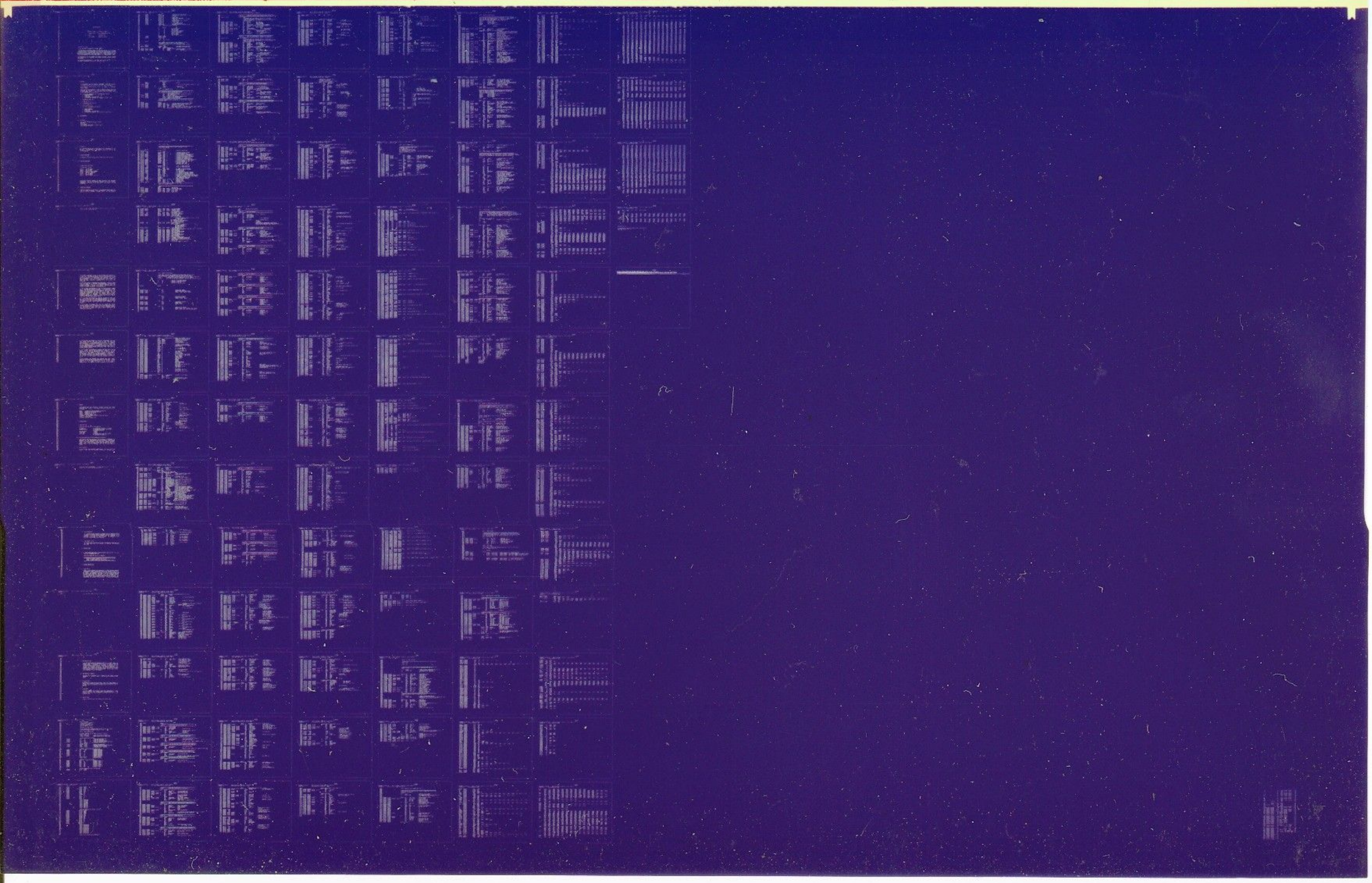
EP-DZADL-B-DL-B

FICHE 1 OF 1

DEC 1976

digital

MADE IN USA



.REM #

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZADL-B
PRODUCT NAME: AD11K PERFORMANCE TEST
DATE: DECEMBER 1976
MAINTAINER: DIAGNOSTIC GROUP

1.0 ABSTRACT

THIS DIAGNOSTIC HAS TWO STARTING ADDRESSES: 200 FOR STANDARD TOLERANCES AND 210 FOR TIGHTER OPTION TEST AREA TOLERANCES.

THIS DIAGNOSTIC TESTS THE AD11K WITH OR WITHOUT A WRAPAROUND MODULE (G5036).

WHEN STARTING THE DIAGNOSTIC, A SET OF TESTS IS LISTED AND THIS STATEMENT IS PRINTED OUT: "TYPE THE LETTER AND CARRIAGE RETURN OF THE DESIRED TEST:". THE FOLLOWING CHART INDICATES WHICH LETTER CORRESPONDS TO WHICH TEST:

W: THE ENTIRE WRAPAROUND TEST (REQUIRES G5036 MODULE)
- A. ANALOG SUBTESTS
- B. NOISE TEST
- C. INTERCHANNEL SETTLING TEST
- D. DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST

C: CALIBRATION TEST ONLY

N: NOISE TEST ONLY

S: INTERCHANNEL SETTLING ONLY

L: LOGIC SUBTESTS ONLY

A: AUTO TEST (REQUIRES G5036 MODULE)

- A. LOGIC SUBTESTS
- B. ANALOG SUBTESTS
- C. NOISE TEST
- D. INTERCHANNEL SETTLING TEST
- E. DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 FAMILY COMPUTER WITH 8K OF MEMORY
TELETYPE
AD11K MODULE
VT55 TERMINAL SUPPORTED FOR GRAPHIC OUTPUT
G5036 WRAPAROUND MODULE

2.2 STORAGE

THIS PROGRAM USES ALL 8K OF MEMORY AND IS NOT "CHAINABLE" ON AN 8K CPU. THE PROGRAM IS "CHAINABLE" ON 12K OR GREATER. THE PROGRAM WILL DESTROY "ABSOLUTE LOADER" ON AN 8K CPU, IF "W" OR "A" IS SELECTED.

3.0 LOADING PROCEDURE

PROCEDURE FOR LOADING NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW15=1 HALT ON ERROR
SW14=1 LOOP ON TEST
SW13=1 INHIBIT ERROR TYPEOUTS
SW12=1 HALT FOR VT55 DISPLAY
SW11=1 INHIBIT ITERATIONS
SW10=1 BELL ON ERROR
SW9 =1 LOOP ON ERROR
SW8 =1 LOOP ON TEST IN SWR <7:0>

200 IS THE STARTING ADDRESS OF THE DIAGNOSTIC FOR STANDARD TOLERANCES. 204 IS THE RESTART ADDRESS. 210 IS THE STARTING ADDRESS OF THE DIAGNOSTIC FOR THE OPTION TEST AREA'S TIGHTER TOLERANCES.

5.0 OPERATING PROCEDURE

START THE DIAGNOSTIC AT 200 OR 210. THE PROGRAM HEADING AND THE LIST OF TESTS AVAILABLE, WILL BE PRINTED OUT FOLLOWED BY A MESSAGE "TYPE THE LETTER AND CARRIAGE RETURN FOR THE DESIRED

```
   161                          TEST:".  THEN  TYPE  THE LETTER YOU WANT, ACCORDING TO THE TABLE
   162                          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 $VECTI (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 $VECTI (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

IF "S" IS TYPED, THE PROGRAM WILL RUN THE INTERCHANNEL SETTLING TEST TAGGED "BEGINS" AND WILL LOOP ON THIS TEST UNTIL THE OPERATOR HALTS IT. AT THE BEGINNING OF THIS TEST, THE OPERATOR MUST RESPOND TO THE STATEMENTS ASKING FOR THE "FROM" CHANNEL AND THE "TO" CHANNEL BY TYPING IN THE CHANNEL VALUE IN OCTAL AND HITTING CARRIAGE RETURN. IF A CERTAIN AD11K IS TO BE TESTED ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO $BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO $VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

IF "A" IS TYPED, THE PROGRAM WILL EXECUTE THE LOGIC TESTS, ANALOG TESTS, NOISE, SETTLE AND DIFFERENTIAL LINEARITY. AT THE BEGINNING OF THE TEST THE PROGRAM WILL TYPE "XX AD11K'S FOUND". WHERE XX IS THE NUMBER OF AD11K'S IN OCTAL IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K. THE PROGRAM SUPPORTS AD11K EXPANSION BEYOND 16. CHANNELS. TO RUN THIS TEST ON A GROUP OF CHANNELS OTHER THAN 0-17, LOAD 20,40, OR 60 INTO LOCATION BASECH (1336) FOR CHANNELS 20-37, 40-57, 60-77.

IF "L" IS TYPED, THE PROGRAM WILL EXECUTE THE LOGIC TESTS, PRINTING "END PASS" WHEN IT HAS COMPLETED AN ENTIRE PASS. AT THE BEGINNING OF THE TEST THE PROGRAM WILL TYPE "XX AD11K'S FOUND". WHERE XX IS THE NUMBER OF AD11K'S IN OCTAL IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K.

6.0 ERRORS

THIS PROGRAM USES THE DIAGNOSTIC "SYSMAC" PACKAGE FOR ERROR REPORTING AND TYPEOUT. THE ERROR INFORMATION CONSISTS OF THE FOLLOWING:

ERRPC: LOCATION AT WHICH AN ERROR WAS DETECTED.
STREG: ADDRESS OF THE STATUS REGISTER.
ADBUFF: ADDRESS OF THE BUFFER
CHANL: CHANNEL VALUE
NOMINAL: EXPECTED CORRECT DATA
TOLERANCE: THE ACCEPTABLE DEVIATION FROM THE NOMINAL
ACTUAL: ACTUAL DATA
EXPECTED: EXPECTED CORRECT DATA

7.0 MISCELLANEOUS

7.1 EXECUTION TIME

EXECUTION TIME FOR EACH OF THE TESTS IS:

CALIBRATION:	8 CONVERSIONS/5 SECONDS @ 110 BAUD
WRAPAROUND TEST:	17 MINUTES FIRST PASS; 35 MINUTES FOR SUCCESSIVE PASSES
SETTLING TEST:	1 MINUTE
NOISE TEST:	1 MINUTE
LOGIC TEST:	1 MINUTE
AUTO TEST:	18 MINUTES FIRST PASS, 36 MINUTES FOR SUCCESSIVE PASSES

7.2 STATUS REGISTER AND VECTOR ADDRESSES AND PRIORITY

WHEN TESTING MORE THAN ONE AD11K, THE DIFFERENCE IN ADDRESSES IS PRESENTLY 40 FOR BUS ADDRESS AND VECTOR ADDRESS. THESE VALUES ARE IN VADR (BUS ADDRESS) (1332) AND VVCT (VECTOR ADDRESS) (1334). THE FIRST AD11K'S STATUS REGISTER ADDRESS MUST BE IN $BASE (1250), ITS VECTOR ADDRESS MUST BE IN THE LOW BYTE OF $VECT1 (1244), AND THE PRIORITY MUST BE IN THE HIGH BYTE OF $VECT1.

7.3 AD11K PRIORITY

IF AD11K IS SET FOR A PRIORITY OTHER THAN 6, THE HIGH BYTE OF $VECT1 (1244) MUST BE ADJUSTED ACCORDINGLY (THE LOW BYTE

```
286                                     CONTAINING THE VECTOR ADDRESS).  IF MORE THAN ONE AD11K IS  BEING
287                                     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 VT55 GRAPHIC OUTPUT

THE SCREEN DISPLAY MAY BE HALTED FOR EXAMINATION BYSETTING BIT 12. AND THEN JUST HIT CONTINUE TO COMPLETE THE PROGRAM'S EXECUTION.

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 14 LOGIC SUBTESTS RUN SEQUENTIALLY WITHOUT FURTHER OPERATOR INTERVENTION AFTER HE/SHE HAS TYPED IN THE NUMBER OF AD11K'S TO BE TESTED. ITS PURPOSE IS TO CHECK THAT EACH OF THE MUX BITS CAN BE LOADED AND PROPERLY READ BACK; THAT INITIALIZE CLEARS THE EXTERNAL START ENABLE BIT, THE DONE BIT, THE INTERRUPT ENABLE BIT, THE OVERFLOW BIT, THE ERROR FLAG, AND THE A/D START BIT. IT ALSO CHECKS THAT THE A/D DONE FLAG SETS AT END OF CONVERSION AND CLEARS WHEN THE CONVERTED VALUE IS READ. IT CHECKS THE INTERRUPT

342 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. R'S NOISE EQUALS 1 STANDARD DEVIATION F 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.

```
                                .TITLE  MAINDEC-11-DZADL-B
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY VERA BREUER
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C0),MAR 21, 1976.
                                ;*
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

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

```
                020000          SW13=   20000
                010000          SW12=   10000
                004000          SW11=   4000
                002000          SW10=   2000
                001000          SW09=   1000
                000400          SW08=   400
                000200          SW07=   200
                000100          SW06=   100
                000040          SW05=   40
                000020          SW04=   20
                000010          SW03=   10
                000004          SW02=   4
                000002          SW01=   2
                000001          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

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

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

```
                    000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                    000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                    000014                  TBITVEC=14              ;;"T" BIT
                    000014                  TRTVEC= 14              ;;TRACE TRAP
                    000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                    000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                    000024                  PWRVEC= 24              ;;POWER FAIL
                    000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                    000034                  TRAPVEC=34              ;;"TRAP" TRAP
                    000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                    000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                    000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                                            .SBTTL  OPERATIONAL SWITCH SETTINGS
                                            ;*
                                            ;*      SWITCH                  USE
                                            ;*      ------          --------------------
                                            ;*        15            HALT ON ERROR
                                            ;*        14            LOOP ON TEST
                                            ;*        13            INHIBIT ERROR TYPEOUTS
                                            ;*        12            HALT FOR VT55 DISPLAY
                                            ;*        11            INHIBIT ITERATIONS
                                            ;*        10            BELL ON ERROR
                                            ;*         9            LOOP ON ERROR
                                            ;*         8            LOOP ON TEST IN SWR<7:0>
                    170400                  ABASE=  170400
                    140340                  AVECT1= 140340
                    000300                  APRIOR= 300

                                            .SBTTL  TRAP CATCHER

                    000000                          .=0
                                            ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                            ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                            ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                    000174                          .=174
000174  000000                             DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                             SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                            .SBTTL  STARTING ADDRESS(ES)
000200  000137  001620                             JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  002256                             JMP     @#BEG2          ;RESTART ADDRESS
000210  000137  001626                             JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;**********************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  011776                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SECP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  002260          $TSTM:  .WORD   1200.   ;;RUN TIM OF LONGEST TEST
001006  000764          $PASTM: .WORD   500.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  003244          $UNITM: .WORD   1700.   ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000033                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                        .SBTTL  COMMON TAGS

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

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

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

```
  626  001206  000000                $UNIT:  .WORD   AUNIT   ;;I O UNIT NUMBER
  627  001210  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
  628  001212  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
  629  001214                        $ETABLE:                ;;APT ENVIRONMENT TABLE
  630  001214     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
  631  001215     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
  632  001216  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
  633  001220  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
  634  001222  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
  635                                :*                      BITS 15-11=CPU TYPE
  636                                :*                              11/04=01,11/05=02,11/20=03,11 40=04,11 45=05
  637                                :*                              11/70=06,PDQ=07,Q=10
  638                                :*                      BIT 10=REAL TIME CLOCK
  639                                :*                      BIT  9=FLOATING POINT PROCESSOR
  640                                :*                      BIT  8=MEMORY MANAGEMENT
  641  001224     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
  642  001225     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
  643                                :*                      MEM.TYPE BYTE   --  (HIGH BYTE)
  644                                :*                              900 NSEC CORE=001
  645                                :*                              300 NSEC BIPOLAR=002
  646                                :*                              500 NSEC MOS=003
  647  001226  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
  648                                :*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
  649  001230     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
  650  001231     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
  651  001232  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
  652  001234     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
  653  001235     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
  654  001236  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
  655  001240     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
  656  001241     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
  657  001242  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
  658  001244  140340                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
  659  001246  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
  660  001250  170400                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
  661  001252  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
  662  001254  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
  663  001256                        $ETEND:
  664                                .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  014146                              EM1                     ;STATUS REG. ERROR
001260  014306                              DH1                     ;ERRPC STREG EXPECTED ACTUAL
001262  014472                              DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
001264  014532                              DF1


                                    ;ITEM   2
001266  014174                              EM2                     ;FAILED TO INTERRUPT
001270  014427                              DH3                     ;ERRPC STREG ACTUAL
001272  014522                              DT3                     ;$ERRPC, STREG,  $BDDAT
001274  014532                              DF1

                                    ;ITEM   3
001276  014224                              EM3                     ;UNEXPECTED INTERRUPT
001300  014427                              DH3                     ;ERRPC STREG
001302  014522                              DT3                     ;$ERRPC, STREG
001304  014532                              DF1

                                    ;ITEM   4
001306  014255                              EM4                     ;ERROR ON A/D CHANNEL
001310  014344                              DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
001312  014504                              DT2                     ;$ERRPC,$STREG,$CHANL,$GDDAT,$SPREAD,$BDDAT
001314  014532                              DF1
```

```
                                        .SBTTL      MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
001316  170400                  STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
001320  170401                  ADST1:  ABASE+1                 ;UPPER BYTE OF STATUS REG.
001322  170402                  ADBUFF: ABASE+2                 ;ADDRESS OF A/D BUFFER
001324  140340                  VECTOR: AVECT1                  ;VECTOR ADDRESS
001326  000300                  BASEBR: APRIOR                  ;INTERRUPT PRIORITY LEVEL
001330  140342                  VECTR1: AVECT1+2
001332  000040                  VADR:   40                      ;INCREMENT FOR BUS ADDRESS
001334  000040                  VVCT:   40                      ;INCREMENT FOR VECTOR ADDRESS
001336  000000                  BASECH: 0                       ;BASE CHANNEL
001340  000060                  KBVECT: 60
001342  000000                  WIDE:   0                       ;NO. OF WIDE STATES
001344  000000                  NARROW: 0                       ;NO. OF NARROW STATES
001346  000000                  FIRST:  0
001350  000000                  SKIPST: 0                       ;NO. OF SKIPPED STATES
001352  000000                  TEMP:   0                       ;WORK AREA
001354  000000                  CH1:    0                       ;FIRST CHANNEL
001356  000000                  CH2:    0                       ;SECOND CHANNEL
001360  000000                  NBEXT:  0                       ;NO. OF AD11K'S TO BE TESTED
001362  000000                  NMBEXT: 0                       ;NO. OF AD11K'S TO BE TESTED
001364  000000                  DUMMY:  0                       ;DUMMY CHANNEL
001366  000000                  CHANL:  0                       ;CHANNEL VALUE
001370  000000                  TADDR:  0                       ;TEST ADDRESS
001372  000000                  RNA:    0                       ;RANDOM
001374  000000                  RNB:    0                       ;NUMBER
001376  000000                  RNC:    0                       ;VALUES
001400  000000                  RMS:    0                       ;RMS NOISE VALUE
001402  000000                  PEAK:   0                       ;PEAK NOISE VALUE
001404  000000                  FLAG:   0                       ;VT55 FLAG
001406  000000                  SPREAD: 0                       ;DEVIATION FROM THE NOMINAL
001410  000000                  DAC:    0                       ;SAR VALUE
001412  000000                  DELAY:  0                       ;TIME DELAY COUNTER
001414  000000                  EDGE:   0                       ;EDGE VALUE
001416  000000                  BITPNT: 0
001420  000000                  MIN:    0                       ;MIN VALUE
001422  000000                  WFTEST: 0                       ;OPTION TEST AREA FLAG
001424  000000                  MAX:    0                       ;MAX VALUE
001426  000000                  PERCNT: 0                       ;PERCENT FOR SAR ROUTINE
001430  000000                  OUT:    0

001432                          UNEXP:
001432  012737  001446  001162          MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
001440  005237  001103                  INC     $ERFLG
001444  104003                          ERROR   3
001446  005037  001162          1$:     CLR     $ESCAPE         ;RETURN ESCAPE TO NORMAL
001452  000002                          RTI                     ;UNEXPECTED INTERRUPT
```

```
                                        .SBTTL      CONTROL A AND C DECODERS
001454  010046                  ISERV:  MOV     R0,-(SP)            ;SAVE R0
001456  017700  177464                  MOV     @#TKB,R0            ;GET CHARACTER
001462  042700  177600                  BIC     #177600,R0
001466  120027  000003                  CMPB    R0,#3               ;IS IT ↑C?
001472  001010                          BNE     1$
001474  104400  012146                  TYPE    ,CMSG               ;ECHO CHARACTER
001500  012706  001100                  MOV     #STACK,SP
001504  004737  011266                  JSR     PC,RST              ;RESET & SET INTRPT. EN.
001510  000137  002256                  JMP     BEG2
001514  120027  000001          1$:     CMPB    R0,#1               ;IS IT ↑A?
001520  001010                          BNE     2$
001522  104400  012141                  TYPE    ,AMSG               ;ECHO CHARACTER
001526  012706  001100                  MOV     #STACK,SP
001532  004737  011266                  JSR     PC,RST              ;RESET & SET INTRPT. EN.
001536  000177  177626                  JMP     @TADDR              ;RETURN TO TEST
001542  120027  000007          2$:     CMPB    R0,#7               ;IS IT ↑G?
001546  001021                          BNE     NONE
001550  023727  001140  177570          CMP     SWR,#177570         ;HARDWARE SWREG?
001556  001415                          BEQ     NONE
001560  104400  012153                  TYPE    ,GMSG               ;ECHO CHARACTER
001564  017746  177350                  MOV     @SWR,-(SP)      ;;SAVE @SWR FOR TYPEOUT
                                                                ;;TYPE SWREG
001570  104402                  TYPOS                           ;;GO TYPE--OCTAL ASCII
001572     006                  .BYTE   6                       ;;TYPE 6 DIGITS
001573     001                  .BYTE   1                       ;;TYPE LEADING ZEROS
001574  104400  012333                  TYPE    ,SLASH
001600  104406                          RDOCT                   ;READ NEW VALUE
001602  012677  177332                  MOV     (SP)+,@SWR      ;LOAD NEW SWREG VALUE
001606  012600                  POPR0:  MOV     (SP)+,R0
001610  000002                  RETURN: RTI
001612  104400  012137          NONE:   TYPE    ,QUEST              ;TYPE "?"
001616  000773                          BR      POPR0
```

