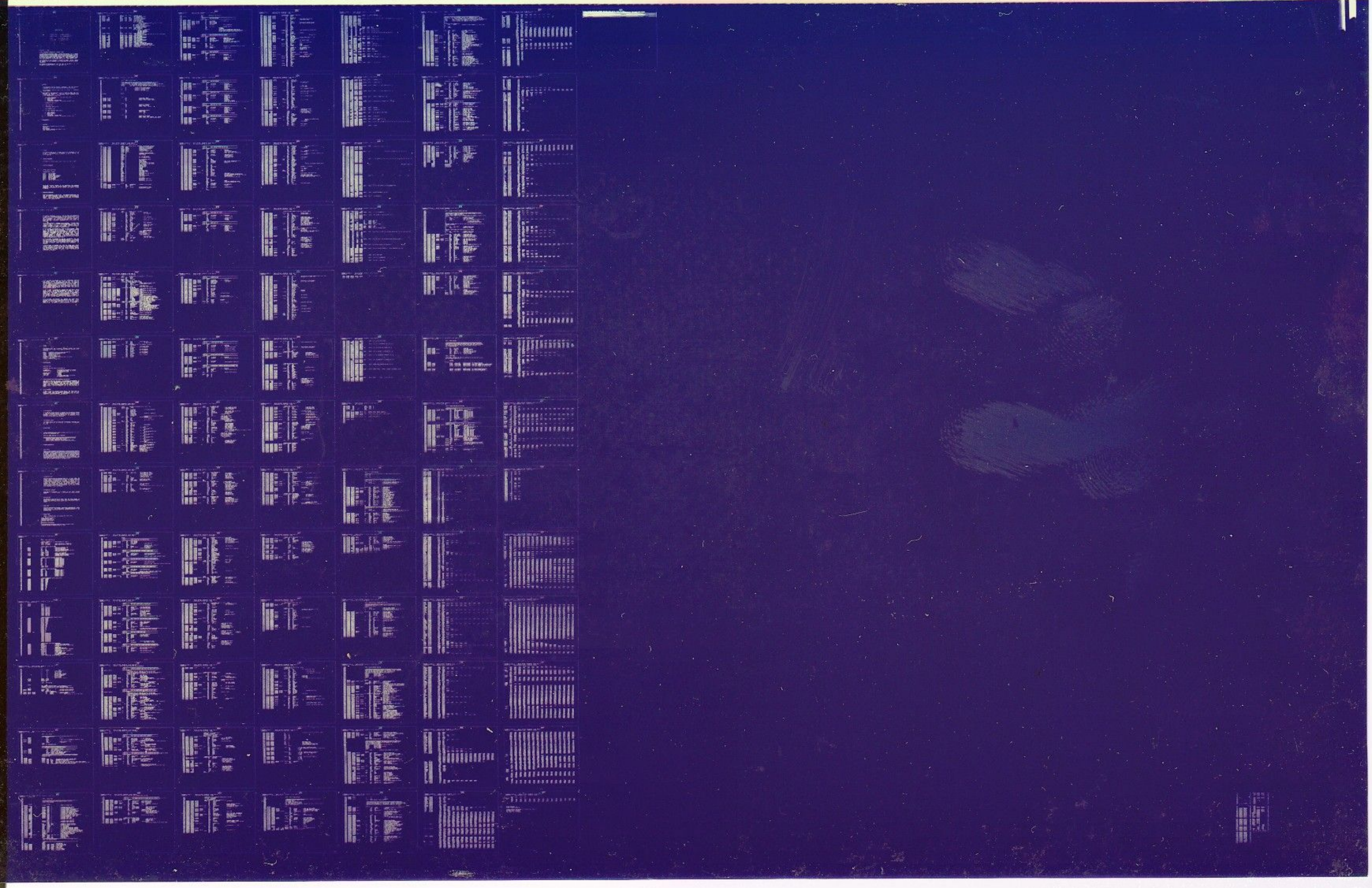
AD11K

PERFORMANCE TEST
MD-11-DZADL-A

EP-DZADL-A-DL-A

FICHE 1 OF 1

NOV 1976
digital
MADE IN USA



.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZADL-A
PRODUCT NAME: AD11K PERFORMANCE TEST
DATE: AUGUST 21, 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 TEST:". THEN TYPE THE LETTER YOU WANT, ACCORDING TO THE TABLE LISTED AND HIT CARRIAGE RETURN.

TWO CONTROL CHARACTERS, ↑A AND ↑C, ARE SET ASIDE FOR INTERRUPTING A TEST AND TRANSFERRING CONTROL TO EITHER THE BEGINNING OF THE DIAGNOSTIC (↑C) OR TO THE BEGINNING OF THE SPECIFIC TEST WHICH WAS IN PROGRESS (↑A). DURING THE LOGIC TESTS WHILE A RESET IS BEING PERFORMED, ↑C OR ↑A WILL NOT BE EXECUTED UNTIL AFTER THE RESET HAS BEEN COMPLETED, THEREFORE HIT ↑C OR ↑A UNTIL IT IS SUCCESSFUL.

FOR MACHINES WITHOUT A HARDWARE SWITCH REGISTER, LOCATION SWREG (176) IS USED AS A SOFTWARE SWITCH REGISTER. TO MODIFY THE CONTENTS OF SWREG, TYPE ↑G. THE PROGRAM RESPONDS WITH THE CURRENT CONTENTS OF SWREG AND A SLASH. TYPE THE DESIRED NEW CONTENTS OF SWREG FOLLOWED BY A CARRIAGE RETURN.

IF "W" IS TYPED, THE PROGRAM WILL TYPE "XX AD11K'S FOUND". WHERE XX IS THE NUMBER OF AD11K'S IN OCTAL. IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K. THE PROGRAM WILL RUN THROUGH THE LOGIC SUBTESTS, THE NOISE TEST ON 8 EDGES, THE INTERCHANNEL SETTLING TEST ON 8 EDGES, AND THE DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST. A G5036 WRAPAROUND MODULE IS REQUIRED. THE PROGRAM SUPPORTS AD11K EXPANSION BEYOND 16. CHANNELS. TO RUN THIS TEST ON A GROUP OF CHANNELS OTHER THAN 0-17, LOAD 20,40, OR 60 INTO LOCATION BASECH (1336) FOR CHANNELS 20-37, 40-57, 60-77.

IF "C" IS TYPED, THE PROGRAM WILL RUN THE CALIBRATION TEST AND WILL LOOP ON THAT TEST UNTIL THE OPERATOR HALTS IT. IF A CERTAIN AD11K IS TO BE TESTED, ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO $BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO THE LOW BYTE OF $VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

F "N" IS TYPED, THE PROGRAM WILL RUN THE NOISE TEST TAGGED "BEGINN" AND WILL LOOP ON THIS TEST UNTIL THE OPERATOR HALTS IT. IF A CERTAIN AD11K IS TO BE TESTED ITS STATUS REGISTER ADDRESS MUST BE LOADED INTO $BASE (1250), AND ITS VECTOR ADDRESS MUST BE LOADED INTO THE LOW BYTE OF $VECT1 (1244) (THE HIGH BYTE CONTAINING THE PRIORITY).

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

IF "A" IS TYPED, THE PROGRAM WILL EXECUTE THE LOGIC TESTS, ANALOG TESTS, NOISE, SETTLE AND DIFFERENTIAL LINEARITY. AT THE BEGINNING OF THE TEST THE PROGRAM WILL TYPE "XX ADV11'S FOUND". WHERE XX IS THE NUMBER OF ADV11'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 ADV11'S FOUND". WHERE XX IS THE NUMBER OF ADV11'S IN OCTAL IF THE NUMBER IS GREATER THAN 1, THE TEST WILL BE RUN SUCCESSIVELY ON EACH AD11K.

6.0 ERRORS

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

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

7.0 MISCELLANEOUS

7.1 EXECUTION TIME

EXECUTION TIME FOR EACH OF THE TESTS IS:

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

7.2 STATUS REGISTER AND VECTOR ADDRESSES AND PRIORITY

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

7.3 AD11K PRIORITY

IF AD11K IS SET FOR A PRIORITY OTHER THAN 6, THE HIGH BYTE OF $VECT1 (1244) MUST BE ADJUSTED ACCORDINGLY (THE LOW BYTE CONTAINING THE VECTOR ADDRESS). IF MORE THAN ONE AD11K IS BEING TESTED, ALL MUST BE SET AT THE SAME PRIORITY.

7.4 SWITCH REGISTER

IF A HARDWARE SWITCH REGISTER IS PRESENT AND THE OPERATOR DESIRES TO USE A SOFTWARE SWITCH REGISTER AND THE ↑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 BCO8R CONNECTION A-A, VV-VV
BCO8R 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 LOGIC AND THE CORRECT SETTING OF THE ERROR FLAG.

9.2 CALIBRATION TEST

THIS TEST BEGINS WHEN THE OPERATOR TYPES "C", IT THEN LOADS THE CHANNEL FROM THE SWITCH REGISTER BITS 0-7 AND DOES A CONVERSION ON THAT CHANNEL. IF SWR BIT 13 IS DOWN, IT PRINTS OUT THE CONVERTED VALUE ON THE TELETYPE; OTHERWISE, IF SWR BIT 13 IS UP, IT PUTS THE CONVERTED VALUE IN THE DISPLAY REGISTER. THE OPERATOR MAY CHANGE THE CHANNEL AT ANY TIME DURING THE TEST, HOWEVER THE NEW VALUES FROM THE NEW CHANNEL WILL NOT BE PRINTED UNTIL THE NEXT LINE OF 8 VALUES IS PRINTED. THE 8 VALUES ON EACH LINE CORRESPOND TO ONLY ONE CHANNEL.

9.3 DIFFERENTIAL LINEARITY

THIS TEST IS TO DETERMINE IF A CHANGE IN THE INPUT VOLTAGE REPRESENTS A SIMILAR CHANGE IN THE RESULTING CONVERTED BINARY VALUE.

9.4 SETTLING TEST

THE PURPOSE OF THIS TEST IS TO CHECK THAT THE TIME NEEDED TO SETTLE AND CORRECTLY REPORT A NEW INPUT VALUE AFTER SWITCHING CHANNELS DOES NOT EXCEED THE EXPECTED AMOUNT OF TIME FOR SUCH A CHANGE.

9.5 NOISE TEST

THIS TEST MEASURES THE INTERNAL SHORT-TERM REPEATABILITY NOISE WITHIN THE A/D. RMS NOISE EQUALS 1 STANDARD DEVIATION OF THE GAUSSIAN CURVE, PEAK NOISE EQUALS 2.3 STANDARD DEVIATION OF THE GAUSSIAN CURVE.