```
                                                .SBTTL          INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
001620  005037  001422            BEGIN:  CLR     WFTEST
001624  000403                            BR      RBEG
001626  012737  000001  001422    BEGIN2: MOV     #1,WFTEST
001634  000005                    RBEG:   RESET
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001636  012706  001100                    MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001642  005026                            CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001644  022706  001140                    CMP     #SWR,R6 ;;DONE?
001650  001374                            BNE     .-6             ;;LOOP BACK IF NO
001652  012706  001100                    MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001656  012737  015130  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001664  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001672  012737  015406  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001700  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001706  012737  016714  000034            MOV     #$TRAP,@#TRAP_VEC ;;TRAP VECTOR FOR TRAP CALLS
001714  012737  000340  000036            MOV     #340,@#TRAP_VEC+2;LEVEL 7
001722  012737  016754  000024            MOV     #$PWRDN,@#PWR_VEC ;;POWER FAILURE VECTOR
001730  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001736  013737  011756  011750            MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001744  005037  001160                    CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001750  005037  001162                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001754  112737  000001  001115            MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001762  012737  001762  001106            MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001770  012737  001770  001110            MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001776  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002002  012737  002036  000004            MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002010  012737  177570  001140            MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002016  012737  177570  001142            MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002024  022777  177777  177106            CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002032  001012                            BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002034  000403                            BR      65$             ;;BRANCH IF NO TIMEOUT
002036  012716  002044            64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002042  000002                            RTI
002044  012737  000176  001140    65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002052  012737  000174  001142            MOV     #DISPREG,DISPLAY
002060  012637  000004            66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002064  005037  001202                    CLR     $PASS           ;;CLEAR PASS COUNT
002070  132737  000200  001215            BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002076  001403                            BEQ     67$             ;;YES,USE NON-APT SWITCH
002100  012737  001216  001140            MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002106                            67$:
```

```
836  002106  005037  001404                    CLR     FLAG            ;CLEAR VT55 FLAG
837  002112  005737  000042                    TST     @#42                    ;IS IT CHAINED?
838  002116  001033                            BNE     REST1
839                                    .SBTTL          DETERMINE IF VT55 TYPE TERMINAL IS PRESENT
840  002120  042777  000100  177016            BIC     #100,@STKS
841  002126  104400  013573                    TYPE    ,CO             ;TYPE ASCIZ STRING
842  002132  004737  002530                    JSR     PC,VTFLG        ;GET A CHARACTER
843  002136  020027  000033                    CMP     R0,#33
844  002142  001017                            BNE     NOVT55          ;NO VT55 PRESENT
845  002144  004737  002530                    JSR     PC,VTFLG        ;GET A CHARACTER
846  002150  020027  000057                    CMP     R0,#57
847  002154  001012                            BNE     NOVT55          ;NO VT55 PRESENT
848  002156  004737  002530                    JSR     PC,VTFLG        ;GET A CHARACTER
849  002162  020027  000103                    CMP     R0,#103
850  002166  001403                            BEQ     VT55            ;VT55 IS PRESENT
851  002170  020027  000105                    CMP     R0,#105
852  002174  001002                            BNE     NOVT55
853  002176  005237  001404            VT55:   INC     FLAG
```

```
                                                .SBTTL      DIALOGUE TO DETERMINE WHICH TEST TO RUN
002202  104400  013736          NOVTST: TYPE    .HEAD1
002206  000005                  REST1:  RESET
002210  004737  005504                  JSR     PC,FIXONE               ;INITIALIZE ADDRESSES
002214  013700  001340                  MOV     KBVECT,R0
002220  012720  001454                  MOV     #ISERV,(R0)+
002224  012710  000340                  MOV     #340,(R0)
002230  012737  062341  001372          MOV     #62341,RNA              ;RANDOM NO, VARIABLES
002236  012737  142315  001374          MOV     #142315,RNB
002244  012737  127623  001376          MOV     #127623,RNC
002252  004737  011552                  JSR     PC,WFADJ                ;STANDARD OR OPTION TEST TOLERANCES?
002256  000005                  BEG2:   RESET                   ;RESTART ADDRESS
002260  012706  001100                  MOV     #STACK,SP       ;RESET STACK IN CASE RESTARTED
002264  005737  000042                  TST     @#42                    ;IS IT CHAINED?
002270  001402                          BEQ     1$                      ;GO TO LOGIC TESTS
002272  000137  005234                  JMP     BEGL
002276  104400  013401          1$:     TYPE    ,MSG71
002302  104405                  TRYAG:  RDLIN
002304  052777  000100  176632          BIS     #100,@$TKS
002312  005037  177776                  CLR     PSW
002316  012600                          MOV     (SP)+,R0                ;READ ANSWER
002320  142710  000040                  BICB    #40,(R0)
002324  121027  000101                  CMPB    (R0),#'A                ;IS IT A?
002330  001002                          BNE     1$              ;;NO, TRY C
002332  000137  005272                  JMP     BEGINA                  ;GO TO AUTO TEST
002336  121027  000103          1$:     CMPB    (R0),#'C                ;IS IT C?
002342  001002                          BNE     2$              ;;NO, TRY L
002344  000137  005050                  JMP     BEGINC                  ;GO TO CALIBRATION TEST
002350  121027  000114          2$:     CMPB    (R0),#'L                ;IS IT L?
002354  001002                          BNE     3$              ;;NO, TRY N
002356  000137  005234                  JMP     BEGL                    ;GO TO LOGIC TESTS
002362  121027  000116          3$:     CMPB    (R0),#'N                ;IS IT N?
002366  001002                          BNE     4$              ;;NO, TRY S
002370  000137  005660                  JMP     BEGINN                  ;GO TO NOISE TEST
002374  121027  000123          4$:     CMPB    (R0),#'S                ;IS IT S?
002400  001002                          BNE     5$              ;;NO, TRY W
002402  000137  005730                  JMP     BEGINS                  ;GO TO SETTLE TEST
002406  121027  000127          5$:     CMPB    (R0),#'W                ;IS IT W?
002412  001002                          BNE     6$              ;;NO.TRY AGAIN
002414  000137  005360                  JMP     BEGINW                  ;GO TO WRAPAROUND TEST
002420  104400  012137          6$:     TYPE    ,QUEST
002424  000726                          BR      TRYAG                   ;WAIT FOR CHARACTER
002426  013737  001250  001126  TESTAD: MOV     $BASE,$BDDAT    ;SETUP TO TEST FOR AD11K'S
002434  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE ERRVEC
002440  012737  002472  000004          MOV     #2$,ERRVEC      ;SET UP FOR TIME OUT ERROR
002446  005037  001360                  CLR     NBEXT           ;CLEAR AD11K COUNTER
002452  005777  176450          1$:     TST     @$BDDAT         ;ADDRESS AD11K
002456  005237  001360                  INC     NBEXT           ;INCREMENT AD11K COUNTER
002462  063737  001332  001126          ADD     VADR,$BDDAT     ;GET NEXT AD11K
002470  000770                          BR      1$              ;;TRY NEXT AD11K
```

```
904  002472  022626                    2$:     CMP     (SP)+,(SP)+     ;POP 2 WORDS OFF STACK
905  002474  013746  001360                    MOV     NBEXT,-(SP)     ;;SAVE NBEXT FOR TYPEOUT
906                                                                     ;;TYPE NUMBER OF AD11K'S
907  002500  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
908  002502     002                            .BYTE   2               ;;TYPE 2 DIGIT(S)
909  002503     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
910  002504  104400  012741                    TYPE    ,MSG50
911  002510  005337  001360                    DEC     NBEXT           ;ADJUST AD11K COUNT
912  002514  013737  001360  001362            MOV     NBEXT,NMBEXT    ;KEEP COUNT OF NUMBER
913  002522  012637  000004                    MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
914  002526  000207                            RTS     PC
915
916  002530  005000                VTFLG:      CLR     R0              ;TEST FOR PRESENCE
917  002532  105777  176406        1$:         TSTB    @$TKS           ;OF VT55
918  002536  100404                            BMI     2$              ;;VT55 RESPONDS WITH <33><57 [<103> OR <105>]
919  002540  005300                            DEC     R0
920  002542  001373                            BNE     1$              ;;
921  002544  005726                            TST     (SP)+           ;POP A WORD OFF STACK
922  002546  000615                            BR      NOVT55          ;;NO VT55 PRESENT
923  002550  017700  176372        2$:         MOV     @$TKB,R0
924  002554  042700  177600                    BIC     #177600,R0              ;TEST VT55 CODE
925  002560  000207                            RTS     PC
```

```
                                BEGINL:
                                ;;*****************************************************************
                                ;*TEST 1         FLOAT A ONE THRU MULTIPLEXER BITS
                                ;;*****************************************************************
002562  012737  002562  001106  TST1:   MOV     #TST1,$LPADR
002570  012737  002562  001110          MOV     #TST1,$LPERR
002576  012737  000400  001124          MOV     #BIT8,$GDDAT            ;LOAD FIRST BIT
002604  004737  003660          2$:     JSR     PC,TESTIT
002610  104001                          ERROR   1                       ;FAILED TO LOAD + READ BIT
002612  006137  001124          1$:     ROL     $GDDAT                  ;GET NEXT BIT
002616  023727  001124  040000          CMP     $GDDAT,#BIT14           ;FINISHED?
002624  001367                          BNE     2$              ;;NO,GO TO NEXT TEST

                                ;;*****************************************************************
                                ;*TEST 2         LOAD AND READ BACK INTERRUPT ENABLE BIT6
                                ;;*****************************************************************
002626  000004                  TST2:   SCOPE
002630  012777  001432  176466          MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
002636  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD EXPECTED DATA
002644  004737  003660                  JSR     PC,TESTIT
002650  104001                          ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE

                                ;;*****************************************************************
                                ;*TEST 3         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
                                ;;*****************************************************************
002652  000004                  TST3:   SCOPE
002654  012737  000040  001124          MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
002662  004737  003660                  JSR     PC,TESTIT
002666  104001                          ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB

                                ;;*****************************************************************
                                ;*TEST 4         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
                                ;;*****************************************************************
002670  000004                  TST4:   SCOPE
002672  012737  000020  001124          MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
002700  004737  003660                  JSR     PC,TESTIT
002704  104001                          ERROR   1               ;FAILED TO LOAD + READ EXT. START ENABLE
                                ;;*****************************************************************
                                ;*TEST 5         LOAD AND READ BACK ERROR FLAG BIT15
                                ;;*****************************************************************
002706  000004                  TST5:   SCOPE
002710  012737  100000  001124          MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
002716  004737  003660                  JSR     PC,TESTIT
002722  104001                          ERROR   1               ;FAILED TO LOAD + READ ERROR FLAG
```

```
                                        ;;*****************************************************************
                                        ;*TEST 6         TEST INIT CLEARS BITS 1,4-6,8-13,15
                                        ;;*****************************************************************
002724  000004                  TST6:   SCOPE
002726  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
002734  005037  001124                  CLR     $GDDAT          ;LOAD EXPECTED DATA
002740  012777  137562  176350  2$:     MOV     #137562,@STREG  ;SET STATUS REGISTER
002746  012737  000010  001352          MOV     #10,TEMP
002754  005337  001352          1$:     DEC     TEMP            ;DELAY FOR ONE-SHOT
002760  001375                          BNE     1$              ;IN ANALOG PACKAGE
002762  000005                          RESET                   ;INITIALIZE
002764  052777  000100  176152          BIS     #100,@$TKS      ;SET INTRPT. ENABLE
002772  017737  176320  001126          MOV     @STREG,$BDDAT   ;READ STATUS REGISTER
003000  001401                          BEQ     TST7            ;;NEXT TEST
003002  104001                          ERROR   1               ;RESET FAILED TO CLEAR AD ST. REG. BITS

                                        ;;*****************************************************************
                                        ;*TEST 7         TEST  DONE FLAG  SETS AND BIT0 CLEARS ON END OF CONV.
                                        ;;*****************************************************************
003004  000004                  TST7:   SCOPE
003006  012700  001000                  MOV     #BIT9,R0        ;STALL TIME COUNTER
003012  005277  176300                  INC     @STREG          ;START CONVERSION
003016  012737  000200  001124          MOV     #BIT7,$GDDAT    ;LOAD EXPECTED
003024  005300                  1$:     DEC     R0              ;STALL
003026  001376                          BNE     1$              ;TIME
003030  042777  100000  176260          BIC     #BIT15,@STREG   ;MASK OUT ERROR BIT
003036  004737  003666                  JSR     PC,TEST
003042  104001                          ERROR   1               ;A/D DONE FLAG FAILED TO SET;BIT0 FAILED TO CLEAR
003044  017700  176252                  MOV     @ADBUFF,R0      ;CLEAR DONE FLAG FOR ITERATIONS

                                        ;;*****************************************************************
                                        ;*TEST 10        TEST INIT CLEARS DONE FLAG
                                        ;;*****************************************************************
003050  000004                  TST10:  SCOPE
003052  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
003060  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
003064  005277  176226                  INC     @STREG          ;START CONVERSION
003070  105777  176222          2$:     TSTB    @STREG
003074  100375                          BPL     2$
003076  000005                          RESET
003100  004737  003666                  JSR     PC,TEST
003104  104001                          ERROR   1               ;DONE FLAG FAILED TO CLEAR
003106  052777  000100  176030          BIS     #100,@$TKS              ;SET INTRPT. EN. BIT

                                        ;;*****************************************************************
                                        ;*TEST 11        TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
                                        ;;*****************************************************************
003114  000004                  TST11:  SCOPE
003116  012777  000001  176172          MOV     #BIT0,@STREG    ;SET A/D START CONVERSION BIT
003124  105777  176166          1$:     TSTB    @STREG          ;WAIT FOR FLAG
003130  100375                          BPL     1$
003132  017700  176164                  MOV     @ADBUFF,R0      ;READ CONVERTED VALUE
003136  004737  003666                  JSR     PC,TEST
003142  104001                          ERROR   1               ;DONE FLAG FAILED TO CLEAR
```

```
                                        ;;*****************************************************************
                                        ;*TEST 12       GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
                                        ;;*****************************************************************
003144  000004                          *ST12:  SCOPE
                                                ;* "ENTERING TEST 12" TYPED OUT TO TELL YOU THE NEXT
                                                ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
                                                ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
                                                ;*EXECUTING TEST "12".
003146  012700  000012                          MOV     #12,R0          ;GET TEST NO.
003152  004737  011164                          JSR     PC,DUMW         ;PRINT MESSAGE
003156  005037  177776                          CLR     PSW             ;RESET PRIORITY
003162  012777  003230  176134                  MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
003170  012777  000101  176120                  MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
003176  105777  176114          2$:     TSTB    @STREG          ;WAIT FOR DONE
003202  100375                          BPL     2$              ;FLAG TO SET
003204  017737  176106  001126                  MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
003212  012737  000300  001124                  MOV     #BIT7!BIT6,$GDDAT               ;GOOD DATA
003220  104002                          ERROR   2               ;FAILED TO INTERRUPT ON DONE
003222  004737  011236                          JSR     PC,DUMC         ;TYPE COMPLETED
003226  000403                          BR      TST13           ;;BRANCH TO NEXT TEST
003230  022626                  1$:     CMP     (SP)+,(SP)+             ;RESET STACK POINTER
003232  004737  011236                          JSR     PC,DUMC         ;TYPE COMPLETED

                                        ;;*****************************************************************
                                        ;*TEST 13       TEST FOR CORRECTNESS OF PRIORITY OF INTERRUPT
                                        ;;*****************************************************************
003236  000004                          *ST13:  SCOPE
                                                ;* "ENTERING TEST 13" TYPED OUT TO TELL YOU THE NEXT
                                                ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
                                                ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
                                                ;*EXECUTING TEST "13".
003240  012700  000013                          MOV     #13,R0          ;GET TEST NO.
003244  004737  011164                          JSR     PC,DUMW         ;PRINT MESSAGE
003250  013737  001326  177776                  MOV     BASEBR,PSW      ;GET ONE LEVEL LESS
003256  162737  000040  177776                  SUB     #40,PSW                 ;PRIORITY
003264  012777  003320  176032                  MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
003272  012777  000101  176016                  MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
003300  105777  176012          2$:     TSTB    @STREG          ;WAIT FOR
003304  100375                          BPL     2$              ;DONE FLAG
003306  017737  176004  001126                  MOV     @STREG,$BDDAT           ;READ ST. REG.
003314  104002                          ERROR   2               ;INTERRUPT NOT SERVICED AT LEVEL 5
003316  000401                          BR      3$
003320  022626                  1$:     CMP     (SP)+,(SP)+             ;RESET STACK POINTER
003322  012777  003402  175774  3$:     MOV     #4$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
003330  013737  001326  177776                  MOV     BASEBR,PSW              ;SET PRIORITY TO 6
003336  012777  000101  175752                  MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE + START CONVERSION
003344  105777  175746          5$:     TSTB    @STREG          ;WAIT FOR
003350  100375                          BPL     5$              ;DONE FLAG
003352  012777  001432  175744  6$:     MOV     #UNEXP,@VECTOR          ;UNEXPECTED INTERRUPT VECTOR
003360  005077  175732                  CLR     @STREG          ;CLEAR BIT 6
003364  005037  177776                  CLR     PSW             ;RESET PRIORITY
003370  017700  175726                  MOV     @ADBUFF,R0              ;CLEAR DONE FLAG
003374  004737  011236                  JSR     PC,DUMC         ;TYPE COMPLETED
003400  000403                          BR      TST14           ;;BRANCH TO NEXT TEST
```

```
 1078  003402  022626                  4$:     CMP     (SP)+,(SP)+             ;RESET STACK POINTER
 1079  003404  104003                          ERROR   3               ;INTERRUPTED AT LEVEL 6 - WRONG
 1080  003406  000761                          BR      6$
 1081
 1082                                  ;;*******************************************************************
 1083                                  ;*TEST 14       TEST DONE FLAG CLEARS AFTER INTERRUPT
 1084                                  ;;*******************************************************************
 1085  003410  000004                  TST14:  SCOPE
 1086  003412  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
 1087  003420  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD GOOD DATA
 1088  003426  012777  003462  175670          MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
 1089  003434  012777  000101  175654          MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
 1090  003442  105777  175650          2$:     TSTB    @STREG          ;WAIT FOR
 1091  003446  100375                          BPL     2$              ;DONE FLAG
 1092  003450  017737  175642  001126          MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
 1093  003456  104002                          ERROR   2               ;FAILED TO GENERATE INTERRUPT
 1094  003460  000412                          BR      TST15           ;;GO TO NEXT TEST
 1095  003462  022626                  1$:     CMP     (SP)+,(SP)+             ;RESET STACK POINTER
 1096  003464  004737  003666                  JSR     PC,TEST
 1097  003470  104001                          ERROR   1               ;DONE FLAG NOT CLEARED AFTER AN INTERRUPT
 1098  003472  017737  175624  001352          MOV     @ADBUFF,TEMP            ;DUMMY READ BUFFER
 1099  003500  012777  001432  175616          MOV     #UNEXP,@VECTOR          ;LOAD INTRPT. ADDRESS
 1100
 1101                                  ;;*******************************************************************
 1102                                  ;*TEST 15       TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
 1103                                  ;;*******************************************************************
 1104  003506  000004                  TST15:  SCOPE
 1105  003510  012777  000001  175600          MOV     #BIT0,@STREG    ;START CONVERSION
 1106  003516  105777  175574          1$:     TSTB    @STREG          ;WAIT FOR
 1107  003522  100375                          BPL     1$
 1108  003524  012737  100200  001124  2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
 1109  003532  012777  000001  175556          MOV     #BIT0,@STREG    ;START 2ND CONVERSION
 1110  003540  012700  001000                  MOV     #BIT9,R0        ;WAIT FOR 2ND
 1111  003544  005300                  3$:     DEC     R0              ;CONVERSION TO END
 1112  003546  001376                          BNE     3$
 1113  003550  004737  003666          4$:     JSR     PC,TEST
 1114  003554  104001                          ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
 1115                                                                  ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
 1116  003556  017700  175540                  MOV     @ADBUFF,R0      ;CLEAR DONE FLAG
```

```
                                        ;;**********************************************************************
                                        ;*TEST 16       TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
                                        ;;**********************************************************************
003562  000004                  TST16:  SCOPE
003564  012737  100000  001124          MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
003572  012700  000100                  MOV     #100,R0         ;STALL TIME COUNTER
003576  012777  000001  175512          MOV     #BIT0,@STREG    ;START CONVERSION
003604  112777  000001  175504          MOVB    #BIT0,@STREG            ;START NEXT CONVERSION
003612  005300                  DEC5:   DEC     R0              ;STALL TIME
003614  001376                          BNE     DEC5
003616  017737  175474  001126          MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
003624  042737  077777  001126          BIC     #77777,$BDDAT           ;MASK OUT BIT 15
003632  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003640  001401                          BEQ     1$              ;;BRANCH OVER ERROR
003642  104001                          ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
                                                                ;CONVERT BEGINS BEFORE FIRST DONE
003644  017700  175452          1$:     MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
003650  005077  175442                  CLR     @STREG                  ;CLEAR STATUS REGISTER
003654  000004                          SCOPE
003656  000207                          RTS     PC              ;RETURN TO TEST SECTION


                                        ;;SUBROUTINE FOR LOGIC TESTS;;
003660  013777  001124  175430  TESTIT: MOV     $GDDAT,@STREG           ;LOAD EXPECTED VALUE
003666  017737  175424  001126  TEST:   MOV     @STREG,$BDDAT           ;READ ST. REG.
003674  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003702  001002                          BNE     RETERR          ;;ERROR RETURN
003704  062716  000002                  ADD     #2,(SP          ;BUMP RETURN ADDRESS TO GET AROUND ERROR
003710  000207                  RETERR: RTS     PC
```

```
                                        .SBTTL      WRAPAROUND TEST SECTION
003712                                  WRAP:
                                        ;;*****************************************************************
                                        ;*TEST 17       TEST CH14 GROUND
                                        ;;*****************************************************************
                                        TST17:  NOP
003712  000240                                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003714  012737  000010  001160                  MOV     #$TN-1,$TSTNM
003722  012737  000017  001102                  MOV     #TST17,$LPERR
003730  012737  003712  001110                  MOV     #TST17,$LPADR
003736  012737  003712  001106
003744  004537  011004                          JSR     R5,CONVRT               ;DO 8 CONVERSIONS
003750  000014                                  14
003752  004537  011116                          JSR     R5,COMPAR               ;COMPARE RESULTS
003756  004000                                  4000                    ;NOMINAL
003760  001630                                  V50                     ;TOLERANCE
003762  104004                                  ERROR   4               ;ERROR-CH14 NOT GROUND-AD11K MUST BE IN
                                                                        ;SINGLE-ENDED CONFIGURATION,G5036 WRAPAROUND
                                                                        ;MODULE MUST BE PRESENT,CHECK CONNECTION A-VV,VV-A

                                        ;;*****************************************************************
                                        ;*TEST 20       TEST CONVERSION FROM EXT. START
                                        ;;*****************************************************************
003764  000004                          TST20:  SCOPE
003766  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003774  005737  001336                          TST     BASECH                  ;TESTING AN AM?
004000  001024                                  BNE     TST21           ;;YES, GOTO NEXT TEST
004002  012777  000020  175306                  MOV     #BIT4,@STREG            ;ENABLE EXTERNAL STAPT
004010  012700  001000                          MOV     #BIT9,R0                ;TIME DELAY COUNTER
004014  012737  000220  001124                  MOV     #BIT7!BIT4,$GDDAT       ;LOAD EXPECTED
004022  012777  000200  175272                  MOV     #200,@ADBUFF            ;SIMULATE EXT. START WITH G5036
                                                                                ;WRAPAROUND MODULE PRESENT
004030  005300                          1$:     DEC     R0
004032  001376                                  BNE     1$
004034  004737  003666                          JSR     PC,TEST
004040  104001                                  ERROR   1                       ;FAILED TO DO CONVERSION FROM EXT. START
004042  017700  175254                          MOV     @ADBUFF,R0
004046  005077  175244                          CLR     @STREG                  ;CLEAR ST. REG.