9.6 ANALOG TESTS

THESE 11 SUBTESTS CHECK THE CHANNELS AND THEIR OUTPUT.

```
        %
.TITLE  MAINDEC-11-DZADL-A
;*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-CO),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  001625                          JMP     @#BEGIN2                ;START ADDRESS FOR OPTION TEST AREA
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                                        $SVPC=.                 ;SAVE PC
000214                                  .=46
000046                                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000046  011764                         .=52
000052                                  .WORD   0               ;;2)SET LOC.52 TO ZERO
000052  000000                         .=$SVPC                 ;; RESTORE PC
000214                          .=1000
001000                          .SBTTL  APT PARAMETER BLOCK

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

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

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                  $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
001206  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001210  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
```

```
000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
                        $ETABLE:                ;;APT ENVIRONMENT TABLE
000                     $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
000                     $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                        ;*                      BITS 15-11=CPU TYPE
                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                              11/70=06,PDQ=07,Q=10
                        ;*                      BIT 10=REAL TIME CLOCK
                        ;*                      BIT  9=FLOATING POINT PROCESSOR
                        ;*                      BIT  8=MEMORY MANAGEMENT
000                     $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
000                     $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                        ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              500 NSEC MOS=003
000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                        ;*                       MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
000                     $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
000                     $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
000                     $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
000                     $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
000                     $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
000                     $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
140340                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
170400                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
                        $ETEND:
                        .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001256                          $ERRTB:


                                ;ITEM   1
001256  014134                          EM1                     ;STATUS REG. ERROR
001260  014274                          DH1                     ;ERRPC STREG EXPECTED ACTUAL
001262  014460                          DT1                     ;$ERRPC, STREG, $GDDAT, $BDDAT
001264  014520                          DF1

                                ;ITEM   2
001266  014162                          EM2                     ;FAILED TO INTERRUPT
001270  014415                          DH3                     ;ERRPC STREG ACTUAL
001272  014510                          DT3                     ;$ERRPC, STREG,  $BDDAT
001274  014520                          DF1

                                ;ITEM   3
001276  014212                          EM3                     ;UNEXPECTED INTERRUPT
001300  014415                          DH3                     ;ERRPC STREG
001302  014510                          DT3                     ;$ERRPC, STREG
001304  014520                          DF1

                                ;ITEM   4
001306  014243                          EM4                     ;ERROR ON A/D CHANNEL
001310  014332                          DH2                     ;ERRPC  STREG  CHAN  NOMINAL  TOL  ACTUAL
001312  014472                          DT2                     ;$ERRPC,STREG,CHANL,$GDDAT,$PREAD,$BDDAT
001314  014520                          DF1
```

```
                                .SBTTL      MISCELLANEOUS, TEMPORARY, AND STORAGE LOCATIONS
001316  170400                  STREG:  ABASE                   ;ADDRESS OF STATUS REGISTER
001320  170401                  PDST1:  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                       ;VTSS 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 ERRCR
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  012134                  TYPE    ,CMSG               ;ECHO CHARACTER
001500  012706  001100                  MOV     #STACK,SP
001504  004737  011254                  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  012127                  TYPE    ,AMSG               ;ECHO CHARACTER
001526  012706  001100                  MOV     #STACK,SP
001532  004737  011254                  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  012141                  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  012321                  TYPE    ,SLASH
001600  104406                          RDOCT                   ;READ NEW VALUE
001602  012677  177332                  MOV     (SP)+,@SWR          ;LOAD NEW SWREG VALUE
001606  012600                  POPRO:  MOV     (SP)+,R0
001610  000002                  RETURN: RTI
001612  104400  012125          NONE:   TYPE    ,QUEST              ;TYPE "?"
001616  000773                          BR      POPRO
```

```
                                                .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  015116  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001664  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001672  012737  015374  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001700  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001706  012737  016702  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001714  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
001722  012737  016742  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001730  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001736  013737  011744  011736                  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,@$TKS
  841  002126  104400  013561                    TYPE    ,C0             ;TYPE ASCIZ STRING
  842  002132  004737  002524                    JSR     PC,VTFLG        ;GET A CHARACTER
  843  002136  020027  000033                    CMP     R0,#33
  844  002142  001017                            BNE     NOVT55          ;NO VT55 PRESENT
  845  002144  004737  002524                    JSR     PC,VTFLG        ;GET A CHARACTER
  846  002150  020027  000057                    CMP     R0,#57
  847  002154  001012                            BNE     NOVT55          ;NO VT55 PRESENT
  848  002156  004737  002524                    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  013724          NOVT55: TYPE    ,HEAD1
002206  000005                  REST1:  RESET
002210  004737  005472                  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  011540                  JSR     PC,WFADJ                ;STANDARD OR OPTION TEST TOLERANCES?
002256  000005                  BEG2:   RESET                   ;RESTART ADDRESS
002260  005737  000042                  TST     @#42                    ;IS IT CHAINED?
002264  001402                          BEQ     1$
002266  000137  005222                  JMP     BEGL                    ;GO TO LOGIC TESTS
002272  104400  013367          1$:     TYPE    ,MSG71
002276  104405                  TRYAG:  RDLIN
002300  052777  000100  176636          BIS     #100,@$TKS
002306  005037  177776                  CLR     PSW
002312  012600                          MOV     (SP)+,R0                ;READ ANSWER
002314  142710  000040                  BICB    #40,(R0)
002320  121027  000101                  CMPB    (R0),#'A                ;IS IT A?
002324  001002                          BNE     1$              ;;NO, TRY C
002326  000137  005260                  JMP     BEGINA                  ;GO TO AUTO TEST
002332  121027  000103          1$:     CMPB    (R0),#'C                ;IS IT C?
002336  001002                          BNE     2$              ;;NO, TRY L
002340  000137  005036                  JMP     BEGINC                  ;GO TO CALIBRATION TEST
002344  121027  000114          2$:     CMPB    (R0),#'L                ;IS IT L?
002350  001002                          BNE     3$              ;;NO, TRY N
002352  000137  005222                  JMP     BEGL                    ;GO TO LOGIC TESTS
002356  121027  000116          3$:     CMPB    (R0),#'N                ;IS IT N?
002362  001002                          BNE     4$              ;;NO, TRY S
002364  000137  005646                  JMP     BEGINN                  ;GO TO NOISE TEST
002370  121027  000123          4$:     CMPB    (R0),#'S                ;IS IT S?
002374  001002                          BNE     5$              ;;NO, TRY W
002376  000137  005716                  JMP     BEGINS                  ;GO TO SETTLE TEST
002402  121027  000127          5$:     CMPB    (R0),#'W                ;IS IT W?
002406  001002                          BNE     6$              ;;NO,TRY AGAIN
002410  000137  005346                  JMP     BEGINW                  ;GO TO WRAPAROUND TEST
002414  104400  012125          6$:     TYPE    ,QUEST
002420  000726                          BR      TRYAG                   ;WAIT FOR CHARACTER
002422  013737  001250  001126  TESTAD: MOV     $BASE,$BDDAT    ;SETUP TO TEST FOR AD11K'S
002430  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE ERRVEC
002434  012737  002466  000004          MOV     #2$,ERRVEC      ;SET UP FOR TIME OUT ERROR
002442  005037  001360                  CLR     NBEXT           ;CLEAR AD11K COUNTER
002446  005777  176454          1$:     TST     @$BDDAT         ;ADDRESS AD11K
002452  005237  001360                  INC     NBEXT           ;INCREMENT AD11K COUNTER
002456  063737  001332  001126          ADD     VADR,$BDDAT     ;GET NEXT AD11K
002464  000770                          BR      1$              ;;TRY NEXT AD11K
```

```
 903  002466  022626                  2$:     CMP     (SP)+,(SP)+     ;POP 2 WORDS OFF STACK
 904  002470  013746  001360                  MOV     NBEXT,-(SP)     ;;SAVE NBEXT FOR TYPECUT
 905                                                                    ;;TYPE NUMBER OF AD11K'S
 906  002474  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 907  002476     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 908  002477     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 909  002500  104400  012727                  TYPE    ,MSG50
 910  002504  005337  001360                  DEC     NBEXT           ;ADJUST AD11K COUNT
 911  002510  013737  001360  001362          MOV     NBEXT,NMBEXT    ;KEEP COUNT OF NUMBER
 912  002516  012637  000C04                  MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
 913  002522  000207                          RTS     PC
 914
 915  002524  005000                  VTFLG:  CLR     R0              ;TEST FOR PRESENCE
 916  002526  105777  176412          1$:     TSTB    @#TKS           ;OF VT55
 917  002532  100404                          BMI     2$              ;;VT55 RESPONDS WITH <33><57>[<103> OR <105>]
 918  002534  005300                          DEC     R0
 919  002536  001373                          BNE     1$
 920  002540  005726                          TST     (SP)+           ;;
 921  002542  000617                          BR      NOVT55          ;POP A WORD OFF STACK
 922  002544  017700  176376          2$:     MOV     @#TKB,R0        ;;NO VT55 PRESENT
 923  002550  042700  177600                  BIC     #177600,R0              ;TEST VT55 CODE
 924  002554  000207                          RTS     PC
```

```
                                BEGINL:
                                ;;*****************************************************************
                                ;*TEST 1         FLOAT A ONE THRU MULTIPLEXER BITS
                                ;;*****************************************************************
002556 012737 002556 001106     TST1:   MOV     #TST1,$LPADR
002564 012737 002556 001110             MOV     #TST1,$LPERR
002572 012737 000400 001124             MOV     #BIT8,$GDDAT            ;LOAD FIRST BIT
002600 004737 003654            2$:     JSR     PC,TESTIT
002604 104001                           ERROR   1                       ;FAILED TO LOAD + READ BIT
002606 006137 001124            1$:     ROL     $GDDAT                  ;GET NEXT BIT
002612 023727 001124 040000             CMP     $GDDAT,#BIT14           ;FINISHED?
002620 001367                           BNE     2$              ;;NO,GO TO NEXT TEST

                                ;;*****************************************************************
                                ;*TEST 2         LOAD AND READ BACK INTERRUPT ENABLE BIT6
                                ;;*****************************************************************
002622 000004                   TST2:   SCOPE
002624 012777 001432 176472             MOV     #UNEXP,@VECTOR          ;SETUP FOR UNEXPECTED INTERUPT
002632 012737 000100 001124             MOV     #BIT6,$GDDAT            ;LOAD EXPECTED DATA
002640 004737 003654                    JSR     PC,TESTIT
002644 104001                           ERROR   1                       ;FAILED TO LOAD + READ INTERRUPT ENABLE

                                ;;*****************************************************************
                                ;*TEST 3         LOAD AND READ BACK CLOCK OVERFLOW START ENABLE BIT5
                                ;;*****************************************************************
002646 000004                   TST3:   SCOPE
002650 012737 000040 001124             MOV     #BIT5,$GDDAT            ;LOAD EXPECTED DATA
002656 004737 003654                    JSR     PC,TESTIT
002662 104001                           ERROR   1                       ;FAILED TO LOAD + READ CLOCK OVERFLOW START ENAB

                                ;;*****************************************************************
                                ;*TEST 4         LOAD AND READ BACK EXTERNAL START ENABLE BIT4
                                ;;*****************************************************************
002664 000004                   TST4:   SCOPE
002666 012737 000020 001124             MOV     #BIT4,$GDDAT    ;LOAD EXPECTED DATA
002674 004737 003654                    JSR     PC,TESTIT
002700 104001                           ERROR   1                       ;FAILED TO LOAD + READ EXT. START ENALLE
                                ;;*****************************************************************
                                ;*TEST 5         LOAD AND READ BACK ERROR FLAG BIT15
                                ;;*****************************************************************
002702 000004                   TST5:   SCOPE
002704 012737 100000 001124             MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
002712 004737 003654                    JSR     PC,TESTIT
002716 104001                           ERROR   1                       ;FAILED TO LOAD + READ ERROR FLAG
```

```
                                        ;;**************************************************************
                                        ;*TEST 6         TEST INIT CLEARS BITS 1,4-6,8-13,15
                                        ;;**************************************************************
002720  000004                  TST6:   SCOPE
002722  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
002730  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED DATA
002734  012777  137562  176354  2$:     MOV     #137562,@STREG  ;SET STATUS REGISTER
002742  012737  000010  001352          MOV     #10,TEMP
002750  005337  001352          1$:     DEC     TEMP            ;DELAY FOR ONE-SHOT
002754  001375                          BNE     1$              ;IN ANALOG PACKAGE
002756  000005                          RESET                   ;INITIALIZE
002760  052777  000100  176156          BIS     #100,@STKS      ;SET INTRPT. ENABLE
002766  017737  176324  001126          MOV     @STREG,$BDDAT   ;READ STATUS REGISTER
002774  001401                          BEQ     TST7            ;;NEXT TEST
002776  104001                          ERROR   1               ;RESET FAILED TO CLEAR AD ST. REG. BITS

                                        ;;**************************************************************
                                        ;*TEST 7         TEST  DONE FLAG  SETS AND BITO CLEARS ON END OF CONV.
                                        ;;**************************************************************
003000  000004                  TST7:   SCOPE
003002  012700  001000                  MOV     #BIT9,R0        ;STALL TIME COUNTER
003006  005277  176304                  INC     @STREG          ;START CONVERSION
003012  012737  000200  001124          MOV     #BIT7,$GDDAT    ;LOAD EXPECTED
003020  005300                  1$:     DEC     R0              ;STALL
003022  001376                          BNE     1$              ;TIME
003024  042777  100000  176264          BIC     #BIT15,@STREG   ;MASK OUT ERROR BIT
003032  004737  003662                  JSR     PC,TEST
003036  104001                          ERROR   1               ;A/D DONE FLAG FAILED TO SET;BITO FAILED TO CLEAR
003040  017700  176256                  MOV     @ADBUFF,R0      ;CLEAR DONE FLAG FOR ITERATIONS

                                        ;;**************************************************************
                                        ;*TEST 10        TEST INIT CLEARS DONE FLAG
                                        ;;**************************************************************
003044  000004                  TST10:  SCOPE
003046  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
003054  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
003060  005277  176232                  INC     @STREG          ;START CONVERSION
003064  105777  176226          2$:     TSTB    @STREG
003070  100375                          BPL     2$
003072  000005                          RESET
003074  004737  003662                  JSR     PC,TEST
003100  104001                          ERROR   1               ;DONE FLAG FAILED TO CLEAR
003102  052777  000100  176034          BIS     #100,@STKS              ;SET INTRPT. EN. BIT

                                        ;;**************************************************************
                                        ;*TEST 11        TEST A/D DONE FLAG CLEARS WHEN READ CONVERTED VALUE
                                        ;;**************************************************************
003110  000004                  TST11:  SCOPE
003112  012777  000001  176176          MOV     #BITO,@STREG    ;SET A/D START CONVERSION BIT
003120  105777  176172          1$:     TSTB    @STREG          ;WAIT FOR FLAG
003124  100375                          BPL     1$
003126  017700  176170                  MOV     @ADBUFF,R0      ;READ CONVERTED VALUE
003132  004737  003662                  JSR     PC,TEST
003136  104001                          ERROR   1               ;DONE FLAG FAILED TO CLEAR
```

```
1023                                    ;;*****************************************************************
1024                                    ;*TEST 12       GENERATE INTERRUPT WHEN DONE FLAG SETS AFTER CONVERSION
1025                                    ;;*****************************************************************
1026  003140  000004                    TST12:  SCOPE
1027                                            ;* "ENTERING TEST 12" TYPED OUT TO TELL YOU THE NEXT
1028                                            ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
1029                                            ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
1030                                            ;*EXECUTING TEST "12".
1031  003142  012700  000012                    MOV     #12,R0          ;GET TEST NO.
1032  003146  004737  011152                    JSR     PC,DUMW         ;PRINT MESSAGE
1033  003152  005037  177776                    CLR     PSW             ;RESET PRIORITY
1034  003156  012777  003224  176140            MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
1035  003164  012777  000101  176124            MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1036  003172  105777  176120          2$:       TSTB    @STREG          ;WAIT FOR DONE
1037  003176  100375                            BPL     2$              ;FLAG TO SET
1038  003200  017737  176112  001126            MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
1039  003206  012737  000300  001124            MOV     #BIT7!BIT6,$GDDAT               ;GOOD DATA
1040  003214  104002                            ERROR   2               ;FAILED TO INTERRUPT ON DONE
1041  003216  004737  011224                    JSR     PC,DUMC         ;TYPE COMPLETED
1042  003222  000403                            BR      TST13           ;;BRANCH TO NEXT TEST
1043  003224  022626                  1$:       CMP     (SP)+,(SP)+             ;RESET STACK POINTER
1044  003226  004737  011224                    JSR     PC,DUMC         ;TYPE COMPLETED
1045
1046                                    ;;*****************************************************************
1047                                    ;*TEST 13       TEST FOR CORRECTNESS OF PRIORITY OF INTERRUPT
1048                                    ;;*****************************************************************
1049  003232  000004                    TST13:  SCOPE
1050                                            ;* "ENTERING TEST 13" TYPED OUT TO TELL YOU THE NEXT
1051                                            ;*TEST THAT IS GOING TO BE EXECUTED. IT IS ONLY TYPED ON PASS 0.
1052                                            ;*THERE IS DANGER THAT THE UNIBUS COULD GET "HUNG" WHILE
1053                                            ;*EXECUTING TEST "13".
1054  003234  0 2700  000013                    MOV     #13,R0          ;GET TEST NO.
1055  003240  004737  011152                    JSR     PC,DUMW         ;PRINT MESSAGE
1056  003244  013737  001326  177776            MOV     BASEBR,PSW      ;GET ONE LEVEL LESS
1057  003252  162737  000040  177776            SUB     #40,PSW                 ;PRIORITY
1058  003260  012777  003314  176036            MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
1059  003266  012777  000101  176022            MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1060  003274  105777  176016          2$:       TSTB    @STREG          ;WAIT FOR
1061  003300  100375                            BPL     2$              ;DONE FLAG
1062  003302  017737  176010  001126            MOV     @STREG,$BDDAT           ;READ ST. REG.
1063  003310  104002                            ERROR   2               ;INTERRUPT NOT SERVICED AT LEVEL 5
1064  003312  000401                            BR      3$
1065  003314  022626                  1$:       CMP     (SP)+,(SP)+             ;RESET STACK POINTER
1066  003316  012777  003376  176000  3$:       MOV     #4$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
1067  003324  013737  001326  177776            MOV     BASEBR,PSW              ;SET PRIORITY TO 6
1068  003332  012777  000101  175756            MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE + START CONVERSION
1069  003340  105777  175752          5$:       TSTB    @STREG          ;WAIT FOR
1070  003344  100375                            BPL     5$              ;DONE FLAG
1071  003346  012777  001432  175750  6$:       MOV     #UNEXP,@VECTOR          ;UNEXPECTED INTERRUPT VECTOR
1072  003354  005077  175736                    CLR     @STREG          ;CLEAR BIT 6
1073  003360  005037  177776                    CLR     PSW             ;RESET PRIORITY
1074  003364  017700  175732                    MOV     @ADBUFF,R0              ;CLEAR DONE FLAG
1075  003370  004737  011224                    JSR     PC,DUMC         ;TYPE COMPLETED
1076  003374  000403                            BR      TST14           ;;BRANCH TO NEXT TEST
1077  003376  022626                  4$:       CMP     (SP)+,(SP)+             ;RESET STACK POINTER
1078  003400  104003                            ERROR   3               ;INTERRUPTED AT LEVEL 6 - WRONG
```

```
1079  003402  000761                          BR      6$
1080
1081                                  ;;*****************************************************************
1082                                  ;*TEST 14       TEST DONE FLAG CLEARS AFTER INTERRUPT
1083                                  ;;*****************************************************************
1084  003404  000004                  TST14:  SCOPE
1085  003406  012737  000300  001160          MOV     #300,$TIMES     ;;DO 300 ITERATIONS
1086  003414  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD GOOD DATA
1087  003422  012777  003456  175674          MOV     #1$,@VECTOR     ;INTERRUPT VECTOR ADDRESS
1088  003430  012777  000101  175660          MOV     #BIT6!BIT0,@STREG ;SET INTERRUPT ENABLE BIT + START CONVERSION
1089  003436  105777  175654          2$:     TSTB    @STREG          ;WAIT FOR
1090  003442  100375                          BPL     2$              ;DONE FLAG
1091  003444  017737  175646  001126          MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
1092  003452  104002                          ERROR   2               ;FAILED TO GENERATE INTERRUPT
1093  003454  000412                          BR      TST15           ;;GO TO NEXT TEST
1094  003456  022626                  1$:     CMP     (SP)+,(SP)+             ;RESET STACK POINTER
1095  003460  004737  003662                  JSR     PC,TEST
1096  003464  104001                          ERROR   1               ;DONE FLAG NOT CLEARED AFTER AN INTERRUPT
1097  003466  017737  175630  001352          MOV     @ADBUFF,TEMP            ;DUMMY READ BUFFER
1098  003474  012777  001432  175622          MOV     #UNEXP,@VECTOR          ;LOAD INTRPT. ADDRESS
1099
1100                                  ;;*****************************************************************
1101                                  ;*TEST 15       TEST ERROR FLAG SETS IF  2ND CONVERSION ENDS BEFORE READING BUFFER
1102                                  ;;*****************************************************************
1103  003502  000004                  TST15:  SCOPE
1104  003504  012777  000001  175604          MOV     #BIT0,@STREG    ;START CONVERSION
1105  003512  105777  175600          1$:     TSTB    @STREG          ;WAIT FOR
1106  003516  100375                          BPL     1$
1107  003520  012737  100200  001124  2$:     MOV     #BIT15!BIT7,$GDDAT ;LOAD EXPECTED VALUE
1108  003526  012777  000001  175562          MOV     #BIT0,@STREG    ;START 2ND CONVERSION
1109  003534  012700  001000                  MOV     #BIT9,R0        ;WAIT FOR 2ND
1110  003540  005300                  3$:     DEC     R0              ;CONVERSION TO END
1111  003542  001376                          BNE     3$
1112  003544  004737  003662          4$:     JSR     PC,TEST
1113  003550  104001                          ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
1114                                                                  ; CONVERT ENDS BEFORE READ BUFFER FROM FIRST
1115  003552  017700  175544                  MOV     @ADBUFF,R0      ;CLEAR DONE FLAG
```

```
1116                                    ;;*****************************************************************
1117                                    ;*TEST 16       TEST ERROR FLAG SETS IF START 2ND CONV. BEFORE DONE FLAG SETS
1118                                    ;;*****************************************************************
1119  003556  000004                    TST16:  SCOPE
1120  003560  012737  100000  001124            MOV     #BIT15,$GDDAT   ;LOAD EXPECTED DATA
1121  003566  012700  000100                    MOV     #100,R0         ;STALL TIME COUNTER
1122  003572  012777  000001  175516            MOV     #BIT0,@STREG    ;START CONVERSION
1123  003600  112777  000001  175510            MOVB    #BIT0,@STREG            ;START NEXT CONVERSION
1124  003606  005300                    DEC5:   DEC     R0              ;STALL TIME
1125  003610  001376                            BNE     DEC5
1126  003612  017737  175500  001126            MOV     @STREG,$BDDAT           ;READ STATUS REGISTER
1127  003620  042737  077777  001126            BIC     #77777,$BDDAT           ;MASK OUT BIT 15
1128  003626  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1129  003634  001401                            BEQ     1$              ;;BRANCH OVER ERROR
1130  003636  104001                            ERROR   1               ;ERROR FLAG NOT SET WHEN 2ND
1131                                            ;CONVERT BEGINS BEFORE FIRST DONE
1132  003640  017700  175456           1$:     MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
1133  003644  005077  175446                    CLR     @STREG                  ;CLEAR STATUS REGISTER
1134  003650  000004                            SCOPE
1135  003652  000207                            RTS     PC              ;RETURN TO TEST SECTION
1136
1137
1138                                    ;;SUBROUTINE FOR LOGIC TESTS;;
1139  003654  013777  001124  175434    TESTIT: MOV     $GDDAT,@STREG           ;LOAD EXPECTED VALUE
1140  003662  017737  175430  001126    TEST:   MOV     @STREG,$BDDAT           ;READ ST. REG.
1141  003670  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1142  003676  001002                            BNE     RETERR          ;;ERROR RETURN
1143  003700  062716  000002                    ADD     #2,(SP)                 ;BUMP RETURN ADDRESS TO GET AROUND ERROR
1144  003704  000207                    RETERR: RTS     PC
```

```
1145                                            .SBTTL      WRAPAROUND TEST SECTION
1146  003706                                    WRAP:
1147                                            ;;*****************************************************************
1148                                            ;*TEST 17      TEST CH14 GROUND
1149                                            ;;*****************************************************************
1150  003706  000240                            TST17:  NOP
1151  003710  012737  000010  001160                    MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1152  003716  012737  000017  001102                    MOV     #STN-1,$TSTNM
1153  003724  012737  003706  001110                    MOV     #TST17,$LPERR
1154  003732  012737  003706  001106                    MOV     #TST17,$LPADR
1155  003740  004537  010772                          JSR     R5,CONVRT               ;DO 8 CONVERSIONS
1156  003744  000014                                  14
1157  003746  004537  011104                          JSR     R5,COMPAR               ;COMPARE RESULTS
1158  003752  004000                                  4000                    ;NOMINAL
1159  003754  011616                                  V5D                     ;TOLERANCE
1160  003756  104004                                  ERROR   4               ;ERROR-CH14 NOT GROUND-AD11K MUST BE IN
1161                                                                          ;SINGLE-ENDED CONFIGURATION,G5036 WRAPAROUND
1162                                                                          ;MODULE MUST BE PRESENT,CHECK CONNECTION A-VV,VV-A
1163
1164
1165                                            ;;*****************************************************************
1166                                            ;*TEST 20      TEST CONVERSION FROM EXT. START
1167                                            ;;*****************************************************************
1168  003760  000004                            TST20:  SCOPE
1169  003762  012737  000010  001160                    MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1170  003770  012777  000020  175320                    MOV     #BIT4,@STREG            ;ENABLE EXTERNAL START
1171  003776  012700  001000                          MOV     #BIT9,R0                ;TIME DELAY COUNTER
1172  004002  012737  000220  001124                    MOV     #BIT7!BIT4,$GDCAT       ;LOAD EXPECTED
1173  004010  012777  000200  175304                    MOV     #200,@ADBUFF            ;SIMULATE EXT. START WITH G5036
1174                                                                          ;WRAPAROUND MODULE PRESENT
1175  004016  005300                          1$:     DEC     R0
1176  004020  001376                                  BNE     1$
1177  004022  004737  003662                          JSR     PC,TEST
1178  004026  104001                                  ERROR   1                       ;FAILED TO DO CONVERSION FROM EXT. START
1179  004030  017700  175266                          MOV     @ADBUFF,R0
1180  004034  005077  175256                          CLR     @STREG                  ;CLEAR ST. REG.
1181
1182
1183                                            ;;*****************************************************************
1184                                            ;*TEST 21      TEST CH0 GROUND
1185                                            ;;*****************************************************************
1186  004040  000004                            TST21:  SCOPE
1187  004042  012737  000010  001160                    MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1188  004050  004537  010772                          JSR     R5,CONVRT               ;CONVERT 8 TIMES
1189  004054  000000                                  0
1190  004056  004537  011104                          JSR     R5,COMPAR               ;COMPARE RESULTS
1191  004062  004000                                  4000                    ;NOMINAL
1192  004064  011610                                  V1                      ;TOLERANCE
1193  004066  104004                                  ERROR   4               ;ERROR ON A/D CHANNEL
```

```
1194                                    ;;**********************************************************************
1195                                    ;*TEST 22        TEST CH1 GROUND
1196                                    ;;**********************************************************************
1197  004070  000004                    TST22:  SCOPE
1198  004072  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1199  004100  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1200  004104  000001                            1                       ;CHANNEL 1
1201  004106  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1202  004112  004000                            4000                    ;NOMINAL
1203  004114  011612                            V2                      ;TOLERANCE
1204  004116  104004                            ERROR   4               ;ERROR ON A/D CHANNEL
1205
1206                                    ;;**********************************************************************
1207                                    ;*TEST 23        TEST CH2 +1 VOLT
1208                                    ;;**********************************************************************
1209  004120  000004                    TST23:  SCOPE
1210  004122  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1211  004130  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1212  004134  000002                            2                       ;CHANNEL 2
1213  004136  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1214  004142  004632                            4632                    ;NOMINAL
1215  004144  011616                            V50                     ;TOLERANCE
1216  004146  104004                            ERROR   4               ;ERROR ON A/D CHANNEL
1217                                                                    ;AD11K MUST BE SET UP FOR +OR- 5V OR +OR- 5.12V
1218
1219                                    ;;**********************************************************************
1220                                    ;*TEST 24        TEST CH3 +2.5 VOLTS
1221                                    ;;**********************************************************************
1222  004150  000004                    TST24:  SCOPE
1223  004152  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1224  004160  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1225  004164  000003                            3                       ;CHANNEL 3
1226  004166  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1227  004172  006000                            6000                    ;NOMINAL
1228  004174  011624                            V240                    ;TOLERANCE
1229  004176  104004                            ERROR   4               ;ERROR ON A/D CHANNEL
1230
1231                                    ;;**********************************************************************
1232                                    ;*TEST 25        TEST CH4 -2.5 VOLTS
1233                                    ;;**********************************************************************
1234  004200  000004                    TST25:  SCOPE
1235  004202  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1236  004210  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1237  004214  000004                            4                       ;CHANNEL 4
1238  004216  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1239  004222  002000                            2000                    ;NOMINAL
1240  004224  011624                            V240                    ;TOLERANCE
1241  004226  104004                            ERROR   4
```

```
1242                                    ;;**********************************************************************
1243                                    ;*TEST 26        TEST VERNIER OFFSET DAC ON CH12
1244                                    ;;**********************************************************************
1245  004230  000004                    TST26:  SCOPE
1246  004232  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1247  004240  005077  175056                    CLR     @ADBUFF                 ;VERNIER DAC=0
1248  004244  004737  005026                    JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
1249  004250  004537  010772                    JSR     R5,CONVRT               ;CONV. CH12, DIRECT VERNIER DAC
1250  004254  000012                            12
1251  004256  013704  001352                    MOV     TEMP,R4                 ;SAVE VALUE IN R4
1252  004262  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1253  004266  002376                            2376                            ;WITH -1.875 VOLTS
1254  004270  011622                            V11S                            ;TOLERANCE OF 10%
1255  004272  104004                            ERROR   4
1256  004274  005037  001424                    CLR     MAX
1257  004300  012702  000001                    MOV     #1,R2
1258  004304  010277  175012            1$:     MOV     R2,@ADBUFF              ;SET UP NEXT VERNIER DAC VALUE
1259  004310  004737  005026                    JSR     PC,DAWAIT               ;DELAY FOR DAC SETTLING
1260  004314  004537  010772                    JSR     R5,CONVRT               ;CONVERT IT
1261  004320  000012                            12
1262  004322  005737  001424                    TST     MAX
1263  004326  001010                            BNE     2$
1264  004330  023727  001352  004000            CMP     TEMP,#4000
1265  004336  002404                            BLT     2$
1266  004340  005237  001424                    INC     MAX
1267  004344  010237  001420                    MOV     R2,MIN
1268  004350  020227  000200            2$:     CMP     R2,#200
1269  004354  001003                            BNE     3$
1270  004356  013737  001352  004446            MOV     TEMP,4$
1271  004364  013703  001352            3$:     MOV     TEMP,R3                 ;SAVE VALUE
1272  004370  160437  001352                    SUB     R4,TEMP                 ;TEMP=DIFF. BETWEEN VALUE&PREVIOUS
1273  004374  010304                            MOV     R3,R4                   ;SET UP PREVIOUS VALUE FOR NEXT TIME THRU
1274  004376  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1275  004402  000006                            6                               ;WITH 15 MILLIVOLTS(1 DAC LSB)
1276  004404  011626                            V5
1277  004406  104004                            ERROR   4
1278  004410  005202                            INC     R2
1279  004412  020227  000400                    CMP     R2,#400                 ;DONE?
1280  004416  001332                            BNE     1$                      ;NO-DO NEXT VERNIER DAC VALUE
1281  004420  000005                            RESET                           ;SHOULD SET VERNIER DAC TO 200
1282  004422  052777  000100  174514            BIS     #100,@$TKS
1283  004430  004737  005026                    JSR     PC,DAWAIT               ;LET DAC SETTLE
1284  004434  004537  010772                    JSR     R5,CONVRT               ;CONVERT IT
1285  004440  000012                            12
1286  004442  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1287  004446  000000            4$:     0
1288  004450  011612                            V2
1289  004452  104004                            ERROR   4
```