                                        ;;*****************************************************************
                                        ;*TEST 21       TEST CH0 GROUND
                                        ;;*****************************************************************
004052  000004                          TST21:  SCOPE
004054  012737  000010  001160                  MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004062  004537  011004                          JSR     R5,CONVRT               ;CONVERT 8 TIMES
004066  000000                                  0
004070  004537  011116                          JSR     R5,COMPAR               ;COMPARE RESULTS
004074  004000                                  4000                    ;NOMINAL
004076  011622                                  V1                      ;TOLERANCE
004100  104004                                  ERROR   4               ;ERROR ON A/D CHANNEL
```

```
                                        ;;*****************************************************************
                                        ;*TEST 22        TEST CH1 GROUND
                                        ;;*****************************************************************
004102  000004                  TST22:  SCOPE
004104  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004112  004537  011004                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004116  000001                          1                       ;CHANNEL 1
004120  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004124  004000                          4000                    ;NOMINAL
004126  011626                          V10                     ;TOLERANCE
004130  104004                          ERROR   4               ;ERROR ON A/D CHANNEL

                                        ;;*****************************************************************
                                        ;*TEST 23        TEST CH2 +1 VOLT
                                        ;;*****************************************************************
004132  000004                  TST23:  SCOPE
004134  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004142  004537  011004                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004146  000002                          2                       ;CHANNEL 2
004150  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004154  004632                          4632                    ;NOMINAL
004156  011630                          V50                     ;TOLERANCE
004160  104004                          ERROR   4               ;ERROR ON A/D CHANNEL
                                                                ;AD11K MUST BE SET UP FOR +OR- 5. OR +OR- 5.12

                                        ;;*****************************************************************
                                        ;*TEST 24        TEST CH3 +2.5 VOLTS
                                        ;;*****************************************************************
004162  000004                  TST24:  SCOPE
004164  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004172  004537  011004                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004176  000003                          3                       ;CHANNEL 3
004200  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004204  006000                          6000                    ;NOMINAL
004206  011636                          V240                    ;TOLERANCE
004210  104004                          ERROR   4               ;ERROR ON A/D CHANNEL

                                        ;;*****************************************************************
                                        ;*TEST 25        TEST CH4 -2.5 VOLTS
                                        ;;*****************************************************************
004212  000004                  TST25:  SCOPE
004214  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004222  004537  011004                  JSR     R5,CONVRT               ;CONVERT 8 TIMES
004226  000004                          4                       ;CHANNEL 4
004230  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004234  002000                          2000                    ;NOMINAL
004236  011636                          V240                    ;TOLERANCE
004240  104004                          ERROR   4
```

```
                                        ;;**********************************************************************
                                        ;*TEST 26       TEST VERNIER OFFSET DAC ON CH12
                                        ;;**********************************************************************
004242  000004                  TST26:  SCOPE
004244  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004252  005077  175044                  CLR     @ADBUFF                 ;VERNIER DAC=0
004256  004737  005040                  JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
004262  004537  011004                  JSR     R5,CONVRT               ;CONV. CH12, DIRECT VERNIER DAC
004266  000012                          12
004270  013704  001352                  MOV     TEMP,R4                 ;SAVE VALUE IN R4
004274  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004300  002376                          2376                            ;WITH -1.875 VOLTS
004302  011634                          V11S                            ;TOLERANCE OF 10%
004304  104004                          ERROR   4
004306  005037  001424                  CLR     MAX
004312  012702  000001                  MOV     #1,R2
004316  010277  175000          1$:     MOV     R2,@ADBUFF              ;SET UP NEXT VERNIER DAC VALUE
004322  004737  005040                  JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
004326  004537  011004                  JSR     R5,CONVRT               ;CONVERT IT
004332  000012                          12
004334  005737  001424                  TST     MAX
004340  001010                          BNE     2$
004342  023727  001352  004000          CMP     TEMP,#4000
004350  002404                          BLT     2$
004352  005237  001424                  INC     MAX
004356  010237  001420                  MOV     R2,MIN
004362  020227  000200          2$:     CMP     R2,#200
004366  001003                          BNE     3$
004370  013737  001352  004460          MOV     TEMP,4$
004376  013703  001352          3$:     MOV     TEMP,R3                 ;SAVE VALUE
004402  160437  001352                  SUB     R4,TEMP                 ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
004406  010304                          MOV     R3,R4                   ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
004410  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004414  000006                          6                               ;WITH 15 MILLIVOLTS(1 DAC LSB)
004416  011640                          V5
004420  104004                          ERROR   4
004422  005202                          INC     R2
004424  020227  000400                  CMP     R2,#400                 ;DONE?
004430  001332                          BNE     1$                      ;NO-DO NEXT VERNIER DAC VALUE
004432  000005                          RESET                           ;SHOULD SET VERNIER DAC TO 200
004434  052777  000100  174502          BIS     #100,@$TKS
004442  004737  005040                  JSR     PC,DAWAIT               ;LET DAC SETTLE
004446  004537  011004                  JSR     R5,CONVRT               ;CONVERT IT
004452  000012                          12
004454  004537  011116                  JSR     R5,COMPAR               ;COMPARE RESULTS
004460  000000                  4$:     0
004462  011624                          V2
004464  104004                          ERROR   4
```

```
1293                                    ;;*****************************************************************
1294                                    ;*TEST 27        TEST CH13 +2.5 VOLTS
1295                                    ;;*****************************************************************
1296  004466  000004                    TST27:  SCOPE
1297  004470  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1298  004476  004537  011004                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1299  004502  000013                            13
1300  004504  004537  011116                    JSR     R5,COMPAR               ;COMPARE RESULTS
1301  004510  006000                            6000                            ;NOMINAL
1302  004512  011632                            V144                            ;TOLERANCE
1303  004514  104004                            ERROR   4
1304                                    ;;*****************************************************************
1305                                    ;*TEST 30        TEST CH17 +4V
1306                                    ;;*****************************************************************
1307  004516  000004                    TST30:  SCOPE
1308  004520  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1309  004526  004537  011004                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1310  004532  000017                            17                              ;CHANNEL 17
1311  004534  004537  011116                    JSR     R5,COMPAR               ;COMPARE RESULTS
1312  004540  007146                            7146                            ;NOMINAL
1313  004542  011636                            V240                            ;TOLERANCE
1314  004544  104004                            ERROR   4                       ;ERROR ON A/D CHANNEL
```

```
                                        ;;*****************************************************************
                                        ;*TEST 31        OFFSET ON CH0
                                        ;;*****************************************************************
004546  000004                  TST31:  SCOPE
004550  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
004556  013737  001336  001366          MOV     BASECH,CHANL            ;LOAD CHANNEL
004564  013737  001336  001364          MOV     BASECH,DUMMY            ;LOAD DUMMY
004572  012737  004001  001414          MOV     #4001,EDGE
004600  004537  006516                  JSR     R5,SARSUB
004604  000062                          50.
004606  013737  001410  001352          MOV     DAC,TEMP
004614  004537  006516                  JSR     R5,$ARSUB
004620  000062                          50.
004622  063737  001410  001352          ADD     DAC,TEMP
004630  162737  000062  001352          SUB     #62,TEMP
004636  013700  001420                  MOV     MIN,R0
004642  006300                          ASL     R0
004644  160037  001352                  SUB     R0,TEMP
004650  104400  013605                  TYPE    ,MOFSET         ;TYPE ASCIZ STRING
004654  013702  001352                  MOV     TEMP,R2
004660  004737  011406                  JSR     PC,DECTYP
004664  104400  013620                  TYPE    ,MLSB           ;TYPE ASCIZ STRING
004670  004537  011116                  JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
004674  000000                          0
004676  011642                          V5OD
004700  000401                          BR      OFFERR          ;NO-ERROR
004702  000403                          BR      OFFOK           ;YES-OK
004704  104400  012407          OFFERR: TYPE    ,ERMSG
004710  000402                          BR      TST32           ;;GO TO NEXT TEST
004712  104400  012376          OFFOK:  TYPE    ,OKMSG
```

```
1345                                    ;;*****************************************************************
1346                                    ;*TEST 32       NOISE TEST ON 8 EDGES
1347                                    ;;*****************************************************************
1348  004716  000004                    TST32:  SCOPE
1349  004720  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1350  004726  012737  000116  001352            MOV     #116,TEMP               ;DAC VALUE
1351  004734  004537  010576                    JSR     R5,NOI8                 ;NOISE AT -FULL SCALE
1352  004740  000015                            15
1353  004742  004537  010576                    JSR     R5,NOI8                 ;NOISE AT MID-RANGE
1354  004746  000007                            7
1355  004750  004537  010576                    JSR     R5,NOI8                 ;NOISE AT +FULL SCALE
1356  004754  000016                            16
1357
1358                                    ;;*****************************************************************
1359                                    ;*TEST 33       SETTLE TEST ON 8 EDGES
1360                                    ;;*****************************************************************
1361  004756  000004                    TST33:  SCOPE
1362  004760  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1363  004766  004537  006232                    JSR     R5,SET8                 ;SETTLE-POSITIVE DIRECTION
1364  004772  000015                            15
1365  004774  000016                            16
1366  004776  012737  000116  001352            MOV     #116,TEMP
1367  005004  004537  006232                    JSR     R5,SET8                 ;SETTLE-NEGATIVE DIRECTION
1368  005010  000016                            16
1369  005012  000015                            15
1370                                    ;;*****************************************************************
1371                                    ;*TEST 34       DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
1372                                    ;;*****************************************************************
1373  005014  000004                    TST34:  SCOPE
1374  005016  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1375  005024  005737  001202                    TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
1376  005030  001402                            BEQ     LEND
1377  005032  004737  006724                    JSR     PC,DIFLIN
1378  005036  000207                    LEND:   RTS     PC                      ;RETURN TO TEST SECTION
1379
1380  005040  005000                    DAWAIT: CLR     R0
1381  005042  105300                    1$:     DECB    R0
1382  005044  001375                            BNE     1$
1383  005046  000207                            RTS     PC
```

```
                                        .SBTTL      CALIBRATION TEST
005050  012737  005050  001370  BEGINC: MOV     #BEGINC,TADDR           ;TEST ADDRESS IN TADDR
005056  005077  174234                  CLR     @STREG                  ;CLEAR STATUS REGISTER
005062  104400  013515                  TYPE    ,HEADS                  ;TYPE OUT HEADING
005066  005037  177776                  CLR     PSW
005072  017700  174042          1$:     MOV     @SWR,R0                 ;READ CHANNEL FROM SWITCH REG.
005076  042700  177700                  BIC     #177700,R0              ;ISOLATE MUX BITS
005102  032777  020000  174030          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
005110  001005                          BNE     2$              ;;YES,SKIP TYPEOUT
005112  104400  012221                  TYPE    ,CH
005116  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;TYPE CHANNEL
005120  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005122     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
005123     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
005124  012777  001610  174172  2$:     MOV     #RETURN,@VECTOR         ;ADDRESS AFTER INTRPT.
005132  000300                          SWAB    R0                      ;SWITCH BYTES
005134  052700  000100                  BIS     #BIT6,R0
005140  010077  174152                  MOV     R0,@STREG               ;LOAD THE CHANNEL
005144  012702  000010                  MOV     #10,R2                  ;TYPEOUT COUNTER
005150  005277  174142          3$:     INC     @STREG                  ;START CONVERSION
005154  000001                          WAIT                            ;WAIT FOR INTRPT.
005156  017700  174140                  MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
005162  032777  020000  173750          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
005170  001403                          BEQ     4$                      ;NOT SET, TYPE OUT LIST
005172  010077  173744                  MOV     R0,@DISPLAY                     ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
005176  000735                          BR      1$                      ;REPEAT CONVERSION
005200  104400  012224          4$:     TYPE    ,SPACE
005204  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                                                ;;PRINT OCTAL CONVERTED VALUE
005206  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005210     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
005211     001                          .BYTE   1               ;;TYPE LEADING ZEROS
005212  012701  010000                  MOV     #10000,R1
005216  005301          5$:     DEC     R1
005220  001376                  BNE     5$
005222  005302                  DEC     R2                      ;DECREMENT THE COUNTER
005224  001351                  BNE     3$                      ;NO CARRIAGE RETURN
005226  104400  001171                  TYPE    ,$CRLF                  ;CARRIAGE RETURN
005232  000717                  BR      1$                      ;REPEAT CONVERSION
```

```
1424                                            .SBTTL      LOGIC TEST SECTION
1425  005234  012737  005234  001370  BEGL:     MOV     #BEGL,TADDR         ;TEST ADDRESS
1426  005242  004737  002426                     JSR     PC,TESTAD           ;NO OF ADDITIONAL AD'S
1427  005246  004737  002562           1$:       JSR     PC,BEGINL           ;LOGIC TESTS
1428  005252  004737  005422                     JSR     PC,BUMPAD           ;MORE TO TEST?
1429  005256  000773                             BR      1$                  ;TEST NEXT A/D
1430  005260  012737  005246  011720             MOV     #1$,AGTST           ;ADDRESS FOR EOP
1431  005266  000137  011722                     JMP     $EOP                ;TYPE END OF PASS
1432
1433                                            .SBTTL      AUTO TEST
1434  005272  012737  005272  001370  BEGINA:   MOV     #BEGINA,TADDR       ;TEST ADDRESS
1435  005300  005037  001202                     CLR     $PASS               ;CLEAR PASS COUNTER
1436  005304  004737  002426                     JSR     PC,TESTAD           ;NO. OF AD'S TO BE TESTED
1437  005310  004737  002562           1$:       JSR     PC,BEGINL           ;LOGIC TESTS
1438  005314  104400  012677                     TYPE    ,MEND               ;TYPE END OF LOGIC TEST
1439  005320  013746  001316                     MOV     $TREG,-(SP)         ;SAVE STREG FOR TYPEOUT
1440  005324  104402                             TYPOS                       ;TYPE OCTAL NUMBER
1441  005326     006                             .BYTE   6                   ;TYPE 6 DIGITS
1442  005327     001                             .BYTE   1                   ;TYPE LEADING ZEROS
1443  005330  104400  001171                     TYPE    ,$CRLF              ;TYPE A CR,LF
1444  005334  004737  003712                     JSR     PC,WRAP
1445  005340  004737  005422                     JSR     PC,BUMPAD           ;TEST NEXT A/D
1446  005344  000761                             BR      1$                  ;TEST NEXT AD
1447  005346  012737  005310  011720             MOV     #1$,AGTST           ;ADDRESS FOR EOP
1448  005354  000137  011722                     JMP     $EOP                ;TYPE END OF PASS
1449
1450                                            .SBTTL      WRAPAROUND TEST
1451  005360  012737  005360  001370  BEGINW:   MOV     #BEGINW,TADDR       ;TEST ADDRESS
1452  005366  005037  001202                     CLR     $PASS               ;CLEAR PASS COUNT
1453  005372  004737  002426                     JSR     PC,TESTAD           ;NO. OF AD'S TO BE TESTED
1454  005376  004737  003712           1$:       JSR     PC,WRAP             ;WRAPAROUND TESTS
1455  005402  004737  005422                     JSR     PC,BUMPAD           ;MORE A/D'S TO BE TESTED?
1456  005406  000773                             BR      1$                  ;YES-GO TEST NEXT AD11K
1457  005410  012737  005376  011720             MOV     #1$,AGTST
1458  005416  000137  011722                     JMP     $EOP                ;INCREMENTS $PASS
```

```
1459                                            .SBTTL      DETERMINE IF MORE AD11K'S TO BE TESTED
1460  005422  005737  001360            BUMPAD: TST     NBEXT                   ;ADDITIONAL AD'S?
1461  005426  001424                            BEQ     FIXADR                  ;NO-INITIALIZE ADDRESSES
1462  005430  063737  001332  001316            ADD     VADR,STREG              ;SET UP NEW ST. REG.
1463  005436  063737  001332  001320            ADD     VADR,ADST1              ;SET UP NEW ADST1
1464  005444  063737  001332  001322            ADD     VADR,ADBUFF             ;SET UP NEW BUFFER ADDRESS
1465  005452  063737  001334  001324            ADD     VVCT,VECTOR             ;SET UP NEW VECTOR
1466  005460  063737  001334  001330            ADD     VVCT,VECTR1
1467  005466  005077  173636                    CLR     @VECTR1
1468  005472  005337  001360                    DEC     NBEXT                   ;ONE LESS AD11K
1469  005476  000446                            BR      BYPASS
1470  005500  062716  000002            FIXADR: ADD     #2,(SP)
1471  005504  013737  001250  001316    FIXONE: MOV     $BASE,STREG             ;RELOAD INITIAL ADDRESSES
1472  005512  013737  001250  001320            MOV     $BASE,ADST1
1473  005520  013737  001250  001322            MOV     $BASE,ADBUFF
1474  005526  005237  001320                    INC     ADST1
1475  005532  062737  000002  001322            ADD     #2,ADBUFF
1476  005540  013737  001244  001324            MOV     $VECT1,VECTOR
1477  005546  042737  170000  001324            BIC     #170000,VECTOR
1478  005554  113737  001245  001326            MOVB    $VECT1+1,BASEBR
1479  005562  105037  001327                    CLRB    BASEBR+1                ;CLEAR HIGH BYTE
1480  005566  013737  001324  001330            MOV     VECTOR,VECTR1
1481  005574  062737  000002  001330            ADD     #2,VECTR1
1482  005602  005077  173522                    CLR     @VECTR1
1483  005606  013737  001362  001360            MOV     NMBEXT,NBEXT            ;RESET COUNTER
1484                                        ;;LOAD .+2 AND HALT TRAP CATCH;;
1485  005614  012700  000216            BYPASS: MOV     #216,R0                 ;FILL .+2
1486  005620  012701  000214                    MOV     #214,R1                 ;LOAD HALT
1487  005624  020137  001340            1$:     CMP     R1,KBVECT
1488  005630  001410                            BEQ     2$
1489  005632  010021                            MOV     R0,(R1)+
1490  005634  005021                            CLR     (R1)+
1491  005636  010100                            MOV     R1,R0
1492  005640  005720                            TST     (R0)+
1493  005642  020027  001002                    CMP     R0,#1002
1494  005646  001366                            BNE     1$
1495  005650  000207                            RTS     PC                      ;TEST NEXT A/D
1496  005652  022021                    2$:     CMP     (R0)+,(R1)+
1497  005654  022021                            CMP     (R0)+,(R1)+
1498  005656  000762                            BR      1$
1499
1500
1501                                            .SBTTL      NOISE TEST, 1 EDGE
1502  005660  012737  005660  001370    BEGINN: MOV     #BEGINN,TADDR           ;TEST ADDRESS IN TADDR
1503  005666  104400  012030                    TYPE    ,NOIMSG                 ;ASK FOR CHANNEL
1504  005672  104400  013534                    TYPE    ,ASKCH
1505  005676  017737  173236  001354    1$:     MOV     @SWR,CH1                ;LOAD CHANNEL
1506  005704  042737  177700  001354            BIC     #177700,CH1
1507  005712  012737  000200  001352            MOV     #200,TEMP               ;LOAD DAC VALUE
1508  005720  004537  010312                    JSR     R5,NOITST               ;GO TO NOISE SUBROUTINE
1509  005724  001354                            CH1
1510  005726  000763                            BR      1$
```

```
1511                                            .SBTTL      INTERCHANNEL SETTLING TEST, 1 EDGE
1512  005730  012737  005730  001370  BEGIN5: MOV     #BEGIN5,TADDR           ;TEST ADDRESS IN TADDR
1513  005736  104400  012050                  TYPE    ,SETMSG                 ;ASK FOR CHANNELS
1514  005742  104406                          RDOCT
1515  005744  012637  001354                  MOV     (SP)+,CH1
1516  005750  104400  012335                  TYPE    ,TOMSG
1517  005754  104406                          RDOCT
1518  005756  012637  001356                  MOV     (SP)+,CH2
1519  005762  012737  000200  001352  BK3:    MOV     #200,TEMP               ;LOAD DAC
1520  005770  013737  001356  001366          MOV     CH2,CHANL
1521  005776  004737  006336                  JSR     PC,GETEDG               ;GET EDGE VALUES
1522  006002  005002                          CLR     R2
1523  006004  004737  006170                  JSR     PC,SET1A                ;SCALING = .02 LSB
1524  006010  004737  006170                  JSR     PC,SET1A                ;MAKE IT .01 LSB
1525  006014  100001                          BPL     POSR2
1526  006016  005402                          NEG     R2
1527  006020  010204          POSR2:  MOV     R2,R4
1528  006022  012737  000001  006514          MOV     #1,EDGFLG
1529  006030  004737  006036                  JSR     PC,TYPSET
1530  006034  000752                          BR      BK3
1531  006036  004737  011406          TYPSET: JSR     PC,DECTYP
1532  006042  104400  012231                  TYPE    ,LSB
1533  006046  013746  001356                  MOV     CH2,-(SP)       ;;SAVE CH2 FOR TYPEOUT
1534                                                                  ;;TYPE CH
1535  006052  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1536  006054     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1537  006055     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1538  006056  104403  013626                  TYPE    ,MAT            ;TYPE ASCIZ STRING
1539  006062  004737  006452                  JSR     PC,TYPEDG
1540  006066  104400  012244                  TYPE    ,SETCH
1541  006072  013746  001354                  MOV     CH1,-(SP)       ;;SAVE CH1 FOR TYPEOUT
1542                                                                  ;;TYPE CH
1543  006076  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1544  006100     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1545  006101     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1546  006102  104400  012266                  TYPE    ,ATMSG
1547  006106  013737  001354  006134          MOV     CH1,1$
1548  006114  163737  001336  006134          SUB     BASECH,1$
1549  006122  012777  000200  173172          MOV     #200,@ADBUFF
1550  006130  004537  011004                  JSR     R5,CONVRT
1551  006134  000000                  1$:     0
1552  006136  013746  001352                  MOV     TEMP,-(SP)      ;;SAVE TEMP FOR TYPEOUT
1553                                                                  ;;TYPE VALUE
1554  006142  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1555  006144     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
1556  006145     001                          .BYTE   1               ;;TYPE LEADING ZEROS
1557  006146  020437  011650                  CMP     R4,VSET
1558  006152  003003                          BGT     ERR
1559  006154  104400  012376                  TYPE    ,OKMSG
1560  006160  000207                          RTS     PC
```

```
                                ERR:    TYPE    ERMSG
                                        RTS     PC


                                ;;SUBROUTINE FOR SETTLING TESTS;;
001356  001364  SETIA:  MOV     CH2,DUMMY               ;LOAD DUMMY
006516                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
000062                  50.
001410                  ADD     DAC,R2          ;ADD RESULT TO R2
001354  001364          MOV     CH1,DUMMY               ;CHANGE DUMMY VALUE
006516                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%.
000062                  50.
001410                  SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
000207                  RTS     PC              ;RETURN

001354          SETB:   MOV     (R5)+,CH1               ;GET FIRST CHANNEL
001356                  MOV     (R5)+,CH2               ;GET SECOND CHANNEL
001336  001354          ADD     BASECH,CH1
001336  001356          ADD     BASECH,CH2
006336                  JSR     PC,GETEDG               ;GET EDGE VALUES
005002                  CLR     R2
000010                  MOV     #10,R3          ;SET UP COUNTER
006170          SETAA:  JSR     PC,SETIA                ;GET SETTLE VALUES
001414                  INC     EDGE
005303                  DEC     R3
001372                  BNE     SETAA           ;REPEAT 8 TIMES
000010  001414          SUB     #10,EDGE
005702                  TST     R2
100001                  BPL     R2POS
005402                  NEG     R2
010204          R2POS:  MOV     R2,R4
012737  000010  001414  MOV     #8.,EDGFLG
004737  006036          JSR     PC,TYPSET               ;TYPE OUT RESULTS
000205                  RTS     R5                      ;RETURN
```