E03


```
1290                                    ;;*****************************************************************
1291                                    ;*TEST 27        TEST CH13 +2.5 VOLTS
1292                                    ;;*****************************************************************
1293  004454  000004                    TST27:  SCOPE
1294  004456  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1295  004464  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1296  004470  000013                            13
1297  004472  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1298  004476  006000                            6000                            ;NOMINAL
1299  004500  011620                            V144                            ;TOLERANCE
1300  004502  104004                            ERROR   4
1301                                    ;;*****************************************************************
1302                                    ;*TEST 30        TEST CH17 +4V
1303                                    ;;*****************************************************************
1304  004504  000004                    TST30:  SCOPE
1305  004506  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1306  004514  004537  010772                    JSR     R5,CONVRT               ;CONVERT 8 TIMES
1307  004520  000017                            17                              ;CHANNEL 17
1308  004522  004537  011104                    JSR     R5,COMPAR               ;COMPARE RESULTS
1309  004526  007146                            7146                            ;NOMINAL
1310  004530  011624                            V240                            ;TOLERANCE
1311  004532  104004                            ERROR   4                       ;ERROR ON A/D CHANNEL
```

```
1312                                    ;;**********************************************************************
1313                                    ;*TEST 31        OFFSET ON CH0
1314                                    ;;**********************************************************************
1315  004534  000004                    TST31:  SCOPE
1316  004536  012737  0C0001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1317  004544  013737  001336  001366            MOV     BASECH,CHANL            ;LOAD CHANNEL
1318  004552  013737  001336  001364            MOV     BASECH,DUMMY            ;LOAD DUMMY
1319  004560  012737  004001  001414            MOV     #4001,EDGE
1320  004566  004537  006504                    JSR     R5,SARSUB
1321  004572  000062                            50.
1322  004574  013737  001410  001352            MOV     DAC,TEMP
1323  004602  004537  006504                    JSR     R5,SARSUB
1324  004606  000062                            50.
1325  004610  063737  001410  001352            ADD     DAC,TEMP
1326  004616  162737  000062  001352            SUB     #62,TEMP
1327  004624  013700  001420                    MOV     MIN,R0
1328  004630  006300                            ASL     R0
1329  004632  160037  001352                    SUB     R0,TEMP
1330  004636  104400  013573                    TYPE    ,MOFSET         ;TYPE ASCIZ STRING
1331  004642  013702  001352                    MOV     TEMP,R2
1332  004646  004737  011374                    JSR     PC,DECTYP
1333  004652  104400  013606                    TYPE    ,MLSB           ;TYPE ASCIZ STRING
1334  004656  004537  011104                    JSR     R5,COMPAR       ;IS RESULT WITHIN LIMITS?
1335  004662  000000                            0
1336  004664  011630                            VSOD
1337  004666  000401                            BR      OFFERR          ;NO-ERROR
1338  004670  000403                            BR      OFFOK           ;YES-OK
1339  004672  104400  012375    OFFERR: TYPE    ,ERMSG
1340  004676  000402                            BR      TST32           ;;GO TO NEXT TEST
1341  004700  104400  012364    OFFOK:  TYPE    ,OKMSG
```

```
1342                                    ;;*****************************************************************
1343                                    ;*TEST 32       NOISE TEST ON 8 EDGES
1344                                    ;;*****************************************************************
1345  004704  000004                    TST32:  SCOPE
1346  004706  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1347  004714  012737  000116  001352            MOV     #116,TEMP               ;DAC VALUE
1348  004722  004537  010564                    JSR     R5,NOI8                 ;NOISE AT -FULL SCALE
1349  004726  000015                            15
1350  004730  004537  010564                    JSR     R5,NOI8                 ;NOISE AT MID-RANGE
1351  004734  000007                            7
1352  004736  004537  010564                    JSR     R5,NOI8                 ;NOISE AT +FULL SCALE
1353  004742  000016                            16
1354
1355                                    ;;*****************************************************************
1356                                    ;*TEST 33       SETTLE TEST ON 8 EDGES
1357                                    ;;*****************************************************************
1358  004744  000004                    TST33:  SCOPE
1359  004746  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1360  004754  004537  006220                    JSR     R5,SET8                 ;SETTLE-POSITIVE DIRECTION
1361  004760  000015                            15
1362  004762  000016                            16
1363  004764  012737  000116  001352            MOV     #116,TEMP
1364  004772  004537  006220                    JSR     R5,SET8                 ;SETTLE-NEGATIVE DIRECTION
1365  004776  000016                            16
1366  005000  000015                            15
1367                                    ;;*****************************************************************
1368                                    ;*TEST 34       DIFFERENTIAL LINEARITY AND RELATIVE ACCURACY TEST
1369                                    ;;*****************************************************************
1370  005002  000004                    TST34:  SCOPE
1371  005004  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
1372  005012  005737  001202                    TST     $PASS                   ;FIRST TIME-SKIP DIFLIN
1373  005016  001402                            BEQ     LEND
1374  005020  004737  006712                    JSR     PC,DIFLIN
1375  005024  000207                    LEND:   RTS     PC                      ;RETURN TO TEST SECTION
1376
1377  005026  005000                    DAWAIT: CLR     R0
1378  005030  105300                    1$:     DECB    R0
1379  005032  001376                            BNE     1$
1380  005034  000207                            RTS     PC
```

```
1381                                            .SBTTL      CALIBRATION TEST
1382  005036  012737  005036  001370  BEGINC: MOV     #BEGINC,TADDR           ;TEST ADDRESS IN TADDR
1383  005044  005077  174246                  CLR     @STREG                  ;CLEAR STATUS REGISTER
1384  005050  104400  013503                  TYPE    ,HEAD5                  ;TYPE OUT HEADING
1385  005054  005037  177776                  CLR     PSW
1386  005060  017700  174054          1$:     MOV     @SWR,R0                 ;READ CHANNEL FROM SWITCH REG.
1387  005064  042700  177700                  BIC     #177700,R0              ;ISOLATE MUX BITS
1388  005070  032777  020000  174042          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
1389  005076  001005                          BNE     2$              ;;YES,SKIP TYPEOUT
1390  005100  104400  012207                  TYPE    ,CH
1391  005104  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
1392                                                                  ;;TYPE CHANNEL
1393  005106  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1394  005110     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1395  005111     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1396  005112  012777  001610  174204  2$:     MOV     #RETURN,@VECTOR         ;ADDRESS AFTER INTRPT.
1397  005120  000300                          SWAB    R0                      ;SWITCH BYTES
1398  005122  052700  000100                  BIS     #BIT6,R0
1399  005126  010077  174164                  MOV     R0,@STREG               ;LOAD THE CHANNEL
1400  005132  012702  000010                  MOV     #10,R2                  ;TYPEOUT COUNTER
1401  005136  005277  174154          3$:     INC     @STREG                  ;START CONVERSION
1402  005142  000001                          WAIT                            ;WAIT FOR INTRPT.
1403  005144  017700  174152                  MOV     @ADBUFF,R0              ;READ CONVERTED VALUE
1404  005150  032777  020000  173762          BIT     #BIT13,@SWR             ;IS BIT 13 SET?
1405  005156  001403                          BEQ     4$                      ;NOT SET, TYPE OUT LIST
1406  005160  010077  173756                  MOV     R0,@DISPLAY                     ;PUT VALUE IN DISPLAY FOR DISPLAY CONTRO
1407  005164  000735                          BR      1$                      ;REPEAT CONVERSION
1408  005166  104400  012212          4$:     TYPE    ,SPACE
1409  005172  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
1410                                                                  ;;PRINT OCTAL CONVERTED VALUE
1411  005174  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1412  005176     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
1413  005177     001                          .BYTE   1               ;;TYPE LEADING ZEROS
1414  005200  012701  010000                  MOV     #10000,R1
1415  005204  005301                  5$:     DEC     R1
1416  005206  001376                          BNE     5$
1417  005210  005302                          DEC     R2                      ;DECREMENT THE COUNTER
1418  005212  001351                          BNE     3$                      ;NO CARRIAGE RETURN
1419  005214  104400  001171                  TYPE    ,$CRLF                  ;CARRIAGE RETURN
1420  005220  000717                          BR      1$                      ;REPEAT CONVERSION
```

```
1421                                            .SBTTL      LOGIC TEST SECTION
1422  005222  012737  005222  001370  BEGL:     MOV       #BEGL,TADDR           ;TEST ADDRESS
1423  005230  004737  002422                    JSR       PC,TESTAD             ;NO OF ADDITIONAL AD'S
1424  005234  004737  002556            1$:     JSR       PC,BEGINL             ;LOGIC TESTS
1425  005240  004737  005410                    JSR       PC,BUMPAD             ;MORE TO TEST?
1426  005244  000773                            BR        1$                    ;TEST NEXT A/D
1427  005246  012737  005234  011706            MOV       #1$,AGTST             ;ADDRESS FOR EOP
1428  005254  000137  011710                    JMP       $EOP                  ;TYPE END OF PASS
1429
1430                                            .SBTTL      AUTO TEST
1431  005260  012737  005260  001370  BEGINA:   MOV       #BEGINA,TADDR         ;TEST ADDRESS
1432  005266  005037  001202                    CLR       $PASS                 ;CLEAR PASS COUNTER
1433  005272  004737  002422                    JSR       PC,TESTAD             ;NO. OF AD'S TO BE TESTED
1434  005276  004737  002556            1$:     JSR       PC,BEGINL             ;LOGIC TESTS
1435  005302  104400  012665                    TYPE      ,MEND                 ;TYPE END OF LOGIC TEST
1436  005306  013746  001316                    MOV       $TREG,-(SP)           ;SAVE STREG FOR TYPEOUT
1437  005312  104402                            TYPOS                           ;TYPE OCTAL NUMBER
1438  005314     006                            .BYTE     6                     ;TYPE 6 DIGITS
1439  005315     001                            .BYTE     1                     ;TYPE LEADING ZEROS
1440  005316  104400  001171                    TYPE      ,$CRLF                ;TYPE A CR,LF
1441  005322  004737  003706                    JSR       PC,WRAP
1442  005326  004737  005410                    JSR       PC,BUMPAD             ;TEST NEXT A/D
1443  005332  000761                            BR        1$                    ;TEST NEXT AD
1444  005334  012737  005276  011706            MOV       #1$,AGTST             ;ADDRESS FOR EOP
1445  005342  000137  011710                    JMP       $EOP                  ;TYPE END OF PASS
1446
1447                                            .SBTTL      WRAPAROUND TEST
1448  005346  012737  005346  001370  BEGINW:   MOV       #BEGINW,TADDR         ;TEST ADDRESS
1449  005354  005037  001202                    CLR       $PASS                 ;CLEAR PASS COUNT
1450  005360  004737  002422                    JSR       PC,TESTAD             ;NO. OF AD'S TO BE TESTED
1451  005364  004737  003706            1$:     JSR       PC,WRAP               ;WRAPAROUND TESTS
1452  005370  004737  005410                    JSR       PC,BUMPAD             ;MORE A/D'S TO BE TESTED?
1453  005374  000773                            BR        1$                    ;YES-GO TEST NEXT AD11K
1454  005376  012737  005364  011706            MOV       #1$,AGTST
1455  005404  000137  011710                    JMP       $EOP                  ;INCREMENTS $PASS
```

```
1456                                            .SBTTL      DETERMINE IF MORE AD11K'S TO BE TESTED
1457  005410  005737  001360                BUMPAD: TST     NBEXT                   ;ADDITIONAL AD'S?
1458  005414  001424                              BEQ     FIXADR                  ;NO-INITIALIZE ADDRESSES
1459  005416  063737  001332  001316              ADD     VADR,STREG              ;SET UP NEW ST. REG.
1460  005424  063737  001332  001320              ADD     VADR,ADST1              ;SET UP NEW ADST1
1461  005432  063737  001332  001322              ADD     VADR,ADBUFF             ;SET UP NEW BUFFER ADDRESS
1462  005440  063737  001334  001324              ADD     VVCT,VECTOR             ;SET UP NEW VECTOR
1463  005446  063737  001334  001330              ADD     VVCT,VECTR1
1464  005454  005077  173650                      CLR     @VECTR1
1465  005460  005337  001360                      DEC     NBEXT                   ;ONE LESS AD11K
1466  005464  000446                              BR      BYPASS
1467  005466  062716  000002              FIXADR: ADD     #2,(SP)
1468  005472  013737  001250  001316      FIXONE: MOV     $BASE,STREG             ;RELOAD INITIAL ADDRESSES
1469  005500  013737  001250  001320              MOV     $BASE,ADST1
1470  005506  013737  001250  001322              MOV     $BASE,ADBUFF
1471  005514  005237  001320                      INC     ADST1
1472  005520  062737  000002  001322              ADD     #2,ADBUFF
1473  005526  013737  001244  001324              MOV     $VECT1,VECTOR
1474  005534  042737  170000  001324              BIC     #170000,VECTOR
1475  005542  113737  001245  001326              MOVB    $VECT1+1,BASEBR
1476  005550  105037  001327                      CLRB    BASEBR+1                ;CLEAR HIGH BYTE
1477  005554  013737  001324  001330              MOV     VECTOR,VECTR1
1478  005562  062737  000002  001330              ADD     #2,VECTR1
1479  005570  005077  173534                      CLR     @VECTR1
1480  005574  013737  001362  001360              MOV     NMBEXT,NBEXT            ;RESET COUNTER
1481                                      ;;LOAD .+2 AND HALT TRAP CATCH;;
1482  005602  012700  000216              BYPASS: MOV     #216,R0                 ;FILL .+2
1483  005606  012701  000214                      MOV     #214,R1                 ;LOAD HALT
1484  005612  020137  001340              1$:     CMP     R1,KBVECT
1485  005616  001410                              BEQ     2$
1486  005620  010021                              MOV     R0,(R1)+
1487  005622  005021                              CLR     (R1)+
1488  005624  010100                              MOV     R1,R0
1489  005626  005720                              TST     (R0)+
1490  005630  020027  001002                      CMP     R0,#1002
1491  005634  001366                              BNE     1$
1492  005636  000207                              RTS     PC                      ;TEST NEXT A/D
1493  005640  022021                      2$:     CMP     (R0)+,(R1)+
1494  005642  022021                              CMP     (R0)+,(R1)+
1495  005644  000762                              BR      1$
1496
1497
1498                                            .SBTTL      NOISE TEST, 1 EDGE
1499  005646  012737  005646  001370      BEGINN: MOV     #BEGINN,TADDR           ;TEST ADDRESS IN TADDR
1500  005654  104400  012016                      TYPE    ,NOIMSG                 ;ASK FOR CHANNEL
1501  005660  104400  013522                      TYPE    ,ASKCH
1502  005664  017737  173250  001354      1$:     MOV     @SWR,CH1                ;LOAD CHANNEL
1503  005672  042737  177700  001354              BIC     #177700,CH1
1504  005700  012737  000200  001352              MOV     #200,TEMP               ;LOAD DAC VALUE
1505  005706  004537  010300                      JSR     R5,NOITST               ;GO TO NOISE SUBROUTINE
1506  005712  001354                              CH1
1507  005714  000763                              BR      1$
```

```
                                        .SBTTL      INTERCHANNEL SETTLING TEST, 1 EDGE
005716  012737  005716  001370  BEGINS: MOV     #BEGINS,TADDR           ;TEST ADDRESS IN TADDR
005724  104400  012036                  TYPE    ,SETMSG                 ;ASK FOR CHANNELS
005730  104406                          RDOCT
005732  012637  001354                  MOV     (SP)+,CH1
005736  104400  012323                  TYPE    ,TOMSG
005742  104406                          RDOCT
005744  012637  001356                  MOV     (SP)+,CH2
005750  012737  000200  001352  BK3:    MOV     #200,TEMP               ;LOAD DAC
005756  013737  001356  001366          MOV     CH2,CHANL
005764  004737  006324                  JSR     PC,GETEDG               ;GET EDGE VALUES
005770  005002                          CLR     R2
005772  004737  006156                  JSR     PC,SET1A                ;SCALING = .02 LSB
005776  004737  006156                  JSR     PC,SET1A                ;MAKE IT .01 LSB
006002  100001                          BPL     POSR2
006004  005402                          NEG     R2
006006  010204                  POSR2:  MOV     R2,R4
006010  012737  000001  006502          MOV     #1,EDGFLG
006016  004737  006024                  JSR     PC,TYPSET
006022  000752                          BR      BK3
006024  004737  011374          TYPSET: JSR     PC,DECTYP
006030  104400  012217                  TYPE    ,LSB
006034  013746  001356                  MOV     CH2,-(SP)       ;;SAVE CH2 FOR TYPEOUT
                                                                ;;TYPE CH
006040  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006042     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
006043     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
006044  104400  013614                  TYPE    ,MAT            ;TYPE ASCIZ STRING
006050  004737  006440                  JSR     PC,TYPEDG
006054  104400  012232                  TYPE    ,SETCH
006060  013746  001354                  MOV     CH1,-(SP)       ;;SAVE CH1 FOR TYPEOUT
                                                                ;;TYPE CH
006064  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006066     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
006067     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
006070  104400  012254                  TYPE    ,ATMSG
006074  013737  001354  006122          MOV     CH1,1$
006102  163737  001336  006122          SUB     BASECH,1$
006110  012777  000200  173204          MOV     #200,@ADBUFF
006116  004537  010772                  JSR     R5,CONVRT
006122  000000                  1$:     0
006124  013746  001352                  MOV     TEMP,-(SP)      ;;SAVE TEMP FOR TYPEOUT
                                                                ;;TYPE VALUE
006130  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006132     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006133     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006134  020437  011636                  CMP     R4,VSET
006140  003003                          BGT     ERR
006142  104400  012364                  TYPE    ,OKMSG
006146  000207                          RTS     PC
```

```
1558  006150  104400  012375          ERR:    TYPE    ERMSG
1559  006154  000207                  RTS     PC
1560
1561
1562
1563                                  ;;SUBROUTINE FOR SETTLING TESTS;;
1564  006156  013737  001356  001364  SET1A:  MOV     CH2,DUMMY               ;LOAD DUMMY
1565  006164  004537  006504          JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
1566  006170  000062                  50.
1567  006172  063702  001410          ADD     DAC,R2          ;ADD RESULT TO R2
1568  006176  013737  001354  001364  MOV     CH1,DUMMY               ;CHANGE DUMMY VALUE
1569  006204  004537  006504          JSR     R5,SARSUB               ;DO SAR ROUTINE AT 50%
1570  006210  000062                  50.
1571  006212  163702  001410          SUB     DAC,R2          ;SUBTRACT RESULT FROM R2
1572  006216  000207                  RTS     PC              ;RETURN
1573
1574  006220  012537  001354  SET8:   MOV     (R5)+,CH1               ;GET FIRST CHANNEL
1575  006224  012537  001356          MOV     (R5)+,CH2               ;GET SECOND CHANNEL
1576  006230  063737  001336  001354  ADD     BASECH,CH1
1577  006236  063737  001336  001356  ADD     BASECH,CH2
1578  006244  004737  006324          JSR     PC,GETEDG               ;GET EDGE VALUES
1579  006250  005002                  CLR     R2
1580  006252  012703  000010          MOV     #10,R3          ;SET UP COUNTER
1581  006256  004737  006156  SETAA:  JSR     PC,SET1A                ;GET SETTLE VALUES
1582  006262  005237  001414          INC     EDGE
1583  006266  005303                  DEC     R3
1584  006270  001372                  BNE     SETAA           ;REPEAT 8 TIMES
1585  006272  162737  000010  001414  SUB     #10,EDGE
1586  006300  005702                  TST     R2
1587  006302  100001                  BPL     R2POS
1588  006304  005402                  NEG     R2
1589  006306  010204          R2POS:  MOV     R2,R4
1590  006310  012737  000010  006502  MOV     #8.,EDGFLG
1591  006316  004737  006024          JSR     PC,TYPSET               ;TYPE OUT RESULTS
1592  006322  000205                  RTS     R5                      ;RETURN
```

```
                                        ;SUBROUTINE TO GET EDGE VALUE
                                        ;CALL=JSR   PC,GETEDG
                                        ;CONVERSIONS ON A/D CHANNEL 'CHANL'
                                        ;RESULT IN EDGE, USES R0
006324  013777  001352  172770  GETEDG: MOV     TEMP,@ADBUFF
006332  113700  001366                  MOVB    CHANL,R0                ;GET CHANNEL
006336  000300                          SWAB    R0              ;SET UP A.D STATUS REG.
006340  052700  000100                  BIS     #100,R0         ;ENABLE INTRPT.
006344  010077  172746                  MOV     R0,@STREG
006350  012700  000100                  MOV     #100,R0         ;DAC SETTLING DELAY
006354  005300                  1$:     DEC     R0
006356  001376                          BNE     1$
006360  005037  001414                  CLR     EDGE
006364  012700  000010                  MOV     #10,R0
006370  012777  001610  172726          MOV     #RETURN,@VECTOR         ;RETURN ADDRESS
006376  005277  172714          CONV:   INC     @STREG                  ;START CONVERSION
006402  000001                          WAIT                            ;WAIT FOR INTERRUPT
006404  067737  172712  001414          ADD     @ADBUFF,EDGE
006412  005300                          DEC     R0
006414  001370                          BNE     CONV
006416  006237  001414                  ASR     EDGE
006422  006237  001414                  ASR     EDGE
006426  006237  001414                  ASR     EDGE
006432  005537  001414                  ADC     EDGE
006436  000207                          RTS     PC

                                        ;;SUBROUTINE TO TYPE EDGE VALUES;;
006440  013703  001414          TYPEDG: MOV     EDGE,R3
006444  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE OCTAL VALUE OF EDGE
006446  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006450     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006451     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006452  023727  006502  000001          CMP     EDGFLG,#1
006460  001407                          BEQ     RET
006462  062703  000007                  ADD     #7,R3
006466  104400  013564                  TYPE    ,C1             ;TYPE ASCIZ STRING
006472  010346                          MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
                                                                ;;TYPE EDGE VALUE
006474  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
006476     004                          .BYTE   4               ;;TYPE 4 DIGIT(S)
006477     001                          .BYTE   1               ;;TYPE LEADING ZEROS
006500  000207          RET:    RTS     PC
006502  000000          EDGFLG: 0
```

```
                                        ;SUBROUTINE TO DO SUCCESSIVE APPROXIMATION ROUTINE
                                        ;CALL=JSR   R5,SARSUB
                                        ;   XXX;XXX=PERCENT
                                        ;RESULT RETURNED IN 'DAC',USES R0,R1,R4
006504 012537 001426          SARSUB:   MOV     (R5)+,PERCNT          ;GET PERCENT
006510 006337 001426                    ASL     PERCNT
006514 006337 001426                    ASL     PERCNT
006520 006337 001426                    ASL     PERCNT                ;RESCALE PERCENT FOR 1600.
006524 006337 001426                    ASL     PERCNT                ;POINTS PER BURST
006530 012737 000200 001416   SAR1:     MOV     #200,BITPNT           ;INITIALIZE BIT POINTER AT MSB
006536 005037 001410                    CLR     DAC                   ;INITIALIZE DAC VALUE
006542 005000                 TRY:      CLR     R0
006544 063737 001416 001410             ADD     BITPNT,DAC            ;TRY BIT
006552 013777 001410 172542             MOV     DAC,@ADBUFF
006560 012737 000100 001412             MOV     #100,DELAY
006566 005337 001412          1$:       DEC     DELAY                 ;STALL TIME
006572 001375                           BNE     1$
006574 012701 003100                    MOV     #1600.,R1             ;SET UP FOR 1600. CONVERSIONS
006600 113777 001364 172512   NXTCVT:   MOVB    DUMMY,@ADST1          ;PRESET MUX TO DUMMY CHANNEL
006606 012777 001610 172510             MOV     #RETURN,@VECTOR       ;RETURN ADDRESS
006614 052777 000101 172474             BIS     #101,@STREG           ;CONVERSION ON DUMMY CHANNEL
006622 000001                           WAIT                          ;WAIT FOR INTERRUPT
006624 017704 172472                    MOV     @ADBUFF,R4            ;DUMMY READ
006630 013704 001366                    MOV     CHANL,R4
006634 000304                           SWAB    R4
006636 052704 000101                    BIS     #101,R4               ;INTERRUPT ENABLE START
006642 010477 172450                    MOV     R4,@STREG             ;JUMP TO CHANNEL + START CONVERT
006646 000001                           WAIT                          ;WAIT FOR INTERRUPT
006650 027737 172446 001414             CMP     @ADBUFF,EDGE
006656 002001                           BGE     2$
006660 005200                           INC     R0                    ;COUNT RESULTS .LT. EDGE
006662 005301                 2$:       DEC     R1
006664 001345                           BNE     NXTCVT
006666 020037 001426                    CMP     R0,PERCNT
006672 003003                           BGT     SHIFT
006674 163737 001416 001410             SUB     BITPNT,DAC            ;TAKE THE BIT OUT
006702 006237 001416          SHIFT:    ASR     BITPNT
006706 001315                           BNE     TRY
006710 000205                           RTS     R5
```

```
                                        ;;DIFFERENTIAL LINEARITY SUBROUTINE;;
006712  104400  013010        DIFLIN: TYPE    ,MSG20
006716  005037  001430                CLR     OUT
006722  012700  017744                MOV     #BUFFER,R0
006726  012701  010000                MOV     #4096.,R1       ;4096 WORDS FOR HISTOGRAM
006732  005020                CLEAR1: CLR     (R0)+           ;CLEAR BUFFER AREA
006734  005301                        DEC     R1
006736  001375                        BNE     CLEAR1
006740  012700  017124                MOV     #DIST,R0        ;DISTRIBUTION BUFFER POINTER
006744  012701  000310                MOV     #200.,R1        ;200. WORDS FOR DISTRIBUTION
006750  005003                        CLR     R3
006752  005037  001430                CLR     OUT
006756  005037  001342                CLR     WIDE
006762  005037  001344                CLR     NARROW
006766  005037  001346                CLR     FIRST
006772  005037  001350                CLR     SKIPST
006776  005020                CLEAR2: CLR     (R0)+           ;CLEAR DISTRIBUTION BUFFER AREA
007000  005301                        DEC     R1
007002  001375                        BNE     CLEAR2
007004  012700  000011        CHANNL: MOV     #11,R0          ;CHANNEL 11
007010  063700  001336                ADD     BASECH,R0
007014  000300                        SWAB    R0                              ;LOAD MUX BITS
007016  052700  000100                BIS     #100,R0
007022  010077  172270                MOV     R0,@STREG
007026  012700  001440                MOV     #800.,R0        ;NOMINAL STATE WIDTH - 1 LSB
007032  012777  001610  172264        MOV     #RETURN,@VECTOR
007040  012701  007776        AGAIN:  MOV     #4094.,R1
007044  004737  010710        NEXT:   JSR     PC,RANDY        ;GET RANDOM NUMBER
007050  013702  001372                MOV     RNA,R2
007054  042702  177760                BIC     #177760,R2      ;MASK IT TO 4 BITS ONLY
007060  001402                        BEQ     CONVR
007062  005302                DELAY3: DEC     R2              ;STALL
007064  001376                        BNE     DELAY3          ;TIME
007066  005277  172224        CONVR:  INC     @STREG          ;START CONVERSION
007072  000001                        WAIT
007074  017702  172222                MOV     @ADBUFF,R2      ;GET CONVERTED VALUE
007100  001413                        BEQ     DELAY1          ;IGNORE IF =0
007102  020227  007777                CMP     R2,#7777        ;IGNORE IF =7777
007106  001413                        BEQ     DELAY2
007110  006302                        ASL     R2
007112  005262  017744                INC     BUFFER(R2)      ;MAKE HISTOGRAM
007116  100013                        BPL     OKAY
007120  012762  077777  017744        MOV     #077777,BUFFER(R2)      ;PREVENT OVERFLOW
007126  000407                        BR      OKAY
007130  020227  007777        DELAY1: CMP     R2,#7777        ;EQUALIZE LOOP TIME
007134  001400                        BEQ     DELAY2          ;WITH DUMMY INSTR.
007136  005201                DELAY2: INC     R1
007140  005263  001352                INC     TEMP(R3)
007144  100403                        BMI     NOTOK
007146  005301                OKAY:   DEC     R1
007150  001335                        BNE     NEXT
007152  000403                        BR      AROUND
007154  005037  001352        NOTOK:  CLR     TEMP
007160  000772                        BR      OKAY
007162  005300                AROUND: DEC     R0
007164  001325                        BNE     AGAIN
```

```
 1732  007166  012700  007776                    MOV     #4094.,R0
 1733  007172  012701  017746                    MOV     #BUFFER+2,R1
 1734  007176  012102                    READ:   MOV     (R1)+,R2        ;GET STATE WIDTH
 1735  007200  006202                            ASR     R2              ;1 LSB = 800.
 1736  007202  006202                            ASR     R2
 1737  007204  006202                            ASR     R2
 1738  007206  005502                            ADC     R2              ;1 LSB = 100.
 1739  007210  020227  000310                    CMP     R2,#200.        ;OUT OF RANGE?
 1740  007214  002403                            BLT     INRNGE
 1741  007216  005237  001430                    INC     OUT             ;YES - INCREMENT COUNTER
 1742  007222  000423                            BR      TYPBAD
 1743  007224  006302                    INRNGE: ASL     R2
 1744  007226  005262  017124                    INC     DIST(R2)        ;MAKE STATE WIDTH DISTRIBUTION
 1745  007232  006202                            ASR     R2
 1746  007234  020227  000062                    CMP     R2,#50.         ;IS IT 1/2 LSB?
 1747  007240  002007                            BGE     NOTNAR
 1748  007242  005237  001344                    INC     NARROW
 1749  007246  005702                            TST     R2              ;IS IT A SKIPPED STATE?
 1750  007250  001002                            BNE     31$
 1751  007252  005237  001350                    INC     SKIPST
 1752  007256  000405                    31$:    BR      TYPBAD
 1753  007260  020227  000226            NOTNAR: CMP     R2,#150.        ;IS IT 1.5 LSB?
 1754  007264  003426                            BLE     LAST
 1755  007266  005237  001342                    INC     WIDE
 1756  007272  005737  001346            TYPBAD: TST     FIRST
 1757  007276  001004                            BNE     60$
 1758  007300  005237  001346                    INC     FIRST
 1759  007304  104400  012167                    TYPE    ,STATE
 1760  007310  010103                    60$:    MOV     R1,R3
 1761  007312  162703  017746                    SUB     #BUFFER+2,R3
 1762  007316  006203                            ASR     R3
 1763  007320  010346                            MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
 1764                                                                    ;;TYPE STATE
 1765  007322  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
 1766  007324     004                            .BYTE   4               ;;TYPE 4 DIGIT(S)
 1767  007325     001                            .BYTE   1               ;;TYPE LEADING ZEROS
 1768  007326  104400  012163                    TYPE    ,DASH
 1769  007332  004737  011374                    JSR     PC,DECTYP
 1770  007336  104400  012154                    TYPE    ,LSBMSG
 1771  007342  005300                    LAST:   DEC     R0
 1772  007344  001314                            BNE     READ
 1773  007346  112737  000177  014453            MOVB    #177,DECPNT
 1774  007354  013702  001350                    MOV     SKIPST,R2               ;GET NO. OF SKIPPED STATES
 1775  007360  004737  011374                    JSR     PC,DECTYP               ;TYPE IT
 1776  007364  104400  012412                    TYPE    ,SKPMSG         ;TYPE MESSAGE
 1777  007370  005737  001350                    TST     SKIPST
 1778  007374  001403                            BEQ     1$
 1779  007376  104400  012375                    TYPE    ,ERMSG          ;TYPE "ERROR"
 1780  007402  000402                            BR      NAR
 1781  007404  104400  012364            1$:     TYPE    ,OKMSG          ;TYPE #OK#
```

```
 1782  007410  013702  001344          NAR:    MOV     NARROW,R2               ;GET NO. OF NARROW STATES
 1783  007414  004737  011374                  JSR     PC,DECTYP               ;TYPE IT
 1784  007420  104400  012434                  TYPE    ,NARMSG         ;TYPE MESSAGE
 1785  007424  013702  001342                  MOV     WIDE,R2
 1786  007430  063702  001430                  ADD     OUT,R2
 1787  007434  004737  011374                  JSR     PC,DECTYP               ;TYPE NO. OF WIDE STATES
 1788  007440  104400  012473                  TYPE    ,WIDMSG         ;TYPE MESSAGE
 1789  007444  013702  001430                  MOV     OUT,R2
 1790  007450  004737  011374                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE 2 LSB
 1791  007454  104400  012532                  TYPE    ,OUTMSG         ;TYPE MESSAGE
 1792  007460  005737  001430                  TST     OUT
 1793  007464  001403                          BEQ     11$
 1794  007466  104400  012375                  TYPE    ,ERMSG          ;TYPE "ERROR"
 1795  007472  000402                          BR      HALF
 1796  007474  104400  012364          11$:    TYPE    ,OKMSG          ;TYPE "OK"
 1797  007500  013702  001344          HALF:   MOV     NARROW,R2
 1798  007504  063702  001342                  ADD     WIDE,R2
 1799  007510  063702  001430                  ADD     OUT,R2
 1800  007514  010200                          MOV     R2,R0
 1801  007516  004737  011374                  JSR     PC,DECTYP               ;TYPE NO. OF STATES OUTSIDE LIMITS
 1802  007522  112737  000056  014453          MOVB    #56,DECPNT
 1803  007530  104400  012565                  TYPE    ,HAFMSG
 1804  007534  020027  000051                  CMP     R0,#41.         ;COMPARE IT TO NOMINAL
 1805  007540  003403                          BLE     21$
 1806  007542  104400  012375                  TYPE    ,ERMSG          ;TYPE "ERROR"
 1807  007546  000402                          BR      SWDIST
 1808  007550  104400  012364          21$:    TYPE    ,OKMSG          ;TYPE "OK"
 1809  007554  005737  001404          SWDIST: TST     FLAG            ;VT55?
 1810  007560  001426                          BEQ     RELACC
 1811  007562  004737  010242                  JSR     PC,DELCLR       ;WAIT AWHILE, THEN CLEAR VT55
 1812  007566  104400  013042                  TYPE    ,MSG16
 1813  007572  104400  013643                  TYPE    ,BUFF1          ;TYPE BUFF1-PRINT GRID
 1814  007576  012700  017124                  MOV     #DIST,R0        ;POINTER TO STATE WIDTH DISTRIBUTION
 1815  007602  013701  000310                  MOV     200.,R1         ;GO 200. TIMES UP TO 2 LSB
 1816  007606  012002          NXTY1:  MOV     (R0)+,R2
 1817  007610  004737  011272                  JSR     PC,LOADY
 1818  007614  005002                          CLR     R2
 1819  007616  004737  011272                  JSR     PC,LOADY
 1820  007622  005301                          DEC     R1
 1821  007624  001370                          BNE     NXTY1
 1822  007626  104400  013566                  TYPE    ,C2             ;TYPE ASCIZ STRING
 1823  007632  004737  010242                  JSR     PC,DELCLR
 1824
```

```
1825                                        ;CHANGE HISTOGRAM ERROR TO RELATIVE ACCURACY ERROR
1826
1827  007636  005001            RELACC: CLR     R1              ;RUNNING ERROR = 0
1828  007640  005003                    CLR     R3              ;MAXIMUM ERROR = 0
1829  007642  104400  013435            TYPE    ,MSG21
1830  007646  012700  017746            MOV     #BUFFER+2,R0
1831  007652  011002            NXTSTA: MOV     (R0),R2         ;STATE WIDTH = R2
1832  007654  162702  001440            SUB     #800.,R2        ;STATE WIDTH ERROR IN R2
1833  007660  060201                    ADD     R2,R1           ;UPDATE RUNNING ERROR
1834  007662  010120                    MOV     R1,(R0)+        ;SAVE IN BUFFER
1835  007664  010104                    MOV     R1,R4           ;SAVE IN R4 ALSO
1836  007666  100001                    BPL     PLUS            ;IS IT POSITIVE?
1837  007670  005404                    NEG     R4              ;NO - MAKE IT POSITIVE
1838  007672  020403            PLUS:   CMP     R4,R3           ;CHECK AGAINST PREVIOUS MAX. ERROR
1839  007674  003405                    BLE     NOTNEW          ;NOT A NEW MAXIMUM
1840  007676  010403                    MOV     R4,R3           ;UPDATE MAXIMUM IN R3
1841  007700  010005                    MOV     R0,R5
1842  007702  162705  017746            SUB     #BUFFER+2,R5
1843  007706  006205                    ASR     R5              ;R5=EDGE VALUE AT MAX. RELACC
1844  007710  020027  037742    NOTNEW: CMP     R0,#BUFFER+8190.  ;DONE?
1845  007714  001356                    BNE     NXTSTA          ;NO - REPEAT
1846  007716  006203                    ASR     R3              ;RESCALE FROM 1 LSB = 800. SCALING
1847  007720  006203                    ASR     R3              ;TO 1 LSB = 100. SCALING
1848  007722  006203                    ASR     R3
1849  007724  005503                    ADC     R3
1850  007726  010302                    MOV     R3,R2
1851  007730  004737  011374            JSR     PC,DECTYP
1852  007734  104400  013462            TYPE    ,LINEA
1853  007740  010546                    MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
1854                                                            ;;TYPE VALUE
1855  007742  104402                    TYPOS                   ;;GO TYPE--OCTAL ASCII
1856  007744     004                    .BYTE   4               ;;TYPE 4 DIGIT(S)
1857  007745     001                    .BYTE   1               ;;TYPE LEADING ZEROS
1858  007746  104400  012321            TYPE    ,SLASH                  ;PRINT '/'
1859  007752  005205                    INC     R5
1860  007754  010546                    MOV     R5,-(SP)        ;;SAVE R5 FOR TYPEOUT
1861                                                            ;;TYPE VALUE
1862  007756  104402                    TYPOS                   ;;GO TYPE--OCTAL ASCII
1863  007760     004                    .BYTE   4               ;;TYPE 4 DIGIT(S)
1864  007761     001                    .BYTE   1               ;;TYPE LEADING ZEROS
1865  007762  020337  011640            CMP     R3,VLIN
1866  007766  003403                    BLE     41$
1867  007770  104400  012375            TYPE    ,ERMSG
1868  007774  000402                    BR      42$
1869  007776  104400  012364    41$:    TYPE    ,OKMSG
1870  010002  005737  001404    42$:    TST     FLAG            ;VT55?
1871  010006  001503                    BEQ     L02
1872  010010  012700  017744            MOV     #BUFFER,R0
1873  010014  012701  010000            MOV     #4096.,R1
```

```
1874  010020  011002                GETDAT: MOV     (R0),R2         ;GET RELATIVE ACCURACY ERROR SCALED 1LSB = 800.
1875  010022  006202                        ASR     R2              ;RESCALE IT TO 1 LSB = 100.
1876  010024  006202                        ASR     R2
1877  010026  006202                        ASR     R2
1878  010030  005502                        ADC     R2
1879  010032  062702  000166              ADD     #118.,R2        ;AND MOVE IT TO MID-SCREEN
1880  010036  010220                      MOV     R2,(R0)+        ;PUT IT BACK INTO BUFFER
1881  010040  005301                      DEC     R1
1882  010042  001366                      BNE     GETDAT
1883  010044  012700  017744              MOV     #BUFFER,R0
1884  010050  012704  017744              MOV     #BUFFER,R4
1885  010054  012705  017746              MOV     #BUFFER+2,R5
1886  010060  012701  001000              MOV     #512.,R1
1887  010064  012702  000007      NXT8:   MOV     #7.,R2
1888  010070  012003                      MOV     (R0)+,R3
1889  010072  010337  001420              MOV     R3,MIN          ;MINIMUM
1890  010076  010337  001424              MOV     R3,MAX          ;MAXIMUM
1891  010102  012003              NXTCMP: MOV     (R0)+,R3
1892  010104  020337  001420              CMP     R3,MIN
1893  010110  002002                      BGE     MAXTST
1894  010112  010337  001420              MOV     R3,MIN          ;NEW MINIMUM
1895  010116  020337  001424      MAXTST: CMP     R3,MAX
1896  010122  003402                      BLE     TST8
1897  010124  010337  001424              MOV     R3,MAX          ;NEW MAXIMUM
1898  010130  005302              TST8:   DEC     R2
1899  010132  001363                      BNE     NXTCMP
1900  010134  013724  001420              MOV     MIN,(R4)+
1901  010140  013725  001424              MOV     MAX,(R5)+
1902  010144  022425                      CMP     (R4)+,(R5)+     ;BUMP EACH ONCE MORE
1903  010146  005301                      DEC     R1
1904  010150  001345                      BNE     NXT8
1905  010152  104400  012750              TYPE    ,MSG18
1906  010156  104400  013671              TYPE    ,BUFF2          ;TYPE BUFF2
1907  010162  012700  017744              MOV     #BUFFER,R0
1908  010166  004737  010220              JSR     PC,LOAD
1909  010172  104400  013571              TYPE    ,C3             ;TYPE ASCIZ STRING
1910  010176  012700  017746              MOV     #BUFFER+2,R0
1911  010202  004737  010220              JSR     PC,LOAD
1912  010206  104400  013566              TYPE    ,C2             ;TYPE ASCIZ STRING
1913  010212  004737  010242              JSR     PC,DELCLR
1914  010216  000207              LO2:    RTS     PC
1915  010220  012701  001000      LOAD:   MOV     #512.,R1
1916  010224  012002              LOADO:  MOV     (R0)+,R2
1917  010226  005720                      TST     (R0)+
1918  010230  004737  011272              JSR     PC,LOADY
1919  010234  005301                      DEC     R1
1920  010236  001372                      BNE     LOADO
1921  010240  000207                      RTS     PC
```

```
 1922  010242  005000                    DELCLR: CLR     R0
 1923  010244  012701  000020                    MOV     #20,R1          ;DELAY BEFORE CLEANING SCREEN
 1924  010250  005300                    1$:     DEC     R0
 1925  010252  001376                            BNE     1$
 1926  010254  005301                            DEC     R1
 1927  010256  001374                            BNE     1$
 1928  010260  032777  010000  170652            BIT     #BIT12,@SWR     ;TEST FOR HALT FOR DISPLAY
 1929  010266  001401                            BEQ     2$              ;;DON'T HALT FOR DISPLAY
 1930  010270  000000                            HALT
 1931  010272  104400  013711            2$:     TYPE    ,VTINIT
 1932  010276  000207                            RTS     PC
 1933                                    ;;NOISE SUBROUTINE;;
 1934  010300  013537  001366            NOITST: MOV     @(R5)+,CHANL            ;LOAD CHANNEL
 1935  010304  013737  001366  001364            MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
 1936  010312  004737  006324                    JSR     PC,GETEDG               ;GET EDGE VALUE
 1937  010316  004737  010472                    JSR     PC,NOIA                 ;GET RMS AND PEAK VALUES
 1938  010322  012737  000001  006502            MOV     #1,EDGFLG
 1939  010330  004737  010336                    JSR     PC,TYPRP                ;TYPE RMS AND PEAK VALUES
 1940  010334  000205                            RTS     R5
 1941
 1942
 1943
 1944
 1945
 1946                                    ;;TYPE RMS AND PEAK VALUES;;
 1947  010336  104400  012261            TYPRP:  TYPE    ,NOI
 1948  010342  005737  001400                    TST     RMS
 1949  010346  100002                            BPL     POSRMS
 1950  010350  005037  001400                    CLR     RMS                     ;RMS<0,SET RMS=0
 1951  010354  005737  001402            POSRMS: TST     PEAK
 1952  010360  100002                            BPL     POSPEA
 1953  010362  005037  001402                    CLR     PEAK                    ;PEAK<0,SET PEAK=0
 1954  010366  013702  001400            POSPEA: MOV     RMS,R2
 1955  010372  004737  011374                    JSR     PC,DECTYP
 1956  010376  104400  012634                    TYPE    ,MESR
 1957  010402  013702  001402                    MOV     PEAK,R2
 1958  010406  004737  011374                    JSR     PC,DECTYP
 1959  010412  104400  012647                    TYPE    ,MESP
 1960  010416  004737  006440                    JSR     PC,TYPEDG
 1961  010422  104400  012271                    TYPE    ,CHAN
 1962  010426  013746  001366                    MOV     CHANL,-(SP)     ;;SAVE CHANL FOR TYPEOUT
 1963                                            ;;TYPE CHANL
 1964  010432  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
 1965  010434     002                            .BYTE   2               ;;TYPE 2 DIGIT(S)
 1966  010435     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
 1967  010436  023737  001400  011632            CMP     RMS,VNR         ;WITHIN LIMITS?
 1968  010444  003007                            BGT     ER
 1969  010446  023737  001402  011634            CMP     PEAK,VNP                ;WITHIN LIMITS?
 1970  010454  003003                            BGT     ER
 1971  010456  104400  012364                    TYPE    ,OKMSG
 1972  010462  000207                            RTS     PC
 1973  010464  104400  012375            ER:     TYPE    ,ERMSG
 1974  010470  000207                            RTS     PC
```

```
                                        ;;SUBROUTINES FOR NOISE TEST;;
010472  005037  001400          NOIA:   CLR     RMS                     ;CLEAR RMS VLAUE
010476  005037  001402                  CLR     PEAK                    ;CLEAR PEAK VALUE
010502  004537  006504          NOI1:   JSR     R5,SARSUB               ;DO SAR ROUTINE AT 16%
010506  000020                          16.
010510  063737  001410  001400          ADD     DAC,RMS         ;ADD RESULT TO RMS
010516  004537  006504                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 84%
010522  000124                          84.
010524  163737  001410  001400          SUB     DAC,RMS         ;SUBTRACT RESULT FROM RMS
010532  004537  006504                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 1%
010536  000001                          1
010540  063737  001410  001402          ADD     DAC,PEAK                ;ADD RESULT TO PEAK
010546  004537  006504                  JSR     R5,SARSUB               ;DO SAR ROUTINE AT 99%
010552  000143                          99.
010554  163737  001410  001402          SUB     DAC,PEAK                ;SUBTRACT RESULT FROM PEAK
010562  000207                          RTS     PC                      ;RETURN

010564  012537  001366          NOI8:   MOV     (R5)+,CHANL             ;GET CHANNEL VALUE
010570  063737  001336  001366          ADD     BASECH,CHANL
010576  013737  001366  001364          MOV     CHANL,DUMMY             ;LOAD DUMMY CHANNEL
010604  004737  006324                  JSR     PC,GETEDG               ;GET EDGE VALUES
010610  005037  001400                  CLR     RMS             ;CLEAR RMS VALUE
010614  005037  001402                  CLR     PEAK            ;CLEAR PEAK VALUE
010620  012737  000010  010706          MOV     #10,10$         ;SET UP COUNTER
010626  004737  010502          1$:     JSR     PC,NOI1         ;GET NOISE VALUES
010632  005237  001414                  INC     EDGE
010636  005337  010706                  DEC     10$
010642  001371                          BNE     1$              ;REPEAT 8 TIMES
010644  162737  000010  001414          SUB     #10,EDGE
010652  006237  001400                  ASR     RMS             ;SCALE IT TO 1 LSB=100.
010656  005537  001400                  ADC     RMS
010662  006237  001402                  ASR     PEAK
010666  005537  001402                  ADC     PEAK
010672  012737  000010  006502          MOV     #8.,EDGFLG
010700  004737  010336                  JSR     PC,TYPRP                ;TYPE RESULTS
010704  000205                          RTS     R5              ;RETURN
010706  000000                  10$:    0                       ;COUNTER


                                        ;;RANDOM NUMBER GENERATOR;;
010710  063737  001374  001372  RANDY:  ADD     RNB,RNA
010716  063737  001376  001372          ADD     RNC,RNA
010724  005537  001372                  ADC     RNA
010730  063737  001372  001374          ADD     RNA,RNB
010736  063737  001376  001374          ADD     RNC,RNB
010744  005537  001374                  ADC     RNB
010750  063737  001372  001376          ADD     RNA,RNC
010756  063737  001374  001376          ADD     RNB,RNC
010764  005537  001376                  ADC     RNC
010770  000207                          RTS     PC
```

```
                                        ;;ROUTINE TO AVERAGE 8 CONVERSIONS;;
010772  012500                  CONVRT: MOV     (R5)+,R0                ;GET CHANNEL VALUE
010774  063700  001336                  ADD     BASECH,R0
011000  010037  001366                  MOV     R0,CHANL
011004  000300                          SWAB    R0
011006  005037  001352                  CLR     TEMP
011012  010077  170300                  MOV     R0,@STREG               ;LOAD CHANNEL INTO MIX BITS
011016  012700  010000                  MOV     #10000,R0
011022  005300                  2$:     DEC     R0
011024  001376                          BNE     2$
011026  012777  001610  170270          MOV     #RETURN,@VECTOR         ;LOAD VECTOR
011034  012700  000010                  MOV     #10,R0          ;SET UP COUNTER
011040  152777  000101  170250  1$:     BISB    #101,@STREG             ;SET INTRPT. EN., START CONV.
011046  000001                          WAIT                    ;WAIT FOR CONVERSION
011050  067737  170246  001352          ADD     @ADBUFF,TEMP            ;READ BUFFER
011056  005300                          DEC     R0
011060  001367                          BNE     1$              ;DO 8 TIMES
011062  006237  001352                  ASR     TEMP            ;AVERAGE VALUE
011066  006237  001352                  ASR     TEMP
011072  006237  001352                  ASR     TEMP
011076  005537  001352                  ADC     TEMP
011102  000205                          RTS     R5              ;RETURN

                                        ;COMPARE $GDDAT AND $BDDAT;;
011104  012537  001124          COMPAR: MOV     (R5)+,$GDDAT            ;GET GOOD DATA
011110  013537  001406                  MOV     @(R5)+,SPREAD           ;GET SPREAD
011114  013737  001352  001126          MOV     TEMP,$BDDAT             ;GET BAD(ACTUAL) DATA
011122  013701  001126                  MOV     $BDDAT,R1
011126  013700  001124                  MOV     $GDDAT,R0
011132  160100                          SUB     R1,R0           ;GET DIFFERENCE
011134  100001                          BPL     7$
011136  005400                          NEG     R0
011140  020037  001406          7$:     CMP     R0,SPREAD               ;COMPARE IT TO SPREAD
011144  003001                          BGT     10$             ;GO TO ERROR PRINTOUT
011146  005725                          TST     (R5)+           ;BUMP RETURN POINTER AROUND ERROR CALL
011150  000205                  10$:    RTS     R5
```

```
2061                                        ;;SUBROUTINE TO TYPE INTRPT. TST MSG.;;
2062  011152  005737  001202          DUMW:   TST     $PASS
2063  011156  001021                        BNE     20$
2064  011160  012737  011222  001110        MOV     #20$,$LPERR
2065  011166  012737  011222  001106        MOV     #20$,$LPADR
2066  011174  104400  013621                TYPE    ,METST                  ;TYPE ASCIZ STRING
2067  011200  010046                        MOV     R0,-(SP)                ;;SAVE R0 FOR TYPEOUT
2068                                                                        ;;TYPE TEST NO.
2069  011202  104402                        TYPOS                           ;;GO TYPE--OCTAL ASCII
2070  011204     002                        .BYTE   2                       ;;TYPE 2 DIGIT(S)
2071  011205     000                        .BYTE   0                       ;;SUPPRESS LEADING ZEROS
2072  011206  104400  012711                TYPE    ,ONAD
2073  011212  013746  001316                MOV     $TREG,-(SP)             ;;SAVE STREG FOR TYPEOUT
2074                                                                        ;;TYPE BUS ADDRESS
2075  011216  104402                        TYPOS                           ;;GO TYPE--OCTAL ASCII
2076  011220     006                        .BYTE   6                       ;;TYPE 6 DIGITS
2077  011221     001                        .BYTE   1                       ;;TYPE LEADING ZEROS
2078  011222  000207                  20$:   RTS     PC
2079
2080  011224  005737  001202          DUMC:   TST     $PASS
2081  011230  001010                        BNE     30$
2082  011232  012737  011252  001110        MOV     #30$,$LPERR
2083  011240  012737  011252  001106        MOV     #30$,$LPADR
2084  011246  104400  012306                TYPE    ,DONE
2085  011252  000207                  30$:   RTS     PC
```

```
                                        ;SUBROUTINE TO RESET & SET INTRPT. EN.;
011254  000005                  RST:    RESET
011256  052777  000100  167660          BIS     #100,@#TKS
011264  005037  177776                  CLR     PSW
011270  000207                          RTS     PC

                                        ;SUBROUTINE LOADY;
011272  005702                  LOADY:  TST     R2              ;ROUTINE TO LOAD VLAUE INTO R2
011274  100001                          BPL     PLUSR2          ;AS A VT55 Y-VALUE
011276  005002                          CLR     R2
011300  020227  000353          PLUSR2: CMP     R2,#235.
011304  002402                          BLT     LESS
011306  012702  000353                  MOV     #235.,R2
011312  010203                  LESS:   MOV     R2,R3
011314  042702  177740                  BIC     #177740,R2
011320  052702  000040                  BIS     #40,R2
011324  105777  167620          B10:    TSTB    @#TPS           ;PRINT CHARACTER
011330  100375                          BPL     B10
011332  110277  167614                  MOVB    R2,@#TPB
011336  006203                          ASR     R3
011340  006203                          ASR     R3
011342  006203                          ASR     R3
011344  006203                          ASR     R3
011346  006203                          ASR     R3
011350  042703  177770                  BIC     #177770,R3
011354  052703  000040                  BIS     #40,R3
011360  105777  167564          B11:    TSTB    @#TPS           ;PRINT CHARACTER