```
                                        ;SUBROUTINE TO GET EDGE VALUE
                                        ;CALL=JSR   PC,GETEDG
                                        ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                        ;RESULT IN EDGE. USES R0
006336  013777  001352  172756  GETEDG: MOV     TEMP,@ADBUFF
006344  113700  001366                  MOVB    CHANL,R0                ;GET CHANNEL
006350  000300                          SWAB    R0              ;SET UP A.D STATUS REG.
006352  052700  000100                  BIS     #100,R0         ;ENABLE INTRPT.
006356  010077  172734                  MOV     R0,@STREG
006362  012700  000100                  MOV     #100,R0         ;DAC SETTLING DELAY
006366  005300                  1$:     DEC     R0
006370  001376                          BNE     1$
006372  005037  001414                  CLR     EDGE
006376  012700  000010                  MOV     #10,R0
006402  012777  001610  172714          MOV     #RETURN,@VECTOR         ;RETURN ADDRESS
006410  005277  172702          CONV:   INC     @STREG                  ;START CONVERSION
006414  000001                          WAIT                            ;WAIT FOR INTERRUPT
006416  067737  172700  001414          ADD     @ADBUFF,EDGE
006424  005300                          DEC     R0
006426  001370                          BNE     CONV
006430  006237  001414                  ASR     EDGE
006434  006237  001414                  ASR     EDGE
006440  006237  001414                  ASR     EDGE
006444  005537  001414                  ADC     EDGE
006450  000207                          RTS     PC

                                        ;;SUBROUTINE TO TYPE EDGE VALUES::
006452  013703  001414          TYPEDG: MOV     EDGE,R3
006456  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE OCTAL VALUE OF EDGE
006460  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006462     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006463     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006464  023727  006514  000000          CMP     EDGFLG,#1
006472  001407                          BEQ     RET
006474  062703  000007                  ADD     #7,R3
006500  104400  013576                  TYPE    ,C1             ;TYPE ASCIZ STRING
006504  010346                          MOV     R3,-(SP         ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE EDGE VALUE
006506  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006510     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006511     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006512  000207          RET:    RTS     PC
006514  000000          EDGFLG: 0
```

```
1640                                        ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
1641                                        ;CALL=JSR   R5,SARSUB
1642                                        ;   XXX;XXX=PERCENT
1643                                        ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
1644  006516  012537  001426          SARSUB: MOV     (R5)+,PERCNT        ;GET PERCENT
1645  006522  006337  001426                  ASL     PERCNT
1646  006526  006337  001426                  ASL     PERCNT
1647  006532  006337  001426                  ASL     PERCNT              ;RESCALE PERCENT FOR 1600.
1648  006536  006337  001426                  ASL     PERCNT              ;POINTS PER BURST
1649  006542  012737  000200  001416  SAR1:   MOV     #200,BITPNT         ;INITIALIZE BIT POINTER AT MSB
1650  006550  005037  001410                  CLR     DAC                 ;INITIALIZE DAC VALUE
1651  006554  005000                  TRY:    CLR     R0
1652  006556  063737  001416  001410          ADD     BITPNT,DAC          ;TRY BIT
1653  006564  013777  001410  172530          MOV     DAC,@ADBUFF
1654  006572  012737  000100  001412          MOV     #100,DELAY
1655  006600  005337  001412          1$:     DEC     DELAY               ;STALL TIME
1656  006604  001375                          BNE     1$
1657  006606  012701  003100                  MOV     #1600.,R1           ;SET UP FOR 1600. CONVERSIONS
1658  006612  113777  001364  172500  NXTCVT: MOVB    DUMMY,@ADST1        ;PRESET MUX TO DUMMY CHANNEL
1659  006620  012777  001610  172476          MOV     #RETURN,@VECTOR     ;RETURN ADDRESS
1660  006626  052777  000101  172462          BIS     #101,@STREG         ;CONVERSION ON DUMMY CHANNEL
1661  006634  000001                          WAIT                        ;WAIT FOR INTERRUPT
1662  006636  017704  172460                  MOV     @ADBUFF,R4          ;DUMMY READ
1663  006642  013704  001366                  MOV     CHANL,R4
1664  006646  000304                          SWAB    R4
1665  006650  052704  000101                  BIS     #101,R4             ;INTERRUPT ENABLE START
1666  006654  010477  172436                  MOV     R4,@STREG           ;JUMP TO CHANNEL + START CONVERT
1667  006660  000001                          WAIT                        ;WAIT FOR INTERRUPT
1668  006662  027737  172434  001414          CMP     @ADBUFF,EDGE
1669  006670  002001                          BGE     2$
1670  006672  005200                          INC     R0                  ;COUNT RESULTS .LT. EDGE
1671  006674  005301                  2$:     DEC     R1
1672  006676  001345                          BNE     NXTCVT
1673  006700  020037  001426                  CMP     R0,PERCNT
1674  006704  003003                          BGT     SHIFT
1675  006706  163737  001416  001410          SUB     BITPNT,DAC          ;TAKE THE BIT OUT
1676  006714  006237  001416          SHIFT:  ASR     BITPNT
1677  006720  001315                          BNE     TRY
1678  006722  000205                          RTS     R5
```

```
                                        ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
006724 004400 013022            DIFLIN: TYPE    ,MSG20
006730 005037 001430                    CLR     OUT
006734 012700 017756                    MOV     #BUFFER,R0
006740 012701 010000                    MOV     #4096.,R1       ;4096 WORDS FOR HISTOGRAM
006744 005020                   CLEAR1: CLR     (R0)+           ;CLEAR BUFFER AREA
006746 005301                           DEC     R1
006750 001375                           BNE     CLEAR1
006752 012700 017136                    MOV     #DIST,R0        ;DISTRIBUTION BUFFER POINTER
006756 012701 000310                    MOV     #200.,R1        ;200. WORDS FOR DISTRIBUTION
006762 005003                           CLR     R3
006764 005037 001430                    CLR     OUT
006770 005037 001342                    CLR     WIDE
006774 005037 001344                    CLR     NARROW
007000 005037 001346                    CLR     FIRST
007004 005037 001350                    CLR     SKIPST
007010 005020                   CLEAR2: CLR     (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
007012 005301                           DEC     R1
007014 001375                           BNE     CLEAR2
007016 012700 000011            CHANNL: MOV     #11,R0          ;CHANNEL 11
007022 063700 001336                    ADD     BASECH,R0
007026 000300                           SWAB    R0                              ;LOAD MUX BITS
007030 052700 000100                    BIS     #100,R0
007034 010077 172256                    MOV     R0,@STREG
007040 012700 001440                    MOV     #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
007044 012777 001610 172252             MOV     #RETURN,@VECTOR
007052 012701 007776            AGAIN:  MOV     #4094.,R1
007056 004737 010722            NEXT:   JSR     PC,RANDY        ;GET RANDOM NUMBER
007062 013702 001372                    MOV     RNA,R2
007066 042702 177760                    BIC     #177760,R2      ;MASK IT TO 4 BITS ONLY
007072 001402                           BEQ     CONVR
007074 005302                   DELAY3: DEC     R2              ;STALL
007076 001376                           BNE     DELAY3          ;TIME
007100 005277 172212            CONVR:  INC     @STREG          ;START CONVERSION
007104 000001                           WAIT
007106 017702 172210                    MOV     @ADBUFF,R2      ;GET CONVERTED VALUE
007112 001413                           BEQ     DELAY1          ;IGNORE IF =0
007114 020227 007777                    CMP     R2,#7777        ;IGNORE IF =7777
007120 001413                           BEQ     DELAY2
007122 006302                           ASL     R2
007124 005262 017756                    INC     BUFFER(R2)      ;MAKE HISTOGRAM
007130 100013                           BPL     OKAY
007132 012762 077777 017756             MOV     #077777,BUFFER(R2)      ;PREVENT OVERFLOW
007140 000407                           BR      OKAY
007142 020227 007777            DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
007146 001400                           BEQ     DELAY2          ;WITH DUMMY INSTR.
007150 005201                   DELAY2: INC     R1
007152 005263 001352                    INC     TEMP(R3,
007156 100403                           BMI     NOTOK
007160 005301                   OKAY:   DEC     R1
007162 001335                           BNE     NEXT
007164 000403                           BR      AROUND
007166 005037 001352            NOTOK:  CLR     TEMP
007172 000772                           BR      OKAY
```

```
1733  007174  005300                  AROUND: DEC     R0
1734  007176  001325                          BNE     AGAIN
1735  007200  012700  007776                  MOV     #4094.,R0
1736  007204  012701  017760                  MOV     #BUFFER+2,R1
1737  007210  012102                  READ:   MOV     (R1)+,R2        ;GET STATE WIDTH
1738  007212  006202                          ASR     R2              ;1 LSB = 800.
1739  007214  006202                          ASR     R2
1740  007216  006202                          ASR     R2
1741  007220  005502                          ADC     R2              ;1 LSB = 100.
1742  007222  020227  000310                  CMP     R2,#200.        ;OUT OF RANGE?
1743  007226  002403                          BLT     INRNGE
1744  007230  005237  001430                  INC     OUT             ;YES - INCREMENT COUNTER
1745  007234  000423                          BR      TYPBAD
1746  007236  006302                  INRNGE: ASL     R2
1747  007240  005262  017136                  INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
1750  007244  006202                          ASR     R2
1751  007246  020227  000062                  CMP     R2,#50.         ;IS IT 1/2 LSB?
1752  007252  002007                          BGE     NOTNAR
1753  007254  005237  001344                  INC     NARROW
1754  007260  005702                          TST     R2              ;IS IT A SKIPPED STATE?
1755  007262  001002                          BNE     31$
1756  007264  005237  001350                  INC     SKIPST
1757  007270  000405                  31$:    BR      TYPBAD
1760  007272  020227  000226          NOTNAR: CMP     R2,#150.        ;IS IT 1.5 LSB?
1761  007276  003426                          BLE     LAST
1762  007300  005237  001342                  INC     WIDE
1763  007304  005737  001346          TYPBAD: TST     FIRST
1764  007310  001004                          BNE     60$
1765  007312  005237  001346                  INC     FIRST
1766  007316  104400  012201                  TYPE    ,STATE
1767  007322  010103                  60$:    MOV     R1,R3
1770  007324  162703  017760                  SUB     #BUFFER+2,R3
1771  007330  006203                          ASR     R3
1772  007332  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
1773                                                                  ;;TYPE STATE
1774  007334  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1775  007336     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
1776  007337     001                          .BYTE   1               ;;TYPE LEADING ZEROS
1777  007340  104400  012175                  TYPE    ,DASH
1778  007344  004737  011406                  JSR     PC,DECTYP
1779  007350  104400  012166                  TYPE    ,LSBMSG
1780  007354  005300                  LAST:   DEC     R0
1781  007356  001314                          BNE     READ
1782  007360  112737  000177  014465          MOVB    #177,DECPNT
1783  007366  013702  001350                  MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
1784  007372  004737  011406                  JSR     PC,DECTYP               ;TYPE IT
1785  007376  104400  012424                  TYPE    ,SKPMSG         ;TYPE MESSAGE
1786  007402  005737  001350                  TST     SKIPST
1787  007406  001403                          BEQ     1$
1788  007410  104400  012407                  TYPE    ,ERMSG          ;TYPE "ERROR"
1789  007414  000402                          BR      NAP
1790  007416  104400  012376          1$:     TYPE    ,OKMSG          ;TYPE #OK#
```

```
 1785  007422  013702  001344          NAR:    MOV     NARROW,R2               ;GET NO. OF NARROW STATES
 1786  007426  004737  011406                  JSR     PC,DECTYP               ;TYPE IT
 1787  007432  104400  012446                  TYPE    ,NARMSG         ;TYPE MESSAGE
 1788  007436  013702  001342                  MOV     WIDE,R2
 1789  007442  063702  001430                  ADD     OUT,R2
 1790  007446  004737  011406                  JSR     PC,DECTYP               ;TYPE NO. OF WIDE STATES
 1791  007452  104400  012505                  TYPE    ,WIDMSG         ;TYPE MESSAGE
 1792  007456  013702  001430                  MOV     OUT,R2
 1793  007462  004737  011406                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE 2 LSB
 1794  007466  104400  012544                  TYPE    ,OUTMSG         ;TYPE MESSAGE
 1795  007472  005737  001430                  TST     OUT
 1796  007476  001403                          BEQ     11$
 1797  007500  104400  012407                  TYPE    ,ERMSG          ;TYPE "ERROR"
 1798  007504  000402                          BR      HALF
 1799  007506  104400  012376          11$:    TYPE    ,OKMSG          ;TYPE "OK"
 1800  007512  013702  001344          HALF:   MOV     NARROW,R2
 1801  007516  063702  001342                  ADD     WIDE,R2
 1802  007522  063702  001430                  ADD     OUT,R2
 1803  007526  010200                          MOV     R2,R0
 1804  007530  004737  011406                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE LIMITS
 1805  007534  112737  000056  014465          MOVB    #56,DECPNT
 1806  007542  104400  012577                  TYPE    ,HAFMSG
 1807  007546  020027  000051                  CMP     R0,#41.         ;COMPARE IT TO NOMINAL
 1808  007552  003403                          BLE     21$
 1809  007554  104400  012407                  TYPE    ,ERMSG          ;TYPE "ERROR"
 1810  007560  000402                          BR      SWDIST
 1811  007562  104400  012376          21$:    TYPE    ,OKMSG          ;TYPE "OK"
 1812  007566  005737  001404          SWDIST: TST     FLAG            ;VT55?
 1813  007572  001426                          BEQ     RELACC
 1814  007574  004737  010254                  JSR     PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
 1815  007600  104400  013054                  TYPE    ,MSG16
 1816  007604  104400  013655                  TYPE    ,BUFF1          ;TYPE BUFF1-PRINT GRID
 1817  007610  012700  017136                  MOV     #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
 1818  007614  012701  000310                  MOV     #200.,R1        ;GO 200. TIMES UP TO 2 LSB
 1819  007620  012002          NXTY1:  MOV     (R0)+,R2
 1820  007622  004737  011304                  JSR     PC,LOADY
 1821  007626  005002                          CLR     R2
 1822  007630  004737  011304                  JSR     PC,LOADY
 1823  007634  005301                          DEC     R1
 1824  007636  001370                          BNE     NXTY1
 1825  007640  104400  013600                  TYPE    ,C2             ;TYPE ASCIZ STRING
 1826  007644  004737  010254                  JSR     PC,DELCLR
 1827
```

```
                                        ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR

007650  005001                RELACC: CLR     R1              ;RUNNING ERROR = 0
007652  005003                        CLR     R3              ;MAXIMUM ERROR = 0
007654  104400  013447                TYPE    ,MSG21
007660  012700  017760                MOV     #BUFFER+2,R0
007664  011002                NXTSTA: MOV     (R0),R2         ;STATE WIDTH = R2
007666  162702  001440                SUB     #800.,R2        ;STATE WIDTH ERROR IN R2
007672  060201                        ADD     R2,R1           ;UPDATE RUNNING ERROR
007674  010120                        MOV     R1,(R0)+        ;SAVE IN BUFFER
007676  010104                        MOV     R1,R4           ;SAVE IN R4 ALSO
007700  100001                        BPL     PLUS            ;IS IT POSITIVE?
007702  005404                        NEG     R4              ;NO - MAKE IT POSITIVE
007704  020403                PLUS:   CMP     R4,R3           ;CHECK AGAINST PREVIOUS MAX. ERROR
007706  003405                        BLE     NOTNEW          ;NOT A NEW MAXIMUM
007710  010403                        MOV     R4,R3           ;UPDATE MAXIMUM IN R3
007712  010005                        MOV     R0,R5
007714  162705  017760                SUB     #BUFFER+2,R5
007720  006205                        ASR     R5              ;R5=EDGE VALUE AT MAX. RELACC
007722  020027  037754                NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
007726  001356                        BNE     NXTSTA          ;NO - REPEAT
007730  006203                        ASR     R3              ;RESCALE FROM 1 LSB = 800. SCALING
007732  006203                        ASR     R3              ;TO 1 LSB = 100. SCALING
007734  006203                        ASR     R3
007736  005503                        ADC     R3
007740  010302                        MOV     R3,R2
007742  004737  011406                JSR     PC,DECTYP
007746  104400  013474                TYPE    ,LINEA
007752  010546                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                              ;;TYPE VALUE
007754  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
007756     004                      .BYTE   4               ;;TYPE 4 DIGIT(S)
007757     001                      .BYTE   1               ;;TYPE LEADING ZEROS
007760  104400  012333                TYPE    ,SLASH                  ;PRINT '/'
007764  005205                        INC     R5
007766  010546                        MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
                                                              ;;TYPE VALUE
007770  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
007772     004                      .BYTE   4               ;;TYPE 4 DIGIT(S)
007773     001                      .BYTE   1               ;;TYPE LEADING ZEROS
007774  020337  011652                CMP     R3,VLIN
010000  003403                        BLE     41$
010002  104400  012407                TYPE    ,ERMSG
010006  000402                        BR      42$
010010  104400  012376        41$:    TYPE    ,OKMSG
010014  005737  001404        42$:    TST     FLAG            ;VT55?
010020  001503                        BEQ     L02
010022  012700  017756                MOV     #BUFFER,R0
010026  012701  010000                MOV     #4096.,R1
```

```
 1877  010032  011002                GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
 1878  010034  006202                        ASR     R2              ;RESCALE IT TO 1 LSB = 100.
 1879  010036  006202                        ASR     R2
 1880  010040  006202                        ASR     R2
 1881  010042  005502                        ADC     R2
 1882  010044  062702  000166              ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
 1883  010050  010220                      MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
 1884  010052  005301                      DEC     R1
 1885  010054  001366                      BNE     GETDAT
 1886  010056  012700  017756              MOV     #BUFFER,R0
 1887  010062  012704  017756              MOV     #BUFFER,R4
 1888  010066  012705  017760              MOV     #BUFFER+2,R5
 1889  010072  012701  001000              MOV     #512.,R1
 1890  010076  012702  000007      NXT8:   MOV     #7.,R2
 1891  010102  012003                      MOV     (R0)+,R3
 1892  010104  010337  001420              MOV     R3,MIN          ;MINIMUM
 1893  010110  010337  001424              MOV     R3,MAX          ;MAXIMUM
 1894  010114  012003              NXTCMP: MOV     (R0)+,R3
 1895  010116  020337  001420              CMP     R3,MIN
 1896  010122  002002                      BGE     MAXTST
 1897  010124  010337  001420              MOV     R3,MIN          ;NEW MINIMUM
 1898  010130  020337  001424      MAXTST: CMP     R3,MAX
 1899  010134  003402                      BLE     TST8
 1900  010136  010337  001424              MOV     R3,MAX          ;NEW MAXIMUM
 1901  010142  005302              TST8:   DEC     R2
 1902  010144  001363                      BNE     NXTCMP
 1903  010146  013724  001420              MOV     MIN,(R4)+
 1904  010152  013725  001424              MOV     MAX,(R5)+
 1905  010156  022425                      CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
 1906  010160  005301                      DEC     R1
 1907  010162  001345                      BNE     NXT8
 1908  010164  104400  012762              TYPE    ,MSG18
 1909  010170  104400  013703              TYPE    ,BUFF2          ;TYPE BUFF2
 1910  010174  012700  017756              MOV     #BUFFER,R0
 1911  010200  004737  010232              JSR     PC,LOAD
 1912  010204  104400  013603              TYPE    ,C3             ;TYPE ASCIZ STRING
 1913  010210  012700  017760              MOV     #BUFFER+2,R0
 1914  010214  004737  010232              JSR     PC,LOAD
 1915  010220  104400  013600              TYPE    ,C2             ;TYPE ASCIZ STRING
 1916  010224  004737  010254              JSR     PC,DELCLR
 1917  010230  000207              LO2:    RTS     PC
 1918  010232  012701  001000      LOAD:   MOV     #512.,R1
 1919  010236  012002              LOADO:  MOV     (R0)+,R2
 1920  010240  005720                      TST     (R0)+
 1921  010242  004737  011304              JSR     PC,LOADY
 1922  010246  005301                      DEC     R1
 1923  010250  001372                      BNE     LOADO
 1924  010252  000207                      RTS     PC
```

```
1925  010254  005000                    DELCLR: CLR     R0
1926  010256  012701  000020                    MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
1927  010262  005300                    1$:     DEC     R0
1928  010264  001376                            BNE     1$
1929  010266  005301                            DEC     R1
1930  010270  001374                            BNE     1$
1931  010272  032777  010000  170640            BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
1932  010300  001401                            BEQ     2$              ;;DON'T HALT FOR DISPLAY
1933  010302  000000                            HALT
1934  010304  104400  013723            2$:     TYPE    ,VTINIT
1935  010310  000207                            RTS     PC
1936                                    ;;NOISE SUBROUTINE;;
1937  010312  013537  001366            NOITST: MOV     @(R5)+,CHANL            ;LOAD CHANNEL
1938  010316  013737  001366  001364            MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
1939  010324  004737  006336                    JSR     PC,GETEDG               ;GET EDGE VALUE
1940  010330  004737  010504                    JSR     PC,NOIA                 ;GET RMS AND PEAK VALUES
1941  010334  012737  000001  006514            MOV     #1,EDGFLG
1942  010342  004737  010350                    JSR     PC,TYPRP                ;TYPE RMS AND PEAK VALUES
1943  010346  000205                            RTS     R5
1944
1945
1946
1947
1948
1949                                    ;;TYPE RMS AND PEAK VALUES;;
1950  010350  104400  012273            TYPRP:  TYPE    ,NOI
1951  010354  005737  001400                    TST     RMS
1952  010360  100002                            BPL     POSRMS
1953  010362  005037  001400                    CLR     RMS                     ;RMS<0,SET RMS=0
1954  010366  005737  001402            POSRMS: TST     PEAK
1955  010372  100002                            BPL     POSPEA
1956  010374  005037  001402                    CLR     PEAK                    ;PEAK<0,SET PEAK=0
1957  010400  013702  001400            POSPEA: MOV     RMS,R2
1958  010404  004737  011406                    JSR     PC,DECTYP
1959  010410  104400  012646                    TYPE    ,MESR
1960  010414  013702  001402                    MOV     PEAK,R2
1961  010420  004737  011406                    JSR     PC,DECTYP
1962  010424  104400  012661                    TYPE    ,MESP
1963  010430  004737  006452                    JSR     PC,TYPEDG
1964  010434  104400  012303                    TYPE    ,CHAN
1965  010440  013746  001366                    MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
1966                                                                    ;;TYPE CHANL
1967  010444  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
1968  010446     002                            .BYTE   2               ;;TYPE 2 DIGIT(S)
1969  010447     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
1970  010450  023737  001400  011644            CMP     RMS,VNR         ;WITHIN LIMITS?
1971  010456  003007                            BGT     ER
1972  010460  023737  001402  011646            CMP     PEAK,VNP                ;WITHIN LIMITS?
1973  010466  003003                            BGT     ER
1974  010470  104400  012376                    TYPE    ,OKMSG
1975  010474  000207                            RTS     PC
1976  010476  104400  012407            ER:     TYPE    ,ERMSG
1977  010502  000207                            RTS     PC
```

```
                                        ;;SUBROUTINES FOR NOISE TEST;;
010504  005037  001400          NOIA:   CLR     RMS                     ;CLEAR RMS VLAUE
010510  005037  001402                  CLR     PEAK                    ;CLEAR PEAK VALUE
010514  004537  006516          NOI1:   JSR     R5,SARSUB               ;DO SAR ROUTINE AT 16%
010520  000020                          16.
010522  063737  001410  001400          ADD     DAC,RMS         ;ADD RESULT TO RMS
010530  004537  006516                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 84%
010534  000124                          84.
010536  163737  001410  001400          SUB     DAC,RMS         ;SUBTRACT RESULT FROM RMS
010544  004537  006516                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 1%
010550  000001                          1
010552  063737  001410  001402          ADD     DAC,PEAK                ;ADD RESULT TO PEAK
010560  004537  006516                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 99%
010564  000143                          99.
010566  163737  001410  001402          SUB     DAC,PEAK                ;SUBTRACT RESULT FROM PEAK
010574  000207                          RTS     PC                      ;RETURN

010576  012537  001366          NOI8:   MOV     (R5)+,CHANL             ;GET CHANNEL VALUE
010602  063737  001336  001366          ADD     BASECH,CHANL
010610  013737  001366  001364          MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
010616  004737  006236                  JSR     PC,GETEDG               ;GET EDGE VALUES
010622  005037  001400                  CLR     RMS             ;CLEAR RMS VALUE
010626  005037  001402                  CLR     PEAK            ;CLEAR PEAK VALUE
010632  012737  000010  010720          MOV     #10,10$         ;SET UP COUNTER
010640  004737  010514          1$:     JSR     PC,NOI1         ;GET NOISE VALUES
010644  005237  001414                  INC     EDGE
010650  005337  010720                  DEC     10$
010654  001371                          BNE     1$              ;REPEAT 8 TIMES
010656  162737  000010  001414          SUB     #10,EDGE
010664  006237  001400                  ASR     RMS             ;SCALE IT TO 1 LSB=100.
010670  005537  001400                  ADC     RMS
010674  006237  001402                  ASR     PEAK
010700  005537  001402                  ADC     PEAK
010704  012737  000010  006514          MOV     #8.,EDGFLG
010712  004737  010350                  JSR     PC,TYPRP                        TYPE RESULTS
010716  000205                          RTS     R5              ;RETURN
010720  000000                  10$:    0                       ;COUNTER


                                        ;;RANDOM NUMBER GENERATOR;;
010722  063737  001374  001372  RANDY:  ADD     RNB,RNA
010730  063737  001376  001372          ADD     RNC,RNA
010736  005537  001372                  ADC     RNA
010742  063737  001372  001374          ADD     RNA,RNB
010750  063737  001376  001374          ADD     RNC,RNB
010756  005537  001374                  ADC     RNB
010762  063737  001372  001376          ADD     RNA,RNC
010770  063737  001374  001376          ADD     RNB,RNC
010776  005537  001376                  ADC     RNC
011002  000207                          RTS     PC
```

```
2028                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
2029  011004  012500                  CONVRT: MOV     (R5)+,R0            ;GET CHANNEL VALUE
2030  011006  063700  001336                  ADD     BASECH,R0
2031  011012  010037  001366                  MOV     R0,CHANL
2032  011016  000300                          SWAB    R0
2033  011020  005037  001352                  CLR     TEMP
2034  011024  010077  170266                  MOV     R0,@STREG           ;LOAD CHANNEL INTO MIX BITS
2035  011030  012700  010000                  MOV     #10000,R0
2036  011034  005300                  2$:     DEC     R0
2037  011036  001376                          BNE     2$
2038  011040  012777  001610  170256          MOV     #RETURN,@VECTOR     ;LOAD VECTOR
2039  011046  012700  000010                  MOV     #10,R0          ;SET UP COUNTER
2040  011052  152777  000101  170236  1$:     BISB    #101,@STREG         ;SET INTRPT. EN., START CONV.
2041  011060  000001                          WAIT                    ;WAIT FOR CONVERSION
2042  011062  067737  170234  001352          ADD     @ADBUFF,TEMP        ;READ BUFFER
2043  011070  005300                          DEC     R0
2044  011072  001367                          BNE     1$              ;DO 8 TIMES
2045  011074  006237  001352                  ASR     TEMP            ;AVERAGE VALUE
2046  011100  006237  001352                  ASR     TEMP
2047  011104  006237  001352                  ASR     TEMP
2048  011110  005537  001352                  ADC     TEMP
2049  011114  000205                          RTS     R5              ;RETURN
2050
2051                                        ;COMPARE $GDDAT AND $BDDAT;;
2052  011116  012537  001124          COMPAR: MOV     (R5)+,$GDDAT            ;GET GOOD DATA
2053  011122  013537  001406                  MOV     @(R5)+,SPREAD           ;GET SPREAD
2054  011126  013737  001352  001126          MOV     TEMP,$BDDAT             ;GET BAD(ACTUAL) DATA
2055  011134  013701  001126                  MOV     $BDDAT,R1
2056  011140  013700  001124                  MOV     $GDDAT,R0
2057  011144  160100                          SUB     R1,R0           ;GET DIFFERENCE
2058  011146  100001                          BPL     7$
2059  011150  005400                          NEG     R0
2060  011152  020037  001406          7$:     CMP     R0,SPREAD               ;COMPARE IT TO SPREAD
2061  011156  003001                          BGT     10$             ;GO TO ERROR PRINTOUT
2062  011160  005725                          TST     (R5)+           ;BUMP RETURN POINTER AROUND ERROR CALL
2063  011162  000205                  10$:    RTS     R5
```

```
2064                                        ;;SUBROUTINE TO TYPE INTRPT. TST MSG.;;
2065  011164  005737  001202          DUMW: TST     $PASS
2066  011170  001021                       BNE     20$
2067  011172  012737  011234  001110       MOV     #20$,$LPERR
2068  011200  012737  011234  001106       MOV     #20$,$LPADR
2069  011206  104400  013633               TYPE    ,METST          ;TYPE ASCIZ STRING
2070  011212  010046                       MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2071                                                                ;;TYPE TEST NO.
2072  011214  104402                       TYPOS                   ;;GO TYPE--OCTAL ASCII
2073  011216     002                       .BYTE   2               ;;TYPE 2 DIGIT(S)
2074  011217     000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
2075  011220  104400  012723               TYPE    ,ONAD
2076  011224  013746  001316               MOV     $TREG,-(SP)     ;;SAVE STREG FOR TYPEOUT
2077                                                                ;;TYPE BUS ADDRESS
2078  011230  104402                       TYPOS                   ;;GO TYPE--OCTAL ASCII
2079  011232     006                       .BYTE   6               ;;TYPE 6 DIGITS
2080  011233     001                       .BYTE   1               ;;TYPE LEADING ZEROS
2081  011234  000207               20$:    RTS     PC
2082
2083  011236  005737  001202          DUMC: TST     $PASS
2084  011242  001010                       BNE     30$
2085  011244  012737  011264  001110       MOV     #30$,$LPERR
2086  011252  012737  011264  001106       MOV     #30$,$LPADR
2087  011260  104400  012320               TYPE    ,DONE
2088  011264  000207               30$:    RTS     PC
```

```
                                        ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011266  000005                  RST:    RESET
011270  052777  000100  167646          BIS     #100,@#TKS
011276  005037  177776                  CLR     PSW
011302  000207                          RTS     PC

                                        ;SUBROUTINE LOADY;
011304  005702                  LOADY:  TST     R2              ;ROUTINE TO LOAD VLAUE INTO R2
011306  100001                          BPL     PLUSR2          ;AS A VT55 Y-VALUE
011310  005002                          CLR     R2
011312  020227  000353          PLUSR2: CMP     R2,#235.
011316  002402                          BLT     LESS
011320  012702  000353                  MOV     #235.,R2
011324  010203                  LESS:   MOV     R2,R3
011326  042702  177740                  BIC     #177740,R2
011332  052702  000040                  BIS     #40,R2
011336  105777  167606          B10:    TSTB    @#TPS           ;PRINT CHARACTER
011342  100375                          BPL     B10
011344  110277  167602                  MOVB    R2,@#TPB
011350  006203                          ASR     R3
011352  006203                          ASR     R3
011354  006203                          ASR     R3
011356  006203                          ASR     R3
011360  006203                          ASR     R3
011362  042703  177770                  BIC     #177770,R3
011366  052703  000040                  BIS     #40,R3
011372  105777  167552          B11:    TSTB    @#TPS           ;PRINT CHARACTER
011376  100375                          BPL     B11
011400  110377  167546                  MOVB    R3,@#TPB
011404  000207                          RTS     PC
```

```
                                        ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
                                        ;;IN R2 AS X.XX;;
011406  005702                DECTYF: TST     R2              ;TEST VALUE TO BE TYPED
011410  100003                        BPL     POS
011412  104400  012135                TYPE    ,MINUS                  ;TYPE MINUS SIGN
011416  005402                        NEG     R2
011420  020227  001747        POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
011424  003402                        BLE     OKAYD
011426  012702  001747                MOV     #999.,R2
011432  105037  014467        OKAYD:  CLRB    ONES            ;CLEAR ONES
011436  105037  014466                CLRB    TENS            ;CLEAR TENS
011442  105037  014464                CLRB    HUNS            ;CLEAR HUNS
011446  005702                TESTR2: TST     R2                      ;CONVERT VALUE TO A DECIMAL VALUE
011450  001424                        BEQ     TYPOUT
011452  005302                        DEC     R2
011454  105237  014467                INCB    ONES
011460  123727  014467  000012        CMPB    ONES,#10.
011466  001367                        BNE     TESTR2
011470  105037  014467                CLRB    ONES
011474  105237  014466                INCB    TENS
011500  123727  014466  000012        CMPB    TENS,#10.
011506  001357                        BNE     TESTR2
011510  105037  014466                CLRB    TENS
011514  105237  014464                INCB    HUNS
011520  000752                        BR      TESTR2
011522  152737  000060  014464  TYPOUT: BISB  #60,HUNS                ;PREPARE FOR TYPOUT
011530  152737  000060  014466        BISB    #60,TENS
011536  152737  000060  014467        BISB    #60,ONES
011544  104400  014464                TYPE    ,HUNS           ;TYPE VALUE
011550  000207                        RTS     PC

011552  012701  011644        WFADJ:  MOV     #VNR,R1                 ;SUBROUTINE TO SET UP LIMITS
011556  005737  001336                TST     BASECH                  ;TESTING AN AM11K'
011562  001403                        BEQ     1$              ;;
011564  012702  011676                MOV     #VARLT3,R2              ;BASECH NOT ZERO, USE AM11K LIMITS
011570  000410                        BR      3$              ;;
011572  005737  001422        1$:     TST     WFTEST
011576  001003                        BNE     2$
011600  012702  011656                MOV     #VARLT1,R2              ;WFTEST=0,USE NORMAL LIMITS
011604  000402                        BR      3$
011606  012702  011666        2$:     MOV     #VARLT2,R2              ;WFTEST=1,USE OPTION AREA LIMITS
011612  012221                3$:     MOV     (R2)+,(R1)+
011614  005711                        TST     (R1)
011616  100375                        BPL     3$
011620  000207                        RTS     PC
```

```
                                V1:     1                       ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
                                V2:     2
                                V10:    10
                                V50:    50
                                V144:   144
                                V115:   115
                                V240:   240
                                V5:     5
                                V500:   50.

                                VNR:    0                       ;RMS NOISE LIMIT
                                VNP:    0                       ;PEAK NOISE LIMIT
                                VSET:   0                       ;INTER-CHANNEL SETTLING LIMIT
                                VLIN:   0                       ;RELATIVE ACCURACY ERROR LIMIT
                                        BIT15

                                VARLT1: 25.             ;.25 LSB,NORMAL LIMITS FOR SYSTEM
                                        200.            ;2. LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
                                        100.            ;1 LSB
                                        100.            ;1 LSB

                                VARLT2: 23.             ;.23 LSB,  TIGHTER LIMITS FOR OPTION
                                        150.            ;1.5 LSB,  AREA USE ON SPEC TESTS
                                        90.             ;.9 LSB
                                        90.             ;.9 LSB

                                VARLT3: 50.             ;.5 LSB,  LIMITS FOR AM11K TESTING
                                        200.            ;2. LSB
                                        150.            ;1.5 LSB
                                        150.            ;1.5 LSB

011706  052777  000100  167230  AGATST: BIS     #100,@#STKS
011714  000177  000000          JMP     @AGTST
011720  001620                  AGTST:  BEGIN
```

```
                                .SBTTL  END OF PASS ROUTINE

                                ;;**************************************************** ************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS"
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO AGATST
                                ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
                                ;*SENDMG CAN BE CHANGED TO 7.

011722                          $EOP:
011722  000240                          NOP
011724  005037  001102                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011730  005037  001160                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
011734  005237  001202                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
011740  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011746  005327                          DEC     (PC)+           ;;LOOP?
011750  000001                  $EOPCT: .WORD   1
011752  003015                          BGT     $DOAGN          ;;YES
011754  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011756  000001                  $ENDCT: .WORD   1
011760  011750                          $EOPCT
011762  104400  012015                  TYPE    ,$ENDMG         ;;TYPE "END PASS"
011766  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011772  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011774  000005                          RESET                   ;;CLEAR THE WORLD
011776  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
012000  000240                          NOP                     ;;SAVE ROOM
012002  000240                          NOP                     ;;FOR
012004  000240                          NOP                     ;;ACT11
012006                          $DOAGN:
012006  000137                          JMP     @(PC)+          ;;RETURN
012010  011706                  $RTNAD: .WORD   AGATST
012012     377        377      000     $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
012015     015  042412  042116  000123  $ENDMG: .ASCIZ  <15><12>/END PASS/
012022  050040  051501  000123
                                .EVEN
```

```
                                                .SBTTL          ASCII MESSAGES
012030  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST--
012036  020105  042524  052123
012044  026455  000040
012050  005015  042523  052124  SETMSG: .ASCIZ  <15><12>/SETTLING TEST-- TYPE DESIRED 'FROM' CHANNEL & CR: /
012056  044514  043516  052040
012064  051505  026524  020055
012072  054524  042520  042040
012100  051505  051111  042105
012106  023440  051106  046517
012114  020047  044103  047101
012122  042516  020114  020046
012130  051103  020072     000
012135     055     000          MINUS:  .BYTE   55,0
012137     077     000          QUEST:  .BYTE   77,0
012141     136     101     040  AMSG:   .BYTE   136,101,40,40,0
012144     040     000
012146     136     103     040  CMSG:   .BYTE   136,103,40,40,0
012151     040     000
012153     136     107     015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
012156     012     123     127
012161     122     105     107
012164     072     000
012166  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
012174     000
012175     055  020055     000  DASH:   .ASCIZ  /-- /
012201     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
012206  026455  053440  042111
012214  044124  005015     000
012221     103  000110          CH:     .ASCIZ  /CH/
012224  020040  020040     000  SPACE:  .ASCIZ  /   /
012231     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
012236  047117  041440  000110
012244  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
012252  047111  020107  051106
012260  046517  041440  000110
012266  040440  020124     000  ATMSG:  .ASCIZ  / AT /
012273     116  044517  042523  NOI:    .ASCIZ  /NOISE: /
012300  020072     000
012303     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
012310  040510  047116  046105
012316  000040
012320  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
012326  042516  005015     000
012333     057     000          SLASH:  .ASCIZ  #/#
012335     124  050131  020105  TOMSG:  .ASCIZ  /TYPE DESIRED 'TO' CHANNEL & CR: /
012342  042504  044523  042522
012350  020104  052047  023517
012356  041440  040510  047116
012364  046105  023040  041440
012372  035122  000040
012376  020040  020040  045517  OKMSG:  .ASCIZ  /    OK/<15><12>
012404  005015     000
```

```
2290  012407     040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
2291  012414  047522  025122  006452
2292  012422  000012
2293  012424  051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
2294  012432  042105  051440  040524
2295  012440  042524  051450  000051
2296  012446  047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
2297  012454  020127  026050  030440
2298  012462  031057  046040  041123
2299  012470  020051  052123  052101
2300  012476  024105  024523  005015
2301  012504     000
2302  012505     040  044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
2303  012512  024040  020076  020061
2304  012520  027461  020062  051514
2305  012526  024502  051440  040524
2306  012534  042524  051450  006451
2307  012542  000012
2308  012544  051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
2309  012552  051450  020051  044527
2310  012560  042504  020122  044124
2311  012566  047101  031040  046040
2312  012574  041123     000
2313  012577     040  052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
2314  012604  026505  044527  052104
2315  012612  024110  024523  047440
2316  012620  052125  044523  042504
2317  012626  025440  047440  020122
2318  012634  020055  027461  020062
2319  012642  051514  000102
2320  012646  046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
2321  012654  051515  020054     000
2322  012661     040  051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
2323  012666  042520  045501  040440
2324  012674  020124     000
2325  012677     015  042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
2326  012704  047440  020106  047514
2327  012712  044507  020103  042524
2328  012720  052123     123
2329  012723     040  047117  040440  CNAD:   .ASCIZ  / ON AD11K AT /
2330  012730  030504  045461  040440
2331  012736  020124     000
2332  012741     040  042101  030461  MSGSC:  .ASCIZ  / AD11K'S FOUND/<15><12>
2333  012746  023513  020123  047506
2334  012754  047125  006504  000012
2335  012762  005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
2336  012770  020062  051514  006502
2337  012776  005012  005012  005012
2338  013004  005012  005012  005012
2339  013012  030455  031057  051514          .ASCIZ  \-1/2LSB\
2340  013020  000102
2341
```

```
2342                                                    .EVEN
2343  013022  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
2344  013030  047105  044524  046101
2345  013036  046040  047111  040505
2346  013044  044522  054524  006472
2347  013052  000012
2348  013054  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
2349  013062  020040  020040  020040
2350  013070  020040  020040  020040
2351  013076  020040  052123  052101
2352  013104  026505  044527  052104
2353  013112  020110  044504  052123
2354  013120  044522  052502  044524
2355  013126  047117  005015  005012
2356  013134  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><
2357  013142  051440  040524  042524
2358  013150  005123  005012  005012
2359  013156  005012  005012  005012
2360  013164  005012  005012  005012
2361  013172  005012
2362  013174  020040  020040  020040          .ASCII  /                                                  STATE WIDTH (LSB)/<15>
2363  013202  020040  020040  020040
2364  013210  020040  020040  020040
2365  013216  020040  020040  020040
2366  013224  020040  020040  020040
2367  013232  020040  020040  020040
2368  013240  020040  020040  020040
2369  013246  020040  020040  020040
2370  013254  051440  040524  042524
2371  013262  053440  042111  044124
2372  013270  024040  051514  024502
2373  013276  005015
2374  013300  030040  020040  020040          .ASCIZ  # 0              1/2              1              1 1/2              2#
2375  013306  020040  020040  020040
2376  013314  020040  020040  027461
2377  013322  020062  020040  020040
2378  013330  020040  020040  020040
2379  013336  020040  020061  020040
2380  013344  020040  020040  020040
2381  013352  020040  030440  030440
2382  013360  031057  020040  020040
2383  013366  020040  020040  020040
2384  013374  020040  031040      000
2385  013401      015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR DESIRED TEST: /
2386  013406  020105  042514  052124
2387  013414  051105  023040  041440
2388  013422  020122  047506  020122
2389  013430  042504  044523  042522
2390  013436  020104  042524  052123
2391  013444  020072      000
2392  013447      122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
2393  013454  053111  020105  041501
2394  013462  052503  040522  054503
2395  013470  006472  000012
```

```
2396  013474  046040  041123  046440  LINEA:  .ASCIZ  . LSB MAXIMUM AT /
2397  013502  054101  046511  046525
2398  013510  040440  020124     000
2399  013515     015  041412  046101  HEADS:  .ASCII  <15><12>/CALIBRATION--/
2400  013522  041111  040522  044524
2401  013530  047117  026455
2402  013534  051440  052105  041440  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
2403  013542  040510  047116  046105
2404  013550  044440  020116  053523
2405  013556  020122  047514  020127
2406  013564  054502  042524  005015
2407  013572     000
2408  013573     033  000132          CO:     .ASCIZ  <33><132>
2409  013576  000055                  C1:     .ASCIZ  <55>
2410  013600  031033     000          C2:     .ASCIZ  <33><62>
2411  013603     112     000          C3:     .ASCIZ  <112>
2412  013605     015  047412  043106  MOFSET: .ASCIZ  <15><12>/OFFSET =/
2413  013612  042523  020124  000075
2414  013620  046040  041123  000040  MLSB:   .ASCIZ  / LSB /
2415  013626  040440  020124     000  MAT:    .ASCIZ  / AT /
2416  013633     015  020012  047105  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
2417  013640  042524  044522  043516
2418  013646  052040  051505  020124
2419  013654     000
2420  013655     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
2421  013660     061     111     062
2422  013663     114     041     063
2423  013666     045     063     051
2424  013671     066     055     071
2425  013674     061     074     110
2426  013677     041     040     112
2427  013702     000
2428  013703     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
2429  013706     047     111     061
2430  013711     104     050     065
2431  013714     044     062     110
2432  013717     040     040     102
2433  013722     000
2434  013723     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0
2435  013726     112     033     061
2436  013731     101     040     033
2437  013734     062     000
2438  013736  005015  046412  026504  HEAD1:  .ASCII  <15><12><12>/MD-11-DZADL-B    AD11K DIAGNOSTIC/<15><12>
2439  013744  030461  042055  040532
2440  013752  046104  041055  020040
2441  013760  020040  042101  030461
2442  013766  020113  044504  043501
2443  013774  047516  052123  041511
2444  014002  005015
2445  014004  040412  020072  052501          .ASCII  <12>/A: AUTO TEST/
2446  014012  047524  052040  051505
2447  014020     124
2448  014021     015  041412  020072          .ASCII  <15><12>/C: CALIBRATION/
2449  014026  040503  044514  051102
```

```
  2450  014034  052101  047511     116
  2451  014041     015  046012  020072          .ASCII  <15><12>/L: LOGIC TEST/
  2452  014046  047514  044507  020103
  2453  014054  042524  052123
  2454  014060  005015  035116  047040          .ASCII  <15><12>/N: NOISE TEST/
  2455  014066  044517  042523  052040
  2456  014074  051505     124
```

```
2457  014077  000015  051412  020072          .ASCII  <15><12>/S: SETTLE TEST/
2458  014104  042523  052124  042514
2459  014112  052040  051505  000124
2460  014117  000015  053412  020072          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
2461  014124  051127  050101  051101
2462  014132  052517  042116  052040
2463  014140  051505  006524  000012
2464  014146  005015  052123  052101  EM1:    .ASCIZ  <15><12>/STATUS REG. ERROR/<15><12>
2465  014154  051525  051040  043505
2466  014162  020056  051105  047522
2467  014170  006522  000012
2468  014174  005015  040506  046111  EM2:    .ASCIZ  <15><12>/FAILED TO INTERRUPT/<15><12>
2469  014202  042105  052040  020117
2470  014210  047111  042524  051122
2471  014216  050125  006524  000012
2472  014224  005015  047125  054105  EM3:    .ASCIZ  <15><12>/UNEXPECTED INTERRUPT/<15><12>
2473  014232  042520  052103  042105
2474  014240  044440  052116  051105
2475  014246  052522  052120  005015
2476  014254  000000
2477  014255  000015  042412  051122  EM4:    .ASCIZ  <15><12>#ERROR ON A/D CHANNEL#<15><12>
2478  014262  051117  047440  020116
2479  014270  027501  020104  044103
2480  014276  047101  042516  006514
2481  014304  000012
2482  014306  051105  050122  020103  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/<15><12>
2483  014314  052123  042522  020107
2484  014322  054105  042520  052103
2485  014330  042105  040440  052103
2486  014336  040525  006514  000012
2487  014344  051105  050122  020103  DH2:    .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
2488  014352  051440  051124  043505
2489  014360  020040  041440  040510
2490  014366  047116  046105  020040
2491  014374  047516  044515  040516
2492  014402  020114  052040  046117
2493  014410  051105  047101  042503
2494  014416  020040  041501  052524
2495  014424  046101  000000
2496  014427  000105  051122  041520  DH3:    .ASCIZ  /ERRPC      STREG    ACTUAL/<15><12>
2497  014434  020040  020040  020040
2498  014442  052123  042522  020107
2499  014450  020040  040440  052103
2500  014456  040525  006514  000012
```