011364  100375                          BPL     B11
011366  110377  167560                  MOVB    R3,@#TPB
011372  000207                          RTS     PC
```

```
                                        ;;SUBROUTINE TO TYPE DECIMAL VALUE;;
                                        ;;IN R2 AS X.XX;;
011374  005702                DECTYP: TST     R2              ;TEST VALUE TO BE TYPED
011376  100003                        BPL     POS
011400  104400  012123                TYPE    ,MINUS                  ;TYPE MINUS SIGN
011404  005402                        NEG     R2
011406  020227  001747        POS:    CMP     R2,#999.                ;>999. REPLACE IT WITH 999.
011412  003402                        BLE     OKAYD
011414  012702  001747                MOV     #999.,R2
011420  105037  014455        OKAYD:  CLRB    ONES            ;CLEAR ONES
011424  105037  014454                CLRB    TENS            ;CLEAR TENS
011430  105037  014452                CLRB    HUNS            ;CLEAR HUNS
011434  005702                TESTR2: TST     R2                      ;CONVERT VALUE TO A DECIMAL VALUE
011436  001424                        BEQ     TYPOUT
011440  005302                        DEC     R2
011442  105237  014455                INCB    ONES
011446  123727  014455  000012        CMPB    ONES,#10.
011454  001367                        BNE     TESTR2
011456  105037  014455                CLRB    ONES
011462  105237  014454                INCB    TENS
011466  123727  014454  000012        CMPB    TENS,#10.
011474  001357                        BNE     TESTR2
011476  105037  014454                CLRB    TENS
011502  105237  014452                INCB    HUNS
011506  000752                        BR      TESTR2
011510  152737  000060  014452  TYPOUT: BISB    #60,HUNS                ;PREPARE FOR TYPOUT
011516  152737  000060  014454        BISB    #60,TENS
011524  152737  000060  014455        BISB    #60,ONES
011532  104400  014452                TYPE    ,HUNS           ;TYPE VALUE
011536  000207                        RTS     PC

011540  012701  011632        WFADJ:  MOV     #VNR,R1                 ;SUBROUTINE TO SET UP LIMITS
011544  005737  001336                TST     BASECH                  ;TESTING AN AM11K?
011550  001403                        BEQ     1$              ;;
011552  012702  011664                MOV     #VARLT3,R2              ;BASECH NOT ZERO, USE AM11K LIMITS
011556  000410                        BR      3$              ;;
011560  005737  001422        1$:     TST     WFTEST
011564  001003                        BNE     2$
011566  012702  011644                MOV     #VARLT1,R2              ;WFTEST=0,USE NORMAL LIMITS
011572  000402                        BR      3$
011574  012702  011654        2$:     MOV     #VARLT2,R2              ;WFTEST=1,USE OPTION AREA LIMITS
011600  012221                3$:     MOV     (R2)+,(R1)+
011602  005711                        TST     (R1)
011604  100375                        BPL     3$
011606  000207                        RTS     PC
```

```
2163  011610  000001                  V1:     1                       ;TOLERANCE VALUES FOR FUNCTIONAL TESTS
2164  011612  000002                  V2:     2
2165  011614  000010                  V10:    10
2166  011616  000050                  V50:    50
2167  011620  000144                  V144:   144
2168  011622  000115                  V115:   115
2169  011624  000240                  V240:   240
2170  011626  000005                  V5:     5
2171  011630  000062                  V50D:   50.
2172
2173  011632  000000                  VNR:    0                       ;RMS NOISE LIMIT
2174  011634  000000                  VNP:    0                       ;PEAK NOISE LIMIT
2175  011636  000000                  VSET:   0                       ;INTER-CHANNEL SETTLING LIMIT
2176  011640  000000                  VLIN:   0                       ;RELATIVE ACCURACY ERROR LIMIT
2177  011642  100000                          BIT15
2178
2179  011644  000031                  VARLT1: 25.             ;.25 LSB,NORMAL LIMITS FOR SYSTEM
2180  011646  000310                          200.            ;2. LSB,  INTEGRATION AND FIELD USE ON SPEC TESTS
2181  011650  000144                          100.            ;1 LSB
2182  011652  000144                          100.            ;1 LSB
2183
2184  011654  000027                  VARLT2: 23.             ;.23 LSB,  TIGHTER LIMITS FOR OPTION
2185  011656  000226                          150.            ;1.5 LSB,  AREA USE ON SPEC TESTS
2186  011660  000132                          90.             ;.9 LSB
2187  011662  000132                          90.             ;.9 LSB
2188
2189  011664  000062                  VARLT3: 50.             ;.5 LSB,  LIMITS FOR AM11K TESTING
2190  011666  000310                          200.            ;2. LSB
2191  011670  000226                          150.            ;1.5 LSB
2192  011672  000226                          150.            ;1.5 LSB
2193
2194  011674  052777  000100  167242  AGATST: BIS     #100,@$TKS
2195  011702  000177  000000                  JMP     @AGTST
2196  011706  001620                  AGTST:  BEGIN
```

```
                                        .SBTTL  END OF PASS ROUTINE

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

011710                                  $EOP:
011710  000240                                  NOP
011712  005037  001102                        CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011716  005037  001160                        CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
011722  005237  001202                        INC     $PASS           ;;INCREMENT THE PASS NUMBER
011726  042737  100000  001202                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011734  005327                                DEC     (PC)+           ;;LOOP?
011736  000001                          $EOPCT: .WORD   1
011740  003015                                  BGT     $DOAGN          ;;YES
011742  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011744  000001                          $ENDCT: .WORD   1
011746  011736                                  $EOPCT
011750  104400  012003                        TYPE    ,$ENDMG         ;;TYPE "END PASS"
011754  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011760  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011762  000005                                  RESET                   ;;CLEAR THE WORLD
011764  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011766  000240                                  NOP                     ;;SAVE ROOM
011770  000240                                  NOP                     ;;FOR
011772  000240                                  NOP                     ;;ACT11
011774                                  $DOAGN:
011774  000137                                  JMP     @(PC)+          ;;RETURN
011776  011674                          $RTNAD: .WORD   AGATST
012000     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
012003     015  042412  042116          $ENDMG: .ASCIZ  <15><12>/END PASS/
012010  050040  051501  000123
```

```
                                                .SBTTL          ASCII MESSAGES
2235  012016  005015  047516  051511  NOIMSG: .ASCIZ  <15><12>/NOISE TEST-- /
2236  012024  020105  042524  052123
2237  012032  026455  000040
2238  012036  005015  042523  052124  SETMSG: .ASCIZ  <15><12>/SETTLING TEST-- TYPE DESIRED 'FROM' CHANNEL & CR: /
2239  012044  044514  043516  052040
2240  012052  051505  026524  020055
2241  012060  054524  042520  042040
2242  012066  051505  051111  042105
2243  012074  023440  051106  046517
2244  012102  020047  044103  047101
2245  012110  042516  020114  020046
2246  012116  051103  020072     000
2247  012123     055     000          MINUS:  .BYTE   55,0
2248  012125     077     000          QUEST:  .BYTE   77,0
2249  012127     136     101     040  AMSG:   .BYTE   136,101,40,40,0
2250  012132     040     000
2251  012134     136     103     040  CMSG:   .BYTE   136,103,40,40,0
2252  012137     040     000
2253  012141     136     107     015  GMSG:   .BYTE   136,107,15,12,123,127,122,105,107,72,0
2254  012144     012     123     127
2255  012147     122     105     107
2256  012152     072     000
2257  012154  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
2258  012162     000
2259  012163     055  020055     000  DASH:   .ASCIZ  /-- /
2260  012167     123  040524  042524  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
2261  012174  026455  053440  042111
2262  012202  044124  005015     000
2263  012207     103  000110          CH:     .ASCIZ  /CH/
2264  012212  020040  020040     000  SPACE:  .ASCIZ  /    /
2265  012217     040  051514  020102  LSB:    .ASCIZ  / LSB ON CH/
2266  012224  047117  041440  000110
2267  012232  051440  052105  046124  SETCH:  .ASCIZ  / SETTLING FROM CH/
2268  012240  047111  020107  051106
2269  012246  046517  041440  000110
2270  012254  040440  020124     000  ATMSG:  .ASCIZ  / AT /
2271  012261     116  044517  042523  NOI:    .ASCIZ  /NOISE: /
2272  012266  020072     000
2273  012271     040  047117  041440  CHAN:   .ASCIZ  / ON CHANNEL /
2274  012276  040510  047116  046105
2275  012304  000040
2276  012306  020040  020040  047504  DONE:   .ASCIZ  /    DONE/<15><12>
2277  012314  042516  005015     000
2278  012321     057     000          SLASH:  .ASCIZ  #/#
2279  012323     124  050131  020105  TOMSG:  .ASCIZ  /TYPE DESIRED 'TO' CHANNEL & CR: /
2280  012330  042504  044523  042522
2281  012336  020104  052047  023517
2282  012344  041440  040510  047116
2283  012352  046105  023040  041440
2284  012360  035122  000040
2285  012364  020040  020040  045517  OKMSG:  .ASCIZ  /    OK/<15><12>
2286  012372  005015     000
```

```
2287  012375  040     025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
2288  012402  047522  025122  006452
2289  012410  000012
2290  012412  051440  044513  050120  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/
2291  012420  042105  051440  040524
2292  012426  042524  051450  000051
2293  012434  047040  051101  047522  NARMSG: .ASCIZ  # NARROW (< 1/2 LSB) STATE(S)#<15><12>
2294  012442  020127  036050  030440
2295  012450  031057  046040  041123
2296  012456  020051  052123  052101
2297  012464  024105  024523  005015
2298  012472  000
2299  012473  040     044527  042504  WIDMSG: .ASCIZ  # WIDE (> 1 1/2 LSB) STATE(S)#<15><12>
2300  012500  024040  020076  020061
2301  012506  027461  020062  051514
2302  012514  024502  051440  040524
2303  012522  042524  051450  006451
2304  012530  000012
2305  012532  051440  040524  042524  OUTMSG: .ASCIZ  / STATE(S) WIDER THAN 2 LSB/
2306  012540  051450  020051  044527
2307  012546  042504  020122  044124
2308  012554  047101  031040  046040
2309  012562  041123  000
2310  012565  040     052123  052101  HAFMSG: .ASCIZ  # STATE-WIDTH(S) OUTSIDE + OR - 1/2 LSB#
2311  012572  026505  044527  052104
2312  012600  024110  024523  047440
2313  012606  052125  044523  042504
2314  012614  025440  047440  020122
2315  012622  020055  027461  020062
2316  012630  051514  000102
2317  012634  046040  041123  051040  MESR:   .ASCIZ  / LSB RMS, /
2318  012642  051515  020054  000
2319  012647  040     051514  020102  MESP:   .ASCIZ  / LSB PEAK AT /
2320  012654  042520  045501  040440
2321  012662  020124  000
2322  012665  015     042412  042116  MEND:   .ASCII  <15><12>/END OF LOGIC TESTS/
2323  012672  047440  020106  047514
2324  012700  044507  020103  042524
2325  012706  052123  123
2326  012711  040     047117  040440  ONAD:   .ASCIZ  / ON AD11K AT /
2327  012716  030504  045461  040440
2328  012724  020124  000
2329  012727  040     042101  030461  MSG50:  .ASCIZ  / AD11K'S F^UND/<15><12>
2330  012734  023513  020123  047506
2331  012742  047125  006504  000012
2332  012750  005012  025412  027461  MSG18:  .ASCII  <12><12><12>#+1/2 LSB#<15><12><12><12><12><12><12><12><12><12><12><12><1
2333  012756  020062  051514  006502
2334  012764  005012  005012  005012
2335  012772  005012  005012  005012
2336  013000  030455  031057  051514          .ASCIZ  \-1/2LSB\
2337  013006  000102
2338
```

```
2339                                                    .EVEN
2340  013010  044504  043106  051105  MSG20:  .ASCIZ  /DIFFERENTIAL LINEARITY:/<15><12>
2341  013016  047105  044524  046101
2342  013024  046040  047111  040505
2343  013032  044522  054524  006472
2344  013040  000012
2345  013042  020040  020040  020040  MSG16:  .ASCII  /                    STATE-WIDTH DISTRIBUTION/<15><12><12><12>
2346  013050  020040  020040  020040
2347  013056  020040  020040  020040
2348  013064  020040  052123  052101
2349  013072  026505  044527  052104
2350  013100  020110  044504  052123
2351  013106  044522  052502  044524
2352  013114  047117  005015  005012
2353  013122  020040  020043  043117          .ASCII  /  # OF STATES/<12><12><12><12><12><12><12><12><12><12><12><12><12><
2354  013130  051440  040524  042524
2355  013136  005123  005012  005012
2356  013144  005012  005012  005012
2357  013152  005012  005012  005012
2358  013160  005012
2359  013162  020040  020040  020040          .ASCII  /                                                STATE WIDTH (LSB)/<15>
2360  013170  020040  020040  020040
2361  013176  020040  020040  020040
2362  013204  020040  020040  020040
2363  013212  020040  020040  020040
2364  013220  020040  020040  020040
2365  013226  020040  020040  020040
2366  013234  020040  020040  020040
2367  013242  051440  040524  042524
2368  013250  053440  042111  044124
2369  013256  024040  051514  024502
2370  013264  005015
2371  013266  030040  020040  020040          .ASCIZ  # 0              1/2              1              1 1/2              2#
2372  013274  020040  020040  020040
2373  013302  020040  020040  027461
2374  013310  020062  020040  020040
2375  013316  020040  020040  020040
2376  013324  020040  020061  020040
2377  013332  020040  020040  020040
2378  013340  020040  030440  030440
2379  013346  031057  020040  020040
2380  013354  020040  020040  020040
2381  013362  020040  031040     000
2382  013367     015  052012  050131  MSG71:  .ASCIZ  <15><12>/TYPE LETTER & CR FOR DESIRED TEST: /
2383  013374  020105  042514  052124
2384  013402  051105  023040  041440
2385  013410  020122  047506  020122
2386  013416  042504  044523  042522
2387  013424  020104  042524  052123
2388  013432  020072     000
2389  013435     122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
2390  013442  053111  020105  041501
2391  013450  052503  040522  054503
2392  013456  006472  000012
2393  013462  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUM AT /
2394  013470  054101  046511  046525
```

```
2395  013476  040440  020124     000
2396  013503     015  041412  046101  HEADS:  .ASCII  <15><12>/CALIBRATION--/
2397  013510  041111  040522  044524
2398  013516  047117  026455
2399  013522  051440  052105  041440  ASKCH:  .ASCIZ  / SET CHANNEL IN SWR LOW BYTE/<15><12>
2400  013530  040510  047116  046105
2401  013536  044440  020116  053523
2402  013544  020122  047514  020127
2403  013552  054502  042524  005015
2404  013560     000
2405  013561     033  000132          CO:     .ASCIZ  <33><132>
2406  013564  000055                  C1:     .ASCIZ  <55>
2407  013566  031033     000          C2:     .ASCIZ  <33><62>
2408  013571     112     000          C3:     .ASCIZ  <112>
2409  013573     015  047412  043106  MOFSET: .ASCIZ  <15><12>/OFFSET =/
2410  013600  042523  020124  000075
2411  013606  046040  041123  000040  MLSB:   .ASCIZ  / LSB /
2412  013614  040440  020124     000  MAT:    .ASCIZ  / AT /
2413  013621     015  020012  047105  METST:  .ASCIZ  <15><12>/ ENTERING TEST /
2414  013626  042524  044522  043516
2415  013634  052040  051505  020124
2416  013642     000
2417  013643     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
2418  013646     061     111     062
2419  013651     114     041     060
2420  013654     045     063     051
2421  013657     066     055     071
2422  013662     061     074     110
2423  013665     041     040     112
2424  013670     000
2425  013671     033     061     101  BUFF2:  .BYTE   33,61,101,47,111,61,104,50,65,44,62,110,40,40,102,0
2426  013674     047     111     061
2427  013677     104     050     065
2428  013702     044     062     110
2429  013705     040     040     102
2430  013710     000
2431  013711     033     110     033  VTINIT: .BYTE   33,110,33,112,33,61,101,40,33,62,0
2432  013714     112     033     061
2433  013717     101     040     033
2434  013722     062     000
2435  013724  005015  046412  026504  HEAD1:  .ASCII  <15><12><12>/MD-11-DZADL-A    AD11K DIAGNOSTIC/<15><12>
2436  013732  030461  042055  040532
2437  013740  046104  040455  020040
2438  013746  020040  042101  030461
2439  013754  020113  044504  043501
2440  013762  047516  052123  041511
2441  013770  005015
2442  013772  040412  020072  052501          .ASCII  <12>/A: AUTO TEST/
2443  014000  047524  052040  051505
2444  014006     124
2445  014007     015  041412  020072          .ASCII  <15><12>/C: CALIBRATION/
2446  014014  040503  044514  051102
2447  014022  052101  047511     116
2448  014027     015  046012  020072          .ASCII  <15><12>/L: LOGIC TEST/
2449  014034  047514  044507  020103
2450  014042  042524  052123
```

```
  2451  014046  005015  035116  047040          .ASCII  <15><12>/N: NOISE TEST/
  2452  014054  044517  042523  052040
  2453  014062  051505     124
```

```
2454  014065  015     051412  020072          .ASCII  <15><12>/S: SETTLE TEST/
2455  014072  042523  052124  042514
2456  014100  052040  051505  124
2457  014105  015     053412  020072          .ASCIZ  <15><12>/W: WRAPAROUND TEST/<15><12>
2458  014112  051127  050101  051101
2459  014120  052517  042116  052040
2460  014126  051505  006524  000012
2461  014134  005015  052123  052101  EM1:    .ASCIZ  <15><12>/STATUS REG. ERROR/<15><12>
2462  014142  051525  051040  043505
2463  014150  020056  051105  047522
2464  014156  006522  000012
2465  014162  005015  040506  046111  EM2:    .ASCIZ  <15><12>/FAILED TO INTERRUPT/<15><12>
2466  014170  042105  052040  020117
2467  014176  047111  042524  051122
2468  014204  050125  006524  000012
2469  014212  005015  047125  054105  EM3:    .ASCIZ  <15><12>/UNEXPECTED INTERRUPT/<15><12>
2470  014220  042520  052103  042105
2471  014226  044440  052116  051105
2472  014234  052522  052120  005015
2473  014242  000
2474  014243  015     042412  051122  EM4:    .ASCIZ  <15><12>#ERROR ON A/D CHANNEL#<15><12>
2475  014250  051117  047440  020116
2476  014256  027501  020104  044103
2477  014264  047101  042516  006514
2478  014272  000012
2479  014274  051105  050122  020103  DH1:    .ASCIZ  /ERRPC STREG EXPECTED ACTUAL/<15><12>
2480  014302  052123  042522  020107
2481  014310  054105  042520  052103
2482  014316  042105  040440  052103
2483  014324  040525  006514  000012
2484  014332  051105  050122  020103  DH2:    .ASCIZ  /ERRPC  STREG   CHANNEL  NOMINAL  TOLERANCE  ACTUAL/
2485  014340  051440  051124  043505
2486  014346  020040  041440  040510
2487  014354  047116  046105  020040
2488  014362  047516  044515  040516
2489  014370  020114  052040  046117
2490  014376  051105  047101  042503
2491  014404  020040  041501  052524
2492  014412  046101  000
2493  014415  105     051122  041520  DH3:    .ASCIZ  /ERRPC      STREG    ACTUAL/<15><12>
2494  014422  020040  020040  020040
2495  014430  052123  042522  020107
2496  014436  020040  040440  052103
2497  014444  040525  006514  000012
```

```
2498  014452  000                     HUNS:   .BYTE   0
2499  014453  056                     DECPNT: .BYTE   56
2500  014454  000                     TENS:   .BYTE   0
2501  014455  000     000             ONES:   .BYTE   0,0
2502          014460                  .EVEN
2503
2504  014460  001116  001316  001124  DT1:    $ERRPC, STREG, $GDDAT, $BDDAT,0
2505  014466  001126  000000
2506  014472  001116  001316  001366  DT2:    $ERRPC,STREG,CHANL,$GDDAT,SPREAD,$BDDAT,0
2507  014500  001124  001406  001126
2508  014506  000000
2509  014510  001116  001316  001126  DT3:    $ERRPC,STREG,$BDDAT,0
2510  014516  000000
2511
2512  014520  000000                  DF1:    0
2513
2514
2515
2516
```

```
                                        .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
                                        ;

014522  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014524  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
014532  105777  164406          1$:     TSTB    @$TKS           ;;WAIT FOR
014536  100375                          BPL     1$              ;;A CHARACTER
014540  117766  164402  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
014546  042766  177600  000004          BIC     #^C<177>,4(SP.  ;;GET RID OF JUNK IF ANY
014554  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014562  001013                          BNE     3$              ;;BRANCH IF NO
014564  105777  164354          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014570  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
014572  117746  164350                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014576  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
014602  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
014606  001366                          BNE     2$              ;;IF NOT DISCARD IT
014610  000750                          BR      1$              ;;YES, RESUME
014612  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
014620  002407                          BLT     4$              ;;BRANCH IF YES
014622  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
014630  003003                          BGT     4$              ;;BRANCH IF YES
014632  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
014640  000002                  4$:     RTI                     ;;GO BACK TO USER
                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE T.Y
                                        ;*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

014642  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
014644  012703  014750          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
014650  022703  014760          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
014654  101405                          BLOS    4$              ;;BR IF YES
014656  104404                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
014660  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
014662  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
014666  001003                          BNE     3$              ;;SKIP IF NOT
014670  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
014674  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
014676  111337  014746          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
014702  104400  014746                  TYPE    ,9$
```

```
2573  014706  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2574  014712  001356                            BNE     2$              ;;LOOP IF NOT RETURN
2575  014714  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2576  014720  104400  001172                    TYPE    ,$LF            ;;TYPE A LINE FEED
2577  014724  012603                            MOV     (SP)+,R3        ;;RESTORE R3
2578  014726  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2579  014730  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2580  014736  012766  014750  000004            MOV     #$TTYIN,4(SP)
2581  014744  000002                            RTI                     ;;RETURN
2582  014746     000                  9$:        .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2583  014747     000                             .BYTE   0               ;;TERMINATOR
2584  014750  000010                  $TTYIN:    .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2585  014760  052536  005015     000  $CNTLU:    .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2586  014765     136  006507  000012  $CNTLG:    .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2587  014772  005015  053523  020122  $MSWR:     .ASCIZ  <15><12>/SWR = /
2588  015000  020075     000
2589  015003     040  047040  053505  $MNEW:     .ASCIZ  /  NEW = /
2590  015010  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

015014  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
015016  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
015024  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015026  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015030  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015032  104405                  1$:    RDLIN                   ;;READ AN ASCIZ LINE
015034  012600                         MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
015036  005001                         CLR     R1              ;;CLEAR DATA WORD
015040  005002                         CLR     R2
015042  112046                  2$:    MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
015044  001412                         BEQ     3$              ;;IF ZERO GET OUT
015046  006301                         ASL     R1              ;;*2
015050  006102                         ROL     R2
015052  006301                         ASL     R1              ;;*4
015054  006102                         ROL     R2
015056  006301                         ASL     R1              ;;*8
015060  006102                         ROL     R2
015062  042716  177770                 BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
015066  062601                         ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
015070  000764                         BR      2$              ;;LOOP
015072  005726                  3$:    TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
015074  010166  000012                 MOV     R1,12(SP)       ;;SAVE THE RESULT
015100  010237  015114                 MOV     R2,$HIOCT
015104  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
015106  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
015110  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
015112  000002                         RTI                     ;;RETURN
015114  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

015116                                  $SCOPE:
015116  032777  040000  164014  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
015124  001114                          BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
015126  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
015130  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015134  012737  015154  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
015142  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
015146  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015152  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
015154  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
015156  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015162  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
015164                          6$:;*****END OF CODE FOR THE XOR TESTER*****
015164  032777  000400  163746          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
015172  001404                          BEQ     2$              ;;BR IF NO
015174  127737  163740  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
015202  001465                          BEQ     $OVER           ;;BR IF YES
015204  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
015210  001421                          BEQ     3$              ;;BR IF NO
015212  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015220  101015                          BHI     3$              ;;BR IF NO
015222  032777  001000  163710          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
015230  001404                          BEQ     4$              ;;BR IF NO
015232  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015240  000446                          BR      $OVER
015242  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
015246  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015252  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015254  032777  004000  163656  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015262  001011                          BNE     1$              ;;BR IF YES
015264  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015270  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015272  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015276  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015304  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015306  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015314  013737  015372  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015322  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
015326  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
015334  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

```
 2685  015340  011637  001110                 MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 2686  015344  005037  001162                 CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 2687  015350  112737  000001  001115         MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 2688  015356  013777  001102  163556 $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 2689  015364  013716  001106                 MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 2690  015370  000002                         RTI                     ;;FIXES PS
 2691  015372  003720                 $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
 2692                                  .SBTTL  ERROR HANDLER ROUTINE
 2693
 2694                                  ;;**************************************************************
 2695                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 2696                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 2697                                  ;*AND GO TO $ERRTYP ON ERROR
 2698                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 2699                                  ;*SW15=1         HALT ON ERROR
 2700                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 2701                                  ;*SW10=1         BELL ON ERROR
 2702                                  ;*SW09=1         LOOP ON ERROR
 2703                                  ;*CALL
 2704                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 2705
 2706  015374                          $ERROR:
 2707  015374  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 2708  015400  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 2709  015402  013777  001102  163532          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 2710  015410  032777  002000  163522          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 2711  015416  001402                          BEQ     1$              ;;NO - SKIP
 2712  015420  104400  001164                  TYPE    ,$BELL          ;;RING BELL
 2713  015424  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 2714  015430  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 2715  015434  162737  000002  001116          SUB     #2,$ERRPC
 2716  015442  117737  163450  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 2717  015450  032777  020000  163462          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 2718  015456  001004                          BNE     20$             ;;SKIP TYPEOUTS
 2719  015460  004737  015570                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
 2720  015464  104400  001171                  TYPE    ,$CRLF
 2721  015470                          20$:
 2722  015470  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 2723  015476  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
 2724  015500  113737  001114  015512          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
 2725  015506  004737  016224                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
 2726  015512     000                  21$:    .BYTE   0
 2727  015513     000                          .BYTE   0
 2728  015514  000777                  22$:    BR      22$             ;;APT ERROR LOOP
 2729  015516  005777  163416          2$:     TST     @SWR            ;;HALT ON ERROR
 2730  015522  100001                          BPL     3$              ;;SKIP IF CONTINUE
 2731  015524  000000                          HALT                    ;;HALT ON ERROR!
 2732  015526  032777  001000  163404  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 2733  015534  001402                          BEQ     4$              ;;BR IF NO
 2734  015536  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 2735  015542  005737  001162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 2736  015546  001402                          BEQ     5$              ;;BR IF NONE
 2737  015550  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 2738  015554                          5$:
 2739  015554  022737  011764  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 2740  015562  001001                          BNE     6$              ;;BRANCH IF NO
```