```
2501  014464  000                       HUNS:   .BYTE   0
2502  014465  056                       DECPNT: .BYTE   56
2503  014466  000                       TENS:   .BYTE   0
2504  014467  000     000               ONES:   .BYTE   0,0
2505          014472                    .EVEN
2506
2507  014472  001116  001316  001124   DT1:    $ERRPC, STREG, $GDDAT, $BDDAT,0
2508  014500  001126  000000
2509  014504  001116  001316  001366   DT2:    $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
2510  014512  001124  001406  001126
2511  014520  000000
2512  014522  001116  001316  001126   DT3:    $ERRPC,STREG,$BDDAT,0
2513  014530  000000
2514
2515  014532  000000                   DF1:    0
2516
2517
2518
2519
```

```
                                .SBTTL  TTY INPUT ROUTINE

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

                                .DSABL  LSB


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

014534  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014536  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
014544  105777  164374          1$:     TSTB    @$TKS           ;;WAIT FOR
014550  100375                          BPL     1$              ;;A CHARACTER
014552  117766  164370  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
014560  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
014566  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014574  001013                          BNE     3$              ;;BRANCH IF NO
014576  105777  164342          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014602  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
014604  117746  164336                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014610  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
014614  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
014620  001366                          BNE     2$              ;;IF NOT DISCARD IT
014622  000750                          BR      1$              ;;YES, RESUME
014624  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
014632  002407                          BLT     4$              ;;BRANCH IF YES
014634  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
014642  003003                          BGT     4$              ;;BRANCH IF YES
014644  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
014652  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;******************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

014654  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
014656  012703  014762          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
014662  022703  014772          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
014666  101405                          BLOS    4$              ;;BR IF YES
014670  104404                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
014672  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
014674  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
014700  001003                          BNE     3$              ;;SKIP IF NOT
014702  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
014706  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
```

```
2574  014710  111337  014760          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2575  014714  104400  014760                  TYPE    ,9$
2576  014720  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2577  014724  001356                          BNE     2$              ;;LOOP IF NOT RETURN
2578  014726  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2579  014732  104400  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
2580  014736  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2581  014740  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2582  014742  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2583  014750  012766  014762  000004          MOV     #$TTYIN,4(SP)
2584  014756  000002                          RTI                     ;;RETURN
2585  014760     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2586  014761     000                          .BYTE   0               ;;TERMINATOR
2587  014762  000010               $TTYIN:    .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2588  014772  052536  005015     000 $CNTLU: .ASCIZ   /↑U/<15><12>    ;;CONTROL "U"
2589  014777     136  006507  000012 $CNTLG: .ASCIZ   /↑G/<15><12>    ;;CONTROL "G"
2590  015004  005015  053523  020122 $MSWR:  .ASCIZ   <15><12>/SWR = /
2591  015012  020075     000
2592  015015     040  047040  053505 $MNEW:  .ASCIZ   /  NEW = /
2593  015022  036440  000040
```

```
                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*CALL:
                                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
015026  011646                 $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
015030  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
015036  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015040  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015042  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015044  104405                 1$:     RDLIN                   ;;READ AN ASCIZ LINE
015046  012600                         MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
015050  005001                         CLR     R1              ;;CLEAR DATA WORD
015052  005002                         CLR     R2
015054  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
015056  001412                         BEQ     3$              ;;IF ZERO GET OUT
015060  006301                         ASL     R1              ;;*2
015062  006102                         ROL     R2
015064  006301                         ASL     R1              ;;*4
015066  006102                         ROL     R2
015070  006301                         ASL     R1              ;;*8
015072  006102                         ROL     R2
015074  042716  177770                 BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
015100  062601                         ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
015102  000764                         BR      2$              ;;LOOP
015104  005726                 3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
015106  010166  000012                 MOV     R1,12(SP)       ;;SAVE THE RESULT
015112  010237  015126                 MOV     R2,$HIOCT
015116  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
015120  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
015122  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
015124  000002                         RTI                     ;;RETURN
015126  000000                 $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
                                        .SBTTL  SCOPE HANDLER ROUTINE

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

015130                                  $SCOPE:
015130  032777  040000  164002  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
015136  001114                          BNE     $OVER           ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
015140  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
015142  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015146  012737  015160  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
015154  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
015160  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015164  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
015166  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
015170  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015174  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
015176                          6$:;#####END OF CODE FOR THE XOR TESTER#####
015176  032777  000400  163734          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
015204  001404                          BEQ     2$              ;;BR IF NO
015206  127737  163726  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR 7:0
015214  001465                          BEQ     $OVER           ;;BR IF YES
015216  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
015222  001421                          BEQ     3$              ;;BR IF NO
015224  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015232  101015                          BHI     3$              ;;BR IF NO
015234  032777  001000  163676          BIT     #B 709,@SWR     ;;LOOP ON ERROR?
015242  001404                          BEQ     4$              ;;BR IF NO
015244  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015252  000446                          BR      $OVER
015254  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
015260  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015264  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015266  032777  004000  163644  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015274  001011                          BNE     1$              ;;BR IF YES
015276  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015302  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015304  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015310  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015316  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015320  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015326  013737  015404  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015334  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
015340  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
015346  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
015352  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
015356  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
015362  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
015370  013777  001102  163544  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
015376  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
015402  000002                          RTI                     ;;FIXES PS
015404  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
                                .SBTTL  ERROR HANDLER ROUTINE

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

015406                  $ERROR:
015406  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
015412  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015414  013777  001102  163520          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015422  032777  002000  163510          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015430  001402                          BEQ     1$              ;;NO - SKIP
015432  104400  001164                  TYPE    ,$BELL          ;;RING BELL
015436  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015442  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015446  162737  000002  001116          SUB     #2,$ERRPC
015454  117737  163436  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015462  032777  020000  163450          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015470  001004                          BNE     20$             ;;SKIP TYPEOUTS
015472  004737  015602                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
015476  104400  001171                  TYPE    ,$CRLF
015502                          20$:
015502  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015510  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
015512  113737  001114  015524          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
015520  004737  016236                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
015524  000                     21$:    .BYTE   0
015525  000                             .BYTE   0
015526  000777                  22$:    BR      22$             ;;APT ERROR LOOP
015530  005777  163404          2$:     TST     @SWR            ;;HALT ON ERROR
015534  100001                          BPL     3$              ;;SKIP IF CONTINUE
015536  000000                          HALT                    ;;HALT ON ERROR!
015540  032777  001000  163372  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
015546  001402                          BEQ     4$              ;;BR IF NO
015550  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
015554  005737  001162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
015560  001402                          BEQ     5$              ;;BR IF NONE
```

```
015562  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
015566                          5$:
015566  022737  011776  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
015574  001001                          BNE     6$              ;;BRANCH IF NO
015576  000000                          HALT                    ;;YES
015600                          6$:
015600  000002                          RTI                     ;;RETURN
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

015602                          $ERRTYP:
015602  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
015606  010046                          MOV     R0,-(SP)        ;;SAVE R0
015610  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
015612  153700  001114                  BISB    @#$ITEMB,R0
015616  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
015620  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
015624  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
015626  000426                          BR      6$              ;;GET OUT
015630  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
015632  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
015634  006300                          ASL     R0
015636  006300                          ASL     R0
015640  062700  001256                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
015644  012037  015654                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
015650  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
015652  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
015654  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
015656  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
015662  012037  015672          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
015666  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
015670  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
015672  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
015674  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
015700  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
015702  001004                          BNE     7$              ;;GO TYPE THE DATA
015704  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
015706  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
015712  000207                          RTS     PC              ;;RETURN
015714                          7$:
015714  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
015716  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
015720  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
015722  001770                          BEQ     6$              ;;BR IF NO
015724  104400  015732                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
015730  000771                          BR      7$              ;;LOOP
015732  020040      000         8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                    015736              .EVEN
```

```
                                        .SBTTL  TYPE ROUTINE

                                        ;;**********************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*       TYPE    .MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*       TYPE
                                        ;*       MESADR
                                        ;*

015736  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
015742  100002                          BPL     1$              ;;BR IF YES
015744  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
015746  000430                          BR      3$              ;;LEAVE
015750  013046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
015752  017600  000002                  MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
015756  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015764  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
015766  132737  000100  001215          BITB    #APTSPCOL,$ENVM ;;SPOOL MESSAGE TO APT
015774  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
015776  010037  016006                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
016002  004737  016226                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
016006  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
016010  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
016016  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
016020  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016022  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
016024  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
016026  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
016030  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
016034  000002                          RTI                     ;;RETURN
016036  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
016042  001430                          BEQ     8$
016044  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
016050  001006                          BNE     5$
016052  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
016054  104400                          TYPE                    ;;TYPE A CR AND LF
016056  001171                          $CRLF
016060  105037  016214                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
016064  000755                          BR      2$              ;;GET NEXT CHARACTER
016066  004737  016150          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
016072  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
016076  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
016100  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
016104  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
016110  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
                                        JSR     PC,$TYPEC       ;;GO TYPE A NULL
                                        DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
                                        BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

                                8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
                                9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
                                        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
                                        BNE     9$              ;;TAB STOP
                                        TST     (SP)+           ;;POP SPACE OFF STACK
                                        BR      2$              ;;GET NEXT CHARACTER
                                $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
                                        BPL     $TYPEC
                                        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
                                        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
                                        BNE     1$              ;;BRANCH IF NO
                                        CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
                                        BR      $TYPEX          ;;EXIT
                                1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
                                        BEQ     $TYPEX          ;;BRANCH IF YES
                                        INCB    (PC)+           ;;COUNT THE CHARACTER
                                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
                                $TYPEX: RTS     PC

                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;**************************************************************
                                $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
                                $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
                                        BR      $ATYC
                                $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
                                $ATYC:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
                                        BEQ     5$              ;;IF NOT:  BR
                                        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
                                        BNE     3$              ;;IF NOT:  BR
                                        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
                                        BEQ     3$              ;;IF NOT:  BR
                                        MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
                                        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
                                1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
                                        BNE     1$              ;;IF NOT:  WAIT
                                        MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
                                2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
                                        BNE     2$
                                        SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
                                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
                                        MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
                                        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
                                        BR      5$
                                3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
```

```
 2902  016356  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 2903  016364  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 2904  016370  004737  015736                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
 2905  016374  000000                  4$:     .WORD   0
 2906  016376                          5$:
 2907  016376  105737  016464          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 2908  016402  001416                          BEQ     12$             ;;IF NOT:  BR
 2909  016404  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
 2910  016410  001413                          BEQ     12$             ;;IF NOT:  BR
 2911  016412  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 2912  016416  001375                          BNE     11$             ;;IF NOT:  WAIT
 2913  016420  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
 2914  016426  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 2915  016434  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 2916  016440  105037  016464          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 2917  016444  105037  016463                  CLRB    $LFLG           ;;CLEAR LOG FLAG
 2918  016450  105037  016462                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 2919  016454  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 2920  016456  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 2921  016460  000207                          RTS     PC              ;;RETURN
 2922  016462     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
 2923  016463     000          $LFLG:  .BYTE   0               ;;LOG FLAG
 2924  016464     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
 2925          016466                  .EVEN
 2926          000200          APTSIZE=200
 2927          000001          APTENV=001
 2928          000100          APTSPOOL=100
 2929          000040          APTCSUP=040
```

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

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

016466  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016472  116637  000001  016711          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016500  112637  016713                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016504  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016510  000406                          BR      $TYPON
016512  112737  000001  016711  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016520  112737  000006  016713          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
016526  112737  000005  016710  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016534  010346                          MOV     R3,-(SP)        ;;SAVE R3
016536  010446                          MOV     R4,-(SP)        ;;SAVE R4
016540  010546                          MOV     R5,-(SP)        ;;SAVE R5
016542  113704  016713                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
016546  005404                          NEG     R4
016550  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
016554  110437  016712                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
016560  113704  016711                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
016564  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
016570  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
016572  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
016574  000404                          BR      3$              ;;GO DO MSB
016576  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
016600  006105                          ROL     R5
016602  006105                          ROL     R5
016604  010503                          MOV     R5,R3
016606  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
016610  105337  016712                  DECB    $OMODE          ;;TYPE THIS DIGIT?
016614  100016                          BPL     7$              ;;BR IF NO
016616  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
016622  001002                          BNE     4$              ;;"ES" FOR 0
```

```
 2984  016624  005704                          TST     R4              ;;SUPPRESS THIS 0?
 2985  016626  001403                          BEQ     5$              ;;BR IF YES
 2986  016630  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 2987  016632  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 2988  016636  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 2989  016642  110337  016706                  MOVB    R3,8$           ;;SAVE FOR TYPING
 2990  016646  104400  016706                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
 2991  016652  105337  016710          7$:     DECB    $OCNT           ;;COUNT BY 1
 2992  016656  003347                          BGT     2$              ;;BR IF MORE TO DO
 2993  016660  002402                          BLT     6$              ;;BR IF DONE
 2994  016662  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 2995  016664  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 2996  016666  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 2997  016670  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 2998  016672  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 2999  016674  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3000  016702  012616                          MOV     (SP)+,(SP)
 3001  016704  000002                          RTI                     ;;RETURN
 3002  016706     000          8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
 3003  016707     000                  .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
 3004  016710     000          $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
 3005  016711     000          $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
 3006  016712  000000          $CMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  TRAP DECODER

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

016714  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
016716  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
016722  005740                          TST     -(R0)           ;;BACKUP BY 2
016724  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
016726  006300                          ASL     R0              ;;POSITION FOR INDEXING
016730  016000  016736                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
016734  000200                          RTS     R0              ;;GO TO ROUTINE

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
016736                          $TRPAD:
016736  015736                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
016740  016512                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
016742  016466                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
016744  016526                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)

016746  014534                          $RDCHR  ;;CALL=RDCHR    TRAP+4(104404)  TTY TYPEIN CHARACTER ROUTINE
016750  014654                          $RDLIN  ;;CALL=RDLIN    TRAP+5(104405)  TTY TYPEIN STRING ROUTINE
016752  015026                          $RDOCT  ;;CALL=RDOCT    TRAP+6(104406)  READ AN OCTAL NUMBER FROM TTY
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
016754  012737  017120  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
016762  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
016770  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016772  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016774  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016776  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
017000  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
017002  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
017004  017746  162130                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
017010  010637  017124                  MOV     SP,$SAVR6       ;;SAVE SP
017014  012737  017026  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
017022  000000                          HALT
017024  000776                          BR      .-2             ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
017026  012737  017120  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
017034  013706  017124                  MOV     $SAVR6,SP       ;;GET SP
017040  005037  017124                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
017044  005237  017124          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
017050  001375                          BNE     1$              ;;OF  WORD
017052  012677  162062                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
017056  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
017060  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
017062  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
017064  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
017066  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
017070  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
017072  012737  016754  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
017100  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
017106  104400                          TYPE                    ;REPORT THE POWER FAILURE
017110  017126                  $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
017112  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEG2
017114  002256                  $PWRAD: .WORD   BEG2             ;;RESTART ADDRESS
017116  000002                          RTI
017120  000000                  $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
017122  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
017124  000000                  $SAVR6: 0                        ;;PUT THE SP HERE
017126  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
017134  000122
                                        .EVEN
                                .EVEN
017136  000310                  DIST:   .BLKW   200.            ;STATE-WIDTH DISTRIBUTION
017756  010000                  BUFFER: .BLKW   4096.           ;BUFFER AREA

000001                          .END
```

```
ABASE = 170400           522*   619    660    710    711    712
ACDW1 = 000000           619    662
ACDW2 = 000000           619
ACPUOP= 000000           619    634
ADBUFF  001322           712*   998   1021   1075   1098   1116   1133   1176*  1182   1250*  1261*  1406   1464*
                        1473*  1475*  1549*  1600*  1613   1653*  1662   1668   1714   2042
ADDW0 = 000000           619
ADDW1 = 000000           619
ADDW10= 000000           619
ADDW11= 000000           619
ADDW12= 000000           619
ADDW13= 000000           619
ADDW14= 000000           619
ADDW15= 000000           619
ADDW2 = 000000           619
ADDW3 = 000000           619
ADDW4 = 000000           619
ADDW5 = 000000           619
ADDW6 = 000000           619
ADDW7 = 000000           619
ADDW8 = 000000           619
ADDW9 = 000000           619
ADEVCT= 000000           619    625
ADEVM = 000000           619    661
ADSTI   001320           711*  1463*  1472*  1474*  1658*
AENV  = 000000           619    630
AENVM = 000000           619    631
AFATAL= 000000           619    622
AGAIN   007052          1705*  1734
AGATST  011706          2197*  2232
AGTST   011720          1430*  1447*  1457*  2198   2199*
AMADR1= 000000           619    647
AMADR2= 000000           619    651
AMADR3= 000000           619    654
AMADR4= 000000           619    657
AMAMS1= 000000           619    641
AMAMS2= 000000           619    649
AMAMS3= 000000           619    652
AMAMS4= 000000           619    655
AMSG    012141           767   2252*
AMSGAD= 000000           619    627
AMSGLG= 000000           619    628
AMSGTY= 000000           619    621
AMTYP1= 000000           619    642
AMTYP2= 000000           619    650
AMTYP3= 000000           619    653
AMTYP4= 000000           619    656
APASS = 000000           619    624
APRIOR= 000300           524*   619    714
APTCSU= 000040          2824   2929*
APTENV= 000001          2725   2817   2885   2927*
APTSIZ= 000200           832   2926*
APTSPC= 000100          2819   2887   2928*
ARCUND  007174          1730   1733*
```

```
ASKCH   013534          1504    2402#
ASWREG= 000000           619     632
ATESTN= 000000           619     623
ATMSG   012266          1546    2273#
AUNIT = 000000           619     626
AUSWR = 000000           619     633
AVECT1= 140340           523#    619     658     713     715
AVECT2= 000000           619     659
BASEBR  001326           714#   1057    1068    1478*   1479*
BASECH  001336           718#   1171    1320    1321    1548    1579    1580    1699    1996    2030    2153
BEGIN   001620           536     789#   2199
BEGINA  005272           678    1434#
BEGINC  005050           881    1395#
BEGIND  002562           926#   1427    1437
BEGINM  005660           887    1502#
BEGINS  005730           890    1512#
BEGINW  005360           693    1451#
BEGIN2  001626           538     791#
BEGL    005234           869     884    1425#
BEGL2   002256           537     764     865#   3077
BITPNT  001416           742#   1649*   1652    1675    1676*
BIT0  = 000001           485#   1019    1036    1060    1069    1089    1105    1109    1123    1124
BIT00 = 000001           485#    495
BIT01 = 000002           484#    494
BIT02 = 000004           483#    493
BIT03 = 000010           482#    492
BIT04 = 000020           481#    491
BIT05 = 000040           480#    490
BIT06 = 000100           479#    489
BIT07 = 000200           478#    488
BIT08 = 000400           477#    487    2661
BIT09 = 001000           476#    486    2669    2735
BIT1  = 000002           494#
BIT10 = 002000           475#   2713
BIT11 = 004000           474#   2676
BIT12 = 010000           473#   1931
BIT13 = 020000           472#   1391    1407    2720
BIT14 = 040000           471#    936    2647
BIT15 = 100000           470#    967     995    1108    1121    2180
BIT2  = 000004           493#
BIT3  = 000010           492#
BIT4  = 000020           491#    960    1173    1175
BIT5  = 000040           490#    952
BIT6  = 000100           489#    944    1036    1040    1060    1069    1087    1089    1401
BIT7  = 000200           488#    992    1040    1108    1175
BIT8  = 000400           487#    932
BIT9  = 001000           486#    990    1110    1174
BK3     005762          1519#   1530
BK3VEC= 000014           502#
BUFFER  017756          1682    1719#   1721#   1736    1764    1833    1845    1947    1875    1886    1887    1889    1910
                        1913    3087#
BUFF1   013655          1816    2420#
BUFF2   013703          1909    2428#
BUMPAD  005422          1428    1445    1455    1460#
```

```
BYPASS  005614            1469    1485#
B10     011336            2105#   2106
B11     011372            2115#   2116
CH      012221            1393    2266#
CHAN    012303            1964    2276#
CHANL   001366             730#   1320*  1520*  1601  1663  1937*  1938  1965  1995*  1996*  1997  2031*  2509
CHANNL  007016            1698#
CH1     001354             725#   1505*  1506*  1509  1515*  1541  1547  1571  1577*  1579*
CH2     001356             726#   1518*  1520   1533  1567   1578*  1580*
CLEAR1  006744            1684#   1686
CLEAR2  007010            1695#   1697
CMSG    012146             761    2254#
COMPAR  011116            1158    1193   1204   1216  1229  1241  1255  1277  1289  1300  1311  1337  2052#
CONV    006410            1611#   1615
CONVR   007100            1709    1712#
CONVRT  011004            1156    1191   1202   1214  1227  1239  1252  1263  1287  1298  1309  1550  2029#
CR    = 000015             410#   2863   2873
CRLF  = 000200             411#   2834   2873
CC      013573             841    2408#
C1      013576            1632    2409#
C2      013600            1825    1915   2410#
C3      013603            1912    2411#
DAC     001410             739#   1325   1328   1570  1574  1650*  1652*  1653  1675*  1983  1986  1989  1992
DASH    012175            1771    2262#
DAWAIT  005040            1251    1262   1296   1380#
DCISP = 177570             417#    600    820
DECPNT  014465            1776*   1805*  2502#
DECTYP  011406            1335    1531   1772   1778  1786  1790  1793  1804  1854  1958  1961  2123#
DECS    003612            1125#   1126
DELAY   001412             740#   1654*  1655*
DELAY1  007142            1715    1723#
DELAY2  007150            1717    1724   1725#
DELAY3  007074            1710#   1711
DELCLR  010254            1814    1826   1916   1925#
DF1     014532             687     694    700    706   2515#
DH1     014306             685    2483#
DH2     014344             704    2487#
DH3     014427             692     698   2496#
DIFLIN  006724            1377    1680#
DISPLA  001142             600#    820*   828*  1409*  2691*  2712*
DISPRE  000174             533#    828
DIST    017136            1687    1747*  1817   3086#
DONE    012320            2087    2279#
DSPAD   177570             416#    599    819
DT1     014472             686    2507#
DT2     014522             705    2509#
DT3     014562             693     699   2512#
DUMC    011236            1042    1045   1076   2083#
DUMMY   001364             729#   1321*  1567*  1571*  1658  1938*  1997*
DUMW    011164            1033    1066   2065#
EDGE    001414             741#   1322*  1585*  1588*  1608*  1613*  1616*  1617*  1618*  1619*  1623  1668  2003*
                          2006#
EDGFLG  006514            1588*   1593*  1629   1639#  1941*  2011*
EMTVEC= 000030             505#    803*   804*
```

```
EM1     014146#   684  2464#
EM2     014174#   691  2468#
EM3     014224#   697  2472#
EM4     014255#   703  2477#
ER      010476#  1971  1973  1976#
ERMSG   012407#  1342  1561  1782  1797  1809  1870  1976  2290#
ERR     006162#  1558  1561#
ERRVEC= 000024#   498#   8:   818*  829*  897   898*  913*  2652  2653*  2655*  2658*
FIRST   001346#   722#  1693*  1759  1761*
FIXERR  005500#  1461  1470#
FIXONE  005524#   857  1471#
FLAG    001424#   737#   836*  853*  1812  1873
HALFDAY 010032#  1877#  1885
HALFEDG 006336#  1521  1581  1600#  1939  1998
HANG    012153#   775  2256#
HANGS   = ******  G   532  3031  3032  3033  3034  3037  3038  3039
HNGMSG  012577#  1806  2313#
HALF    007512#  1798  1800#
HERO1   013736#   855  2438#
HERO5   013515#  1387  2399#
HT      = 000011#   408#  2832  2873
HUNS    014464#  2132*  2144*  2146*  2149  2501#
INRNGE  007236#  1743  1746#
IOTVEC= 000020#   503#   801*  802*
ISERV   001454#   756#   859
KBVECT  001340#   719#   858  1487
LAST    007354#  1757  1774#
LEND    005036#  1376  1378#
LESS    011324#  2100  2102#
LF      = 000012#   409#  2867  2873
LINEA   013474#  1855  2396#
LORD    010232#  1911  1914  1918#
LORDY   011304#  1820  1822  1921  2096#
LORDO   010236#  1919#  1923
LO2     010230#  1874  1917#
LSB     012231#  1532  2268#
LSBMSG  012166#  1773  2260#
MAT     013626#  1538  2415#
MAX     001424#   745#  1259*  1265  1269*  1893*  1898  1900*  1904
MAXTST  010130#  1896  1898#
MEMO    012677#  1438  2325#
MESP    012661#  1962  2322#
MESR    012646#  1959  2320#
METST   013633#  2069  2416#
MIN     001420#   743#  1270*  1330  1892*  1895  1897*  1903
MINUS   012135#  2125  2250#
MLSB    013620#  1336  2414#
MOFSET  013605#  1333  2412#
MSG16   013054#  1815  2348#
MSG18   012762#  1908  2335#
MSG20   013022#  1680  2343#
MSG21   013447#  1832  2392#
MSG22   012741#   910  2332#
MSG71   013401#   870  2385#
```

```
NAR     007422  1783  1785#
NARMSG  012446  1787  2296#
NARROW  001344   721#  1692*  1751*  1785  1800
NBEXT   001360   727#   899*   901*   905   911*   912  1460  1468*  1483*
NEXT    007056  1706#  1729
NMBEXT  001362   728#   912*  1483
NO      012273  1950  2274#
NOIA    010524  1940  1979#
NOIMSG  012038  1503  2238#
NOIST   010318  1508  1937#
NOI     010514  1981#  2002
NOIB    010534  1351  1353  1355  1995#
NOME    001644   772   774   786#
NOMSG   007220  1755  1756#
NOMEM   007220  1842  1847#
NOP     007166  1727  1731#
NOVRFL  002202   844   847   852   855#   922
NOXCMP  010114  1894#  1902
NOXCVT  006512  1658#  1672
NOXSTA  007664  1834#  1848
NOTYI   007620  1819#  1824
NOXTB   010076  1890#  1907
OFFERR  004704  1340  1342#
OFFOK   004712  1341  1344#
OKAY    007160  1720  1722  1728#  1732
OKAYD   011432  2128  2130#
OKMSG   012376  1344  1559  1784  1799  1811  1872  1974  2288#
ONAC    012723  2075  2329#
ONES    011467  2130*  2136*  2137  2139*  2148*  2504#
OUT     001430   747#  1681*  1690*  1744*  1789  1792  1795  1802
OUTMSG  012544  1794  2308#
PC    =000007   429#   763*   769*   842*   845*   848*   857*   864*   914*   925*   933*   945*   953#
                961#   968#   996*  1010*  1022*  1033*  1042*  1045*  1056*  1076*  1096*  1113*  1136#
               1145*  1180*  1251*  1262*  1286*  1335*  1377*  1378*  1383*  1426*  1427*  1428*  1436#
               1437*  1444*  1445*  1453*  1454*  1455*  1495*  1521*  1523*  1524*  1529*  1531*  1539#
               1560*  1562*  1575*  1581*  1584*  1594*  1620*  1638*  1706*  1772*  1778*  1786*  1790#
               1793*  1804*  1814*  1820*  1822*  1826*  1854*  1911*  1914*  1916*  1917*  1921*  1924#
               1935*  1939*  1940*  1942*  1958*  1961*  1963*  1975*  1977*  1993*  1998*  2002*  2012#
               2027*  2081*  2088*  2093*  2118*  2150*  2165*  2216*  2219*  2226*  2231   2722*  2728#
               2784*  2822*  2841*  2848*  2855*  2869*  2871*  2904*  2921*  3076
PEAK    001402   736#  1954   1956*  1960   1972   1980*  1989*  1992*  2000*  2009*  2010*
PERCNT  001426   746#  1644*  1645*  1646*  1647*  1648*  1673
PIRQ  = 177772   415#
PIRQVE= 000240   509#
PLUS    007704  1839  1841#
PLUSR2  011312  2097  2099#
POPPC   001606   784#   787
POS     011420  2124  2127#
POSPER  010400  1955  1957#
POSPMS  010366  1952  1954#
POSR2   006020  1525  1527#
PR0   = 000000   432#
PR1   = 000040   433#
PR2   = 000100   434#
```