```
                                        HALT                    ;;YES
                                6$:
                                        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.

                                $ERRTYP:
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                        MOV     R0,-(SP)        ;;SAVE R0
                                        CLR     R0              ;;PICKUP THE ITEM INDEX
                                        BISB    @#$ITEMB,R0
                                        BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
                                        MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        BR      6$              ;;GET OUT
                                1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
                                        ASL     R0              ;;      WORK FOR THE ERROR TABLE
                                        ASL     R0
                                        ASL     R0
                                        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
                                        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
                                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
                                        TYPE                    ;;TYPE THE "ERROR MESSAGE"
                                2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
                                        BEQ     5$              ;;SKIP TYPEOUT IF 0
                                        TYPE                    ;;TYPE THE "DATA HEADER"
                                4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
                                        BNE     7$              ;;GO TYPE THE DATA
                                6$:     MOV     (SP)+,R0        ;;RESTORE R0
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                        RTS     PC              ;;RETURN
                                7$:
                                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
                                        BEQ     6$              ;;BR IF NO
                                        TYPE    ,8$             ;;TYPE TWO(2) SPACES
                                        BR      7$              ;;LOOP
                                8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                                        .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
                                        ;*

015724  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
015730  100002                          BPL     1$              ;;BR IF YES
015732  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
015734  000430                          BR      3$              ;;LEAVE
015736  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
015740  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
015744  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015752  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
015754  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
015762  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
015764  010037  015774                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
015770  004737  016214                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
015774  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
015776  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
016004  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
016006  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016010  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
016012  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
016014  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
016016  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
016022  000002                          RTI                     ;;RETURN
016024  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
016030  001430                          BEQ     8$
016032  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
016036  001006                          BNE     5$
016040  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
016042  104400                          TYPE                    ;;TYPE A CR AND LF
016044  001171                          $CRLF
016046  105037  016202                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
016052  000755                          BR      2$              ;;GET NEXT CHARACTER
016054  004737  016136          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
016060  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
016064  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
016066  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
016072  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
016076  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
016100  004737  016136                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
016104  105337  016202                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
 2847  016110  000770                          BR      7$              ;;LOOP
 2848
 2849                                  ;HORIZONTAL TAB PROCESSOR
 2850
 2851  016112  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 2852  016116  004737  016136          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 2853  016122  132737  000007  016202          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 2854  016130  001372                          BNE     9$              ;;TAB STOP
 2855  016132  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 2856  016134  000724                          BR      2$              ;;GET NEXT CHARACTER
 2857  016136  105777  163006          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 2858  016142  100375                          BPL     $TYPEC
 2859  016144  116677  000002  163000        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 2860  016152  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 2861  016160  001003                          BNE     1$              ;;BRANCH IF NO
 2862  016162  105037  016202                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 2863  016166  000406                          BR      $TYPEX          ;;EXIT
 2864  016170  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 2865  016176  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 2866  016200  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 2867  016202  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 2868  016204  000207          $TYPEX: RTS     PC
 2869
 2870                                  .SBTTL  APT COMMUNICATIONS ROUTINE
 2871
 2872                                  ;;**************************************************************
 2873  016206  112737  000001  016452  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
 2874  016214  112737  000001  016450  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
 2875  016222  000403                          BR      $ATYC
 2876  016224  112737  000001  016452  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
 2877  016232                          $ATYC:
 2878  016232  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2879  016234  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2880  016236  105737  016450                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
 2881  016242  001450                          BEQ     5$              ;;IF NOT:  BR
 2882  016244  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
 2883  016252  001031                          BNE     3$              ;;IF NOT:  BR
 2884  016254  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 2885  016262  001425                          BEQ     3$              ;;IF NOT:  BR
 2886  016264  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
 2887  016270  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 2888  016276  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
 2889  016302  001375                          BNE     1$              ;;IF NOT:  WAIT
 2890  016304  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
 2891  016310  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
 2892  016312  001376                          BNE     2$
 2893  016314  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
 2894  016320  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
 2895  016322  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
 2896  016326  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
 2897  016334  000413                          BR      5$
 2898  016336  017637  000004  016362  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
 2899  016344  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 2900  016352  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 2901  016356  004737  015724                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
 2902  016362  000000          4$:     .WORD   0
```