CROSS REFERENCE TABLE

```
PR3     = 000140            435*
PR4     = 000200            436*
PR5     = 000240            437*
PR6     = 000300            438*
PR7     = 000340            439*
PS      = 177776            412*    413
PSW     = 177776            413*    873*   1034*   1057*   1058*   1068*   1074*   1388*   2092*
PWRVEC= 000024              504*    807*    808*   3044*   3045*   3054*   3060*   3072*   3073*
QUEST   012137              786     894    2251*
RANDY   010722             1706    2018*
RBEG    001634              790     792*
RDCHR   = 104404           2568    3037*
RDLIN   = 104405            871    2609    3038*
RDOCT   = 104406            782    1514    1517    3039*
READ    007210             1737*   1775
RELACC  007650             1813    1830*
RESET   002206              838     856*
RESVEC= 000010              499*
RET     006512             1630    1638*
RETERR  003710             1143    1145*
RETURN  001610              785*   1399    1610    1659    1704    2038
RM5     001400              735*   1951    1953*   1957    1970    1979*   1983*   1986*   1999*   2007*   2008*
RM4     001372              732*    861*   1707    2018*   2019*   2020*   2021    2024
RM3     001374              733*    862*   2018    2021*   2022*   2023*   2025
RM2     001376              734*    863*   2019    2022    2024*   2025*   2026*
RM1     011266              763     769    2090*
R0      =000000             420*    756     757*    758*    759     765     771     784*    843     846     849     851     858*
                            859*    860*    874*    875*    876     879     882     885     888     891     916*    919*    923*
                            924*    990*    993*    998*   1021*   1032*   1055*   1075*   1110*   1111*   1116*   1122*   1125*
                           1133*   1174*   1178*   1182*   1330*   1331*   1332    1380*   1381*   1389*   1390*   1394    1400*
                           1401*   1402    1406*   1409    1412    1485*   1489    1491*   1492    1493    1496    1497    1601*
                           1602*   1603*   1604    1605*   1606*   1609*   1614*   1651*   1670*   1673    1682*   1684*   1687*
                           1695*   1698*   1699*   1700*   1701*   1702    1703*   1733*   1735*   1774*   1803*   1807    1817*
                           1819    1833*   1834    1837*   1844    1847    1875*   1877    1883*   1886*   1891    1894    1910*
                           1913*   1919    1920    1925*   1927*   2029*   2030*   2031    2032*   2034    2035*   2036*   2039*
                           2043*   2056*   2057*   2059*   2060    2070    2223*   2226    2606    2610*   2613    2629*   2756
                           2757*   2758*   2765*   2766*   2767*   2768*   2769*   2770    2775    2780*   2792*   2786    2788
                           2815    2816*   2821    2826    2829*   2881    2889*   2893    2994    2896*   2897*   2898    2920*
                           3015    3016*   3017    3018*   3019*   3020*   3021*   3046    3071*
R1      =000001             421*   1417*   1418*   1486*   1487    1489*   1490*   1491    1496    1497    1657*   1671*   1683*
                           1685*   1688*   1696*   1705*   1725*   1728*   1736*   1737    1763    1818*   1823*   1830*   1836*
                           1837    1838    1876*   1884*   1889*   1906*   1918*   1922*   1926*   1929*   2055*   2057    2152*
                           2162*   2163    2607    2611*   2615*   2617*   2619*   2622*   2625    2628*   2882    2919*   3047
                           3070*
R2      =000002             422*   1260*   1261    1270    1271    1281*   1282    1334*   1403*   1420*   1522*   1526*   1527
                           1570*   1574*   1582*   1589    1591*   1592    1707*   1708*   1710*   1714*   1716    1718*   1719*
                           1721*   1723    1737*   1738*   1739*   1740*   1741*   1742    1746*   1747*   1748*   1749    1752
                           1756    1777*   1785*   1788*   1789*   1792*   1800*   1801*   1802*   1803    1819*   1821*   1834*
                           1835*   1836    1853*   1877*   1878*   1879*   1880*   1881*   1882*   1883    1890*   1901*   1919*
                           1957*   1960*   2096    2098*   2099    2101*   2102    2103*   2104*   2107    2123    2126*   2127
                           2129*   2133    2135*   2155*   2159*   2161*   2162    2608    2612*   2616*   2618*   2620*   2626
                           2627*   3048    3069*
R2POS   006320              1590    1592*
R3      =000003             423*   1274*   1276    1583*   1586*   1623*   1624    1631*   1633    1683*   1726*   1763*   1764*
```

```
                    1765*  1766   1831*  1841   1843*  1849*  1850*  1851*  1852*  1853   1868   1891*  1892
                    1893   1894*  1895   1897   1898   1900   2102*  2108*  2109*  2110*  2111*  2112*  2113*
                    2114*  2117   2564   2565*  2566   2565*  2570   2574   2576   2576*  2580*  2963   2972*
                    2978*  2979*  2982*  2987*  2988*  2989   2998*  3049   3068*
R4      =:000004     424*  1254*  1275   1276*  1527*  1557   1592*  1662*  1663*  1664*  1665*  1666   1838*
                    1840*  1841   1843   1867*  1903*  1905   2964   2966*  2967*  2968*  2969   2970*  2984
                    2986*  2994*  2997*  3050   3067*
R5      =:000005     425*  1156*  1158*  1191*  1193*  1202*  1204*  1214*  1216*  1227*  1229*  1239*  1241*
                    1252*  1255*  1263*  1277*  1287*  1289*  1298*  1300*  1309*  1311*  1323*  1326*  1337*
                    1351*  1353*  1355*  1363*  1367*  1508*  1550*  1568*  1572*  1577   1578   1595*  1644
                    1678*  1844*  1945*  1846*  1856   1862*  1863   1888*  1904*  1905   1937   1943*  1981*
                    1984*  1987*  1990*  1995   2013*  2029   2049*  2052   2053   2062   2063*  2965   2971*
                    2973*  2975*  2976*  2977*  2978   2996*  3051   3066*
R6      =:000006     426*   428    795*   796*   797
R7      =:000007     427*   429
SARSJB  006516      1323   1326   1568   1572   1644*  1981   1984   1987   1990
SAR1    006542      1649*
SETAA   006270      1584*  1587
SETCH   012244      1540   2270*
SETMSG  012050      1513   2241*
SET1A   006170      1523   1524   1567*  1584
SET8    006232      1363   1367   1577*
SHIFT   006714      1674   1676*
SKIPST  001350       723*  1694*  1754*  1777   1780
SKPMSG  012424      1779   2293*
SLASH   012333       781   1861   2281*
SP      =:000006     428*   756*   762*   768*   776*   783    784    799*   817*   825*   829    866*   874
                     897*   904    905*   913    921    1044   1066   1078   1095   1144*  1394*  1412*  1439*
                    1470*  1515   1518   1533*  1541*  1552*  1624*  1633*  1766*  1856*  1863*  1965*  2070*
                    2076*  2536*  2537*  2540*  2541*  2542   2546*  2547*  2548   2551   2553   2555*  2564*
                    2569   2580   2581*  2582*  2583*  2604*  2605*  2606*  2607*  2608*  2610   2613*  2621*
                    2622   2624   2625*  2627   2628   2629   2652*  2655   2657   2658   2687   2688   2692*
                    2717   2737*  2740*  2756*  2761*  2782   2786*  2815*  2816   2826*  2828   2829   2830*
                    2832   2837   2836   2842   2844*  2846*  2854*  2858   2862   2863   2867   2881*  2882*
                    2889   2890*  2901   2902*  2903*  2913   2914*  2919   2920   2955*  2956   2957   2958*
                    2963*  2964*  2965*  2971   2996   2997   2998   2999*  3000*  3015*  3016   3046*  3047*
                    3048*  3049*  3050*  3051*  3052*  3053   3061*  3065   3066   3067   3068   3069   3070
                    3071   3076*
SPACE   012224      1411   2267*
SPREAD  001406       738*  2053*  2060   2509
STACK = 001100       403*   762    768    799    866
STATE   012201      1762   2263*
STKLMT= 177774       414*
STREG   001316       710*   976*   982    991*   995*   1006*  1007   1018*  1019   1036*  1037   1039   1060*
                    1061   1063   1069*  1070   1073*  1089*  1090   1092   1105*  1106   1109*  1123*  1124*
                    1127   1134*  1140*  1141   1173*  1183*  1386*  1402*  1404*  1439   1462*  1471*  1604*
                    1611*  1660*  1666*  1702*  1712*  2034*  2040*  2076   2507   2509   2512
SWDIST  007566      1810   1812*
SWR     001140       599*   773    776    783*   797    819*   821    827*   834*   1389   1391   1407   1505
                    1931   2647   2661   2663   2669   2676   2713   2720   2732   2735   3052   3065*
SWREG   000176       534*   827
SWC     = 000001     467*
SWDD    = 000001     457*   467
SWD1    = 000002     456*   466
```

```
SW02  = 000004          455#    465
SW03  = 000010          454#    464
SW04  = 000020          453#    463
SW05  = 000040          452#    462
SW06  = 000100          451#    461
SW07  = 000200          450#    460
SW08  = 000400          449#    459
SW09  = 001000          448#    458
SW1   = 000002          466#
SW10  = 002000          447#
SW11  = 004000          446#
SW12  = 010000          445#
SW13  = 020000          444#
SW14  = 040000          443#
SW15  = 100000          442#
SW2   = 000004          465#
SW3   = 000010          464#
SW4   = 000020          463#
SW5   = 000040          462#
SW6   = 000100          461#
SW7   = 000200          460#
SW8   = 000400          459#
SW9   = 001000          458#
TADDR   001370          731#    770    1385*  1425*  1434*  1451*  1502*  1512*
TBITVE= 000014          500#
TEMP    001352          724#    977*    978*  1098*  1254   1267   1273   1274   1275*  1325*  1329*  1329*  1332*
                        1334   1350*  1366*  1507*  1519*  1552   1600   1726*  1731*  2033*  2042*  2045*  2046*
                        2047*  2048*  2054
TENS    014466          2131*  2140*  2141   2143*  2147*  2503#
TEST    003666           996   1010   1022   1096   1113   1141#  1180
TESTAC  002426           896#  1426   1436   1453
TESTIT  003660           933    945    953    961    968   1140#
TESTR2  011446          2133#  2138   2142   2145
TKVEC = 000060           507#
TOMSG   012335          1516   2282#
TPVEC = 000064           508#
TRAPVE= 000034           506#    805*    806*
TRTVEC= 000014           501#
TRY     006554          1651#  1677
TRYAG   002302           871#   895
TST1    002562           930#   931
TST10   003050          1003#
TST11   003114          1017#
TST12   003144          1027#
TST13   003236          1043   1050#
TST14   003410          1077   1085#
TST15   003506          1094   1104#
TST16   003562          1120#
TST17   003712          1151#  1154   1155
TST2    002626           942#
TST20   003764          1169#
TST21   004052          1172   1189#
TST22   004102          1200#
TST23   004132          1212#
```

```
TST24   004162          1225#
TST25   004212          1237#
TST26   004242          1248#
TST27   004466          1296#
TST3    002652           951#
TST30   004516          1307#
TST31   004546          1318#
TST32   004716          1343    1348#
TST33   004756          1361#
TST34   005014          1373#
TST4    002670           959#
TST5    002706           966#
TST6    002724           973#
TST7    003004           983     989#
TST8    010142          1899    1901#
TYPBAD  007304          1745    1755    1759#
TYPE  = 104400           761     767     775     781     786     841     855     870     894     910    1323    1336    1342
                        1344    1387    1393    1411    1422    1438    1443    1503    1504    1513    1516    1532    1538
                        1540    1546    1559    1561    1632    1680    1762    1771    1773    1779    1782    1784    1787
                        1791    1794    1797    1799    1806    1809    1811    1815    1816    1825    1832    1855    1861
                        1870    1872    1908    1909    1912    1915    1934    1950    1959    1962    1964    1974    1976
                        2069    2075    2087    2125    2149    2222    2572    2575    2579    2715    2723    2755    2772
                        2774    2777    2779    2783    2790    2837    2990    3031#   3074
TYPEDG  006452          1539    1623#   1963
TYPOC = 104401          2763    2787    3032#
TYPON = 104403          3034#
TYPOS = 104402           778     907    1396    1414    1440    1535    1543    1554    1626    1635    1768    1858    1865
                        1967    2072    2078    3033#
TYPOUT  011522          2134    2146#
TYPRP   010350          1942    1950#   2012
TYPSET  006036          1529    1531#   1594
UNEXP   001432           749#    943    1072    1099
VADR    001332           716#    902    1462    1463    1464
VARLT1  011656          2159    2182#
VARLT2  011666          2161    2187#
VARLT3  011676          2155    2192#
VECTOR  001324           713#    943#   1035#   1059#   1067#   1072#   1088#   1099#   1399#   1465#   1476#   1477#   1480
                        1610#   1659#   1704#   2038#
VECTR1  001330           715#   1466#   1467#   1480#   1481#   1482#
VLIN    011652          1968    2179#
VNP     011646          1972    2177#
VNR     011644          1970    2152    2176#
VSET    011650          1557    2178#
VTFLG   002530           842     845     848     916#
VTINIT  013723          1934    2434#
VTSS    002176           850     853#
VVCT    001334           717#   1465    1466
V1      011622          1195    2166#
V10     011626          1206    2168#
V115    011634          1257    2171#
V144    011632          1302    2170#
V2      011624          1291    2167#
V240    011636          1231    1243    1313    2172#
V5      011640          1279    2173#
```

```
VSO     011630          1160    1218    2169#
VSOC    011642          1339    2174#
WFADJ   011552           864    2152#
WFTEST  001422           744#    789*    791*   2157
WIDE    001342           720#   1691*   1758*   1788    1801
WIDMSG  012505          1791    2302#
WRAP    003712          1147#   1444    1454
$APTHD  001000           559     565#
$ASTAT= ****** G        2907    2922
$ATYC   016244          2878    2880#
$ATY1   016220          2876#
$ATY3   016226          2822    2877#
$ATY4   016236          2728    2879#
$AUTOB  001134           596#
$BASE   001250           660#    896    1471    1472    1473
$BDADR  001122           591#
$BDDAT  001126           593#    896*    900     902*    982*   1039*   1063*   1092*   1127*   1128*   1129    1141*   1142
                        2054*   2055    2507    2509    2512
$BELL   001164           611#   2715    2747
$CDWI   001254           662#
$CHARC  016214          2829*   2849*   2856    2865*   2870#
$CKSWR= ****** G        3037
$CMTAG  001100           579#    794     795     803     809     810     811
$CM3  = 000000           609#
$CNTLG  014777          2589#
$CNTLU  014772          2588#
$CPUOP  001222           634#
$CRLF   001171           613#   1422    1443    2588    2723    2747    2755    2774    2779    2783    2838    2873
$DEVCT  001204           625#
$DEVM   001252           661#
$DOAGN  012006          2218    2224    2230#
$ENDAD  011776           545    2226#   2742
$ENDCT  011756           809    2220#
$ENDMG  012015          2222    2234#
$ENULL  012012          2233#
$ENV    001214           630#   2725    2817    2885    2909
$ENVM   001215           631#    832    2819    2824    2887
$EOP    011722          1431    1448    1458    2210#
$EOPCT  011750           809*   2217#   2221
$ERFLG  001103           582#    751*   2637    2665    2667    2673*   2695    2710*   2747
$ERMAX  001115           588#    812*   2667    2690*   2695
$ERROR  015406           803    2709#
$ERRPC  001116           589#   2507    2509    2512    2717*   2718*   2719    2747    2761
$ERRTB  001256           679#   2769
$ERRTY  015602          2722    2754#
$ERTTL  001112           586#   2716*   2747
$ESCAP  001162           610#    750*    753*    811*   2689*   2738    2740    2747
$ETABL  001214           629#
$ETEND  001256           571     663#
$FATAL  001176           622#   2913#
$FFLG   016464          2876*   2879*   2907    2916*   2924#
$FILLC  001156           607#   2842    2873
$FILLS  001155           606#   2873
$GCACR  001120           590#
```

```
$GDDAT  DC1124          592#   932#   935#   936    944#   952#   960#   967#   975#   992#  1005#  1040#  1087#
                       1108#  1121#  1129   1140   1142   1175#  2052#  2056   2507   2509
$GET42  011766         2223#
$GTSWR= ****** J       3036
$HC   = 000000          400
$HIBTS  001000          566#
$HIOCT  015126         2626#  2631#
$ICNT   001104          583#  2680#  2681   2683#  2694
$ILLUP  017120         3044   3060   3079#
$INTAG  DC1135          597#
$ITEMB  001114          587#  2719#  2727   2747   2758
$LF     001172          614#  2579   2588   2747   2873
$LFLG   016463         2917#  2923#
$LPADR  001106          594#   813#   930#  1155#  2068#  2086#  2671#  2687#  2692   2694
$LPERR  001110          585#   814#   931#  1154#  2067#  2085#  2671   2688#  2694   2737
$MADR1  001226          647#
$MADR2  001232          651#
$MADR3  001236          654#
$MADR4  001242          657#
$MAIL   001174          567    571    620#   831    2686   2725   2817
$MAMS1  001224          641#
$MAMS2  001230          649#
$MAMS3  001234          652#
$MAMS4  001240          655#
$MBADR  001002          567#
$MFLG   016462         2877#  2683   2918#  2922#
$MNEW   015015         2592#
$MSGAD  001210          627#  2893#  2896
$MSGLG  001212          628#  2898#
$MSGTY  001174          621#  2891   2899#  2911   2915#
$MSWR   015004         2590#
$MTYP1  001225          642#
$MTYP2  001231          650#
$MTYP3  001235          653#
$MTYP4  001241          656#
$MXCNT  015404         2684   2694#
$NULL   001154          605#  2844   2873
$NWTST= 000001          927#   939#   948#   956#   963#   970#   986#  1000#  1014#  1024#  1047#  1082#  1101#
                       1117#  1148#  1166#  1186#  1197#  1209#  1222#  1234#  1245#  1293#  1304#  1315#  1345#
                       1358#  1370#
$QCNT   016710         2962#  2991#  3004#
$QMODE  016712         2957#  2961#  2966   2969#  2980#  3006#
$QVER   015370         2648   2664   2672   2682   2691#
$PASS   001202          624#   831#  1375   1435#  1452#  2065   2083   2214#  2215#  2233   2678   2695
$PASTM  001006          569#
$POWER  017126         3075   3082#
$PWRAD  017114         3077#
$PWRDN  016754          807   3044#  3072
$PWRMG  017110         3075#
$PWRUP  017026         3054   3060#
$QUES   001170          612#  2572   2588   2747   2873
$RDCHR  014534         2536#  3037
$RDDEC= ****** U       3040
$RCLIN  014654         2564#  3038
```

```
$RDOCT  015026          2604*   3039
$RDSZ = 000010          2557*
$RTNAD  012010          2232*
$R2A  = ****** U        3040
$SAVRE= ****** U        3040
$SAVR6  017124          3053*   3061    3062*   3063*   3081*
$SCOPE  015130           801    2646*
$SETUP= 000037           680*    800     801     803     805     807     809     810     811     813    2212    2525    2534
                        2647    2710    2735    2742
$STUP = 177777           680*
$SVLAD  015334          2656    2685*
$SVPC = 000214           543*    548
$SWR  = 167400           390*    400     514     515     516     517     518     519     520     521     609     610     611
                         810     811     813     814     931     943     952     960     967     974     990    1004    1018
                        1028    1051    1086    1105    1121    1152    1170    1190    1201    1213    1226    1238    1249
                        1297    1308    1319    1349    1362    1374    2205    2213    2225    2231    2233    2638    2639
                        2640    2641    2642    2647    2659    2661    2662    2665    2666    2667    2674    2675    2676
                        2688    2691    2694    2701    2702    2703    2704    2705    2713    2720    2732    2735    2747
                        3078
$SWREG  001216           632*    834
$SWRMK= 000000           521     522    2642    2643    2663
$TESTN  001200           623*   2686*
$TIMES  001160           609*    810*    974*   1004*   1086*   1152*   1170*   1190*   1201*   1213*   1226*   1238*   1249*
                        1297*   1308*   1319*   1349*   1362*   1374*   2213*   2674*   2681    2684*   2694
$TKB    001146           602*    757     923    2523    2540    2546
$TKS    001144           601*    840*    872*    917     981*   1012*   1285*   2091*   2197*   2523    2538    2544
$TN   = 000035           390*    400     927     931*    939     943*    948     952*    956     960*    963     967*    970
                         974*    983     986     990*   1000    1004*   1014    1018*   1024    1028*   1043    1047    1051*
                        1077    1082    1086*   1094    1101    1105*   1117    1121*   1148    1152*   1153    1166    1170*
                        1172    1186    1190*   1197    1201*   1209    1213*   1222    1226*   1234    1239*   1245    1249*
                        1293    1297*   1304    1308*   1315    1319*   1343    1345    1349*   1358    1362*   1370    1374*
$TPB    001152           604*   2107*   2117*   2862*   2873
$TPFLG  001157           608*   2811    2873
$TPS    001150           603*   2105    2115    2860    2873
$TRAP   016714           805    3015*
$TRP  = 000007          3023*   3032*   3033*   3034*   3035*   3037    3038*   3039*   3040*
$TRPAD  016736          3020    3030*
$TSTM   001004           568*
$TSTNM  001102           581*   1153*   2212*   2637    2663    2685*   2686    2691    2695    2712    2747
$TTYIN  014762          2565    2566    2583    2587*
$TYPBN= ****** U        3035
$TYPOS= ****** U        3035
$TYPE   015736          2811*   2904    3023    3031
$TYPEC  016150          2841    2848    2855    2860*   2861
$TYPEX  016216          2866    2868    2871*
$TYPOC  016512          2960*   3032
$TYPON  016526          2959    2962*   3034
$TYPOS  016466          2955*   3033
$UNIT   001206           626*
$UNITM  001210           570*
$USWR   001220           633*
$VECT1  001244           658*   1476    1478
$VECT2  001246           659*
$XTSTR  015140          2650*
```

```
$$GET4=  000000          2225#
$OFILL   016711          2956*   2960*   2970    3005#
$40CAT=  ****** L        2647    2722
.        = 037756         528#    532#    543     544#    546#    548#    549#    555     556#    558#    560#    578#    615
                          798     813     814    2233  A 2236    2505#   2523    2587#   2588    2594    2694    2695    2747
                         2793#   2873    2925#   3056    3080    3086#   3087#
.$ASTA=  ****** U        2877    2880
.$X    = 001000          555#    560
```

```
COMMEN     1#    510#
DUMWRN   683#   1028    1051
ENDCOM     1#    510#
ERROR    404#    752     934     946     954     962     969     984     997    1011    1023    1041    1064    1079    1093
        1097   1114    1131    1161    1181    1196    1207    1219    1232    1244    1258    1280    1292    1303    1314
ESCAPE     1#    510#    749
GETPRI     1#    510#
GETSWR     1#    510#
MULT       1#    510#
NEWTST     1#    510#    927     939     948     956     963     970     986    1000    1014    1024    1047    1082    1101
        1117   1148    1166    1186    1197    1209    1222    1234    1245    1293    1304    1315    1345    1358    1370
POP        1#    510#   2627    2919    2920    3065    3066
PUSH       1#    510#   2606    2880    2882    2903    3046    3052
REPORT     1#    510#
SCOPE    405#    942     951     959     966     973     989    1003    1017    1027    1050    1085    1104    1120    1135
        1169   1189    1200    1212    1225    1237    1248    1296    1307    1318    1348    1361    1373
SETPRI     1#    510#
SETTRP  3023#   3032    3033    3034    3037    3038    3039
SETUP      1#    510#    793
SKIP       1#    510#    877     880     883     896     899     992     903     918     920     922     937     983    1043
        1077   1094    1130    1143    1172    1343    1392    1932    2154    2156
SLASH      1#    510#
SPACE    510#
STARS      1#    510#    541     552     554     561     574     615     618     927     929     939     941     948     950
         956    958     963     965     970     972     996     998    1000    1002    1014    1016    1024    1026    1047
        1049   1082    1084    1101    1103    1117    1119    1148    1150    1166    1168    1186    1188    1197    1199
        1209   1211    1222    1224    1234    1236    1245    1247    1293    1295    1304    1306    1315    1317    1345
        1347   1358    1360    1370    1372    2202    2522    2528    2557    2536    2634    2697    2749    2796    2875
        2932   3009    3042    3058
SWPSL      1#    510#    815#
TRMTRP  3023#
TYPBIN     1#    510#
TYPDEC     1#    510#
TYPNAM     1#    510#
TYPNUM     1#    510#
TYPOCS     1#    510#    776     905    1394    1412    1533    1541    1552    1624    1633    1766    1856    1863    1965
        2070   2076
TYPOCT     1#    510#   2761    2785
TYPTXT     1#    510#
$$CMRE   572#
$$CMTM   572#
$$ESCA     1#    510#
$$NEWT     1#    510#    927     939     948     956     963     970     986    1000    1014    1024    1047    1082    1101
        1117   1148    1166    1186    1197    1209    1222    1234    1245    1293    1304    1315    1345    1358    1370
$$SET   3023#   3032    3033    3034    3037    3038    3039
$$SETM   831#
$$SKIP     1#    510#    983    1043    1077    1094    1172    1343
.EQUAT     1#    390#    400
.HEADE     1#    390#
.KT11      1#
.SETUP     1#    390#    680
.SWPHY     1#    390#    510
.SWR_0   522#
.$ACT1     1#    390#    539
```

```
.$APTB      1#    390#    616#
.$APTH      1#    390#    550
.$APTY      1#    390#   2873
.$ASTA      1#
.$CATC      1#    390#    526
.$CMTA      1#    390#    572
.$DB2D      1#
.$DB20      1#
.$DIV       1#
.$ECOP      1#    390#   2200
.$ERRC      1#    390#   2695
.$ERRT      1#    390#   2747
.$MULT      1#
.$PARM    390#
.$POWE      1#    390#   3040
.$RAND      1#    390#
.$RDDE      1#
.$RDCC      1#    390#   2594
.$READ      1#    390#   2520
.$R2A2      1#
.$SAVE      1#    390#
.$SB2D      1#
.$SB20      1#
.$SCOP      1#    390#   2632
.$SIZE      1#
.$SPAC    390#
.$SJPR      1#
.$SWDC    390#
.$TRAP      1#    390#   3007
.$TYPB      1#
.$TYPC      1#    390#
.$TYPE      1#    390#   2794
.$TYPO      1#    390#   2930
.$4CCA      1#
.1170       1#
```

```
ABO      853   901   991  1006  1269  1281  1404  1474  1585  1611  1670  1712  1719  1725
         1744  1745  1751  1754  1758  1761  1862  2003  2214  2680  2716  2915  2996  2997
         3063
ABOB     2140  2144  2685  2710  2869
ABS      537   538   764   770   869   878   881   884   887   890   893  1431  1448  1458
         2031
ABS      842   845   848   857   864   933   945   953   961   968   996  1010  1022
         1045  1056  1076  1096  1113  1156  1158  1180  1191  1193  1202  1204  1214
         1229  1239  1241  1251  1252  1255  1262  1263  1277  1286  1287  1289  1298
         1311  1323  1326  1335  1337  1351  1353  1355  1363  1367  1377  1426  1427
         1437  1444  1445  1453  1454  1455  1506  1521  1523  1524  1529  1531  1539
         1568  1578  1581  1584  1594  1706  1772  1778  1786  1790  1793  1804  1814  1820
         1911  1914  1916  1921  1939  1940  1942  1958  1961  1963  1981  1984
         1988  2002  2012  2226  2722  2728  2822  2841  2848  2855  2904
         756   757   762   768   776   783   784   791   795   799   801   802   803   804
         806   807   808   809   813   814   817   818   819   820   825   827   828   829
         858   859   860   861   862   863   866   874   896   897   898   905   912   913
         930   931   932   943   944   952   960   967   974   976   977   982   990   992
         1014  1018  1021  1032  1035  1036  1039  1040  1055  1057  1059  1060  1063  1067
         1069  1075  1086  1087  1088  1089  1092  1098  1099  1105  1108  1109  1110
         1121  1122  1123  1127  1133  1140  1141  1152  1153  1154  1155  1170  1173  1174
         1176  1182  1190  1201  1213  1226  1238  1249  1254  1260  1261  1270  1273  1274
         1297  1308  1319  1320  1321  1322  1325  1330  1334  1349  1350  1362  1366  1374
         1383  1394  1399  1402  1403  1406  1409  1412  1417  1425  1430  1434  1439  1447
         1457  1471  1472  1473  1476  1480  1483  1485  1486  1489  1491  1502  1505  1507
         1515  1518  1519  1520  1527  1528  1533  1541  1547  1549  1552  1567  1571  1577
         1583  1592  1593  1600  1604  1605  1609  1610  1623  1624  1633  1644  1649  1653
         1657  1659  1662  1663  1666  1682  1683  1687  1688  1698  1702  1703  1704  1705
         1714  1721  1735  1736  1737  1763  1766  1777  1785  1788  1792  1800  1803  1817
         1819  1833  1834  1836  1838  1843  1844  1853  1856  1863  1875  1876  1877  1883
         1887  1888  1889  1890  1891  1892  1893  1894  1897  1900  1903  1904  1910  1913
         1919  1926  1937  1938  1941  1957  1960  1965  1995  1997  2001  2011  2029  2031
         2035  2038  2039  2052  2053  2054  2055  2056  2067  2068  2070  2076  2085  2086
         2102  2129  2152  2155  2159  2161  2162  2219  2223  2536  2537  2564  2565  2580
         2582  2593  2604  2605  2606  2607  2608  2610  2625  2626  2627  2628  2629  2652
         2655  2658  2671  2683  2684  2687  2688  2691  2692  2712  2717  2737  2740  2756
         2775  2775  2780  2782  2786  2815  2816  2821  2829  2844  2881  2882  2889  2993
         2899  2901  2903  2913  2919  2920  2955  2963  2964  2965  2971  2978  2996  2997
         2999  3000  3015  3016  3020  3044  3045  3046  3047  3048  3049  3050  3051  3052
         3054  3060  3061  3065  3066  3067  3068  3069  3070  3071  3072  3073  3076
ABSB     1124  1478  1601  1658  1776  1805  2107  2117  2540  2546  2569  2574  2613  2686
         2719  2727  2826  2954  2862  2876  2877  2879  2956  2957  2960  2961  2962  2966
         2969  2970  2989  3018
ACS      1526  1591  1840  2059  2126  2967
ACSB     1551  2211  2227  2228  2229
ADDSE    756   856   965   980   1009  1284  2090  2225
ADDSS    936   2616  2618  2620  2973  2975  2976  2977  2979
ADDSS    714   786   826   2556  2584  2630  2693  2746  2831  3001  3078
ADDSS    1314  925   1136  1145  1378  1383  1495  1560  1562  1575  1595  1620  1638  1678  1917
ADDSS    1924  1935  1943  1975  1977  1993  2013  2027  2049  2063  2081  2088  2093  2119  2150
ADDSS    2655  2784  2871  2921  3021
ADDSS    1258  1275  1329  1332  1548  1574  1588  1675  1764  1835  1845  1986  1992  2006  2057
         2718  2896
```

```
                960   963   965   967   970   972   974   975   983   986   988   990  1000  1002  1004
               1006  1014  1016  1018  1024  1026  1028  1043  1047  1049  1051  1077  1082  1094  1096
               1098  1094  1101  1103  1105  1117  1119  1121  1130  1143  1148  1150  1152  1153  1156
               1168  1170  1171  1172  1186  1189  1190  1191  1197  1199  1201  1202  1209  1211  1213
               1214  1223  1224  1226  1227  1234  1236  1238  1239  1245  1247  1249  1250  1293  1295
               1298  1298  1304  1306  1308  1309  1315  1317  1319  1320  1343  1345  1347  1349  1350
               1358  1360  1362  1363  1370  1372  1374  1375  1392  1397  1398  1415  1416  1536  1537
               1544  1545  1555  1556  1627  1628  1636  1637  1769  1770  1859  1860  1866  1867  1932
               1968  1969  2073  2074  2079  2080  2154  2156  2202  2203  2204  2205  2206  2207  2211
               2217  2220  2222  2223  2225  2231  2233  2234  2236  2522  2524  2525  2528  2529  2557
               2565  2566  2570  2571  2587  2588  2594  2596  2599  2611  2634  2637  2642  2647  2659
               2661  2662  2663  2665  2666  2667  2676  2678  2686  2688  2693  2694  2695  2697  2700
               2710  2713  2720  2722  2723  2725  2732  2735  2742  2746  2747  2749  2764  2780  2796
               2815  2875  2877  2880  2907  2922  2932  3009  3015  3019  3023  3032  3033  3034  3035
               3036  3037  3038  3039  3040  3042  3052  3053  3058  3065  3066  3074  3076  3078  3082
.ERR            402   517   520   521   522   542   546   548   553   555   562   575   578   581   609
                615   619   750   779   781   799   877   880   883   886   889   892   903   909   918
                920   922   927   928   929   930   931   937   940   941   942   943   949   950   951
                956   957   958   959   960   964   965   966   967   971   972   973   974   984   987
                988   989   990  1001  1002  1003  1004  1015  1016  1017  1018  1025  1026  1027  1028
               1044  1048  1049  1050  1051  1078  1083  1084  1085  1086  1095  1102  1103  1104  1105
               1118  1119  1120  1121  1130  1143  1148  1149  1150  1151  1152  1167  1168  1169  1170
               1173  1187  1188  1189  1190  1198  1199  1200  1201  1210  1211  1212  1213  1223  1224
               1225  1226  1235  1236  1237  1238  1246  1247  1248  1249  1250  1294  1295  1296  1297
               1305  1306  1307  1308  1316  1317  1318  1319  1320  1344  1346  1347  1348  1349  1350
               1359  1360  1361  1362  1363  1371  1372  1373  1374  1375  1392  1398  1416  1417  1537
               1545  1556  1557  1628  1629  1637  1638  1770  1771  1860  1861  1867  1869  1932  1969
               2074  2079  2081  2154  2156  2203  2206  2211  2218  2221  2233  2523  2525  2529  2531
               2536  2557  2558  2567  2571  2588  2597  2635  2660  2663  2664  2667  2694  2695  2698
               2700  2713  2742  2747  2750  2765  2794  2797  2876  2933  3010  3016  3043  3059  3076
.IFT            2531  2536  2615  2631  2632  2675  2723
.IFTF           2525  2529  2532  2611  2615  2631  2673  2722
.IIF             390   395   400   514   515   516   518   521   522   532   615   619   777   800   803
                 809   810   811   813   814   906  1395  1413  1534  1542  1553  1625  1634  1767  1857
                1864  1966  2071  2077  2204  2205  2212  2213  2233  2236  2523  2580  2588  2594  2638
                2639  2640  2641  2642  2643  2647  2674  2675  2691  2694  2695  2701  2702  2703  2704
                2705  2710  2735  2742  2747  2762  2787  2873  3031  3032  3033  3034  3037  3038  3039
.IRP             600   927   939   948   956   963   970   986  1000  1014  1024  1047  1082  1101  1117
                1148  1166  1186  1197  1209  1222  1234  1245  1293  1304  1315  1345  1358  1370  2211
                2606  2627  2881  2882  2903  2919  2920  3046  3052  3065  3066
.LIST              1   390   510   521   532   609   616   619   680   815   927   931   939   943   948
                 952   956   960   963   967   970   974   986   990  1000  1004  1014  1018  1024  1028
                1047  1051  1082  1086  1101  1105  1117  1121  1148  1152  1166  1170  1186  1190  1197
                1201  1209  1213  1222  1226  1234  1238  1245  1249  1293  1297  1304  1308  1315  1319
                1345  1349  1358  1362  1370  1374  2212  2225  2557  2642  2742  3023  3031  3032  3033
                3034  3035  3037  3038  3039  3040
.MACRO             1   522   572   683   831  3023
.MCALL           390   510   616   815
.MEXIT           664
.NLIST             1   390   510   521   532   609   616   619   680   815   927   931   939   943   948
                 952   956   960   963   967   972   974   986   990  1000  1004  1014  1018  1024  1028
                1047  1051  1082  1086  1101  1105  1117  1121  1148  1152  1166  1170  1186  1190  1197
                1201  1209  1213  1222  1226  1234  1238  1245  1249  1293  1297  1304  1308  1315  1319
                1345  1349  1358  1362  1370  1374  2212  2225  2557  2642  2742  3023  3031  3032  3033
```

E08

	3034	3035	3037	3038	3039	3040									
.PAGE	572	665													
.REM	9														
.REPT	532														
.SBTTL	400	510	526	535	539	550	572	616	665	709	755	788	793	839	854
	927	939	948	956	963	970	986	1000	1014	1024	1047	1082	1101	1117	1146
	1148	1166	1186	1197	1209	1222	1234	1245	1293	1304	1315	1345	1358	1370	1384
	1424	1433	1450	1459	1501	1511	2200	2237	2520	2594	2632	2695	2747	2794	2873
	2930	3007	3023	3040											
.TITLE	390														
.WORD	532	533	534	547	566	567	568	569	570	571	580	583	584	595	586
	589	590	591	592	593	594	595	598	599	600	621	622	623	624	625
	626	627	628	632	633	634	647	651	654	657	658	659	660	661	662
	2217	2220	2232	2631	2773	2778	2823	2870	2905	3006	3075	3077			

ERRORS DETECTED: 0

*,DZADLB.SEQ/SOL_CRF/ML:TOC=DSKZ:SYSMAC.SML,DZADLB.P11
RUN-TIME: 85.62 B SECONDS
CORE USED: 33K

F08

Spooler runtime 14 Seconds, 62 KCS, 340 disk reads, 3 disk writes, 95 pages

[illegible]

```
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111
0000000111111111122222222223333333333444444444455555555556666666666777777777788888888889999999999000000000011111111112222222222333
```

```
  :0                        ...B1     2130  011432  105037  ...B5
  63                        ...C1     2175  011642  000062  ...C5
 116                        ...D1     2209                  ...D5
                            ...E1     2246  012106  023440  ...E5
 172                        ...F1     2299  012470  020051  ...F5
 213                        ...G1     2351  013076  020040  ...G5
 241                        ...H1     2405  013556  020122  ...H5
                            ...I1                           ...I5
 297                        ...J1     2466  014162  020056  ...J5
                            ...K1     2510  014512  001124  ...K5
 352                        ...L1     2529                  ...L5
 388                        ...M1     2583  014750  012766  ...M5
 453          000020        ...N1     2603                  ...N5

 507          000060        ...B2     2641                  ...B6
 548          000214        ...C2     2695                  ...C6
 581  001102      000        ...D2     2749                  ...D6
 635                        ...E2     2803                  ...E6
 644                        ...F2     2857  016142  001372  ...F6
 718  001336  000000        ...G2     2911  016412  005737  ...G6
 764  001510  000137        ...H2     2939                  ...H6
 797  001644  022706        ...I2     2993  016660  002402  ...I6
 845  002144  004737        ...J2     3016  016716  016600  ...J6
 863  002244  012737        ...K2     3049  016776  010346  ...K6
 913  002522  012637        ...L2  ADDW11=  000000          ...L6
 935  002612  006137        ...M2  BASECH   001336          ...M6
 979  002760  001375        ...N2  CLEAR1   006744          ...N6

1033  003152  004737        ...B3  FIXADR   005500          ...B7
1087  003420  012737        ...C3  NOITST   010312          ...C7
1126  003614  001376        ...D3  RANDY    010722          ...D7
1155  003736  012737        ...E3                           ...E7
1206  004126  011626        ...F3  SW10   = 002000          ...F7
1254  004270  013704        ...G3  TST34    005014          ...G7
1302  004512  011632        ...H3  $ATYC    016244          ...H7
1324  004604  000062        ...I3  $INTAG   001135          ...I7
1354  004746  000007        ...J3  $STUP = 177777          ...J7
1393  005112  104400        ...K3                           ...K7
1433                        ...L3  NEWTST      1*    510*   ...L7
1468  005472  005337        ...M3  .$EOP       1*    390*   ...M7
1520  005770  013737        ...N3             2714   2736   ...N7

1570  006204  063702        ...B4             1216   1227   ...B8
1605  006362  012700        ...C4  .ASCIZ     611    614   ...C8
1649  006542  012737        ...D4             2217   2220   ...D8
1688  006756  012701        ...E4  .TITLE     390   3007   ...E8
1742  007222  020227        ...F4   **END** USER DAVIES,TOM ...F8
1794  007466  104400        ...G4
1837  007674  010120        ...H4
1886  010056  012700        ...I4
1934  010304  104400        ...J4
1987  010544  004537        ...K4
2037  011036  001376        ...L4
2073  011216     002        ...M4
2098  011310  005002        ...N4
```