```
2903  016364                          5$:
2904  016364  105737  016452          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2905  016370  001416                          BEQ     12$             ;;IF NOT:  BR
2906  016372  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
2907  016376  001413                          BEQ     12$             ;;IF NOT:  BR
2908  016400  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2909  016404  001375                          BNE     11$             ;;IF NOT:  WAIT
2910  016406  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
2911  016414  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2912  016422  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2913  016426  105037  016452          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2914  016432  105037  016451                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2915  016436  105037  016450                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2916  016442  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2917  016444  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2918  016446  000207                          RTS     PC              ;;RETURN
2919  016450     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
2920  016451     000          $LFLG:  .BYTE   0               ;;LOG FLAG
2921  016452     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
2922          016454                  .EVEN
2923          000200          APTSIZE=200
2924          000001          APTENV=001
2925          000100          APTSPOOL=100
2926          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

016454  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016460  116637  000001  016677          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016466  112637  016701                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016472  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016476  000406                          BR      $TYPON
016500  112737  000001  016677  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016506  112737  000006  016701          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
016514  112737  000005  016676  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016522  010346                          MOV     R3,-(SP)        ;;SAVE R3
016524  010446                          MOV     R4,-(SP)        ;;SAVE R4
016526  010546                          MOV     R5,-(SP)        ;;SAVE R5
016530  113704  016701                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
016534  005404                          NEG     R4
016536  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
016542  110437  016700                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
016546  113704  016677                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
016552  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
016556  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
016560  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
016562  000404                          BR      3$              ;;GO DO MSB
016564  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
016566  006105                          ROL     R5
016570  006105                          ROL     R5
016572  010503                          MOV     R5,R3
016574  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
016576  105337  016700                  DECB    $OMODE          ;;TYPE THIS DIGIT?
016602  100016                          BPL     7$              ;;BR IF NO
016604  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
016610  001002                          BNE     4$              ;;TEST FOR 0
016612  005704                          TST     R4              ;;SUPPRESS THIS 0?
016614  001403                          BEQ     5$              ;;BR IF YES
```

```
 2983  016616  005204                    4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 2984  016620  052703  000060                    BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 2985  016624  052703  000040              5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 2986  016630  110337  016674                    MOVB    R3,8$           ;;SAVE FOR TYPING
 2987  016634  104400  016674                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
 2988  016640  105337  016676              7$:   DECB    $OCNT           ;;COUNT BY 1
 2989  016644  003347                          BGT     2$              ;;BR IF MORE TO DO
 2990  016646  002402                          BLT     6$              ;;BR IF DONE
 2991  016650  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 2992  016652  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 2993  016654  012605                    6$:   MOV     (SP)+,R5        ;;RESTORE R5
 2994  016656  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 2995  016660  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 2996  016662  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 2997  016670  012616                          MOV     (SP)+,(SP)
 2998  016672  000002                          RTI                     ;;RETURN
 2999  016674     000                    8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 3000  016675     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 3001  016676     000                 $OCNT:   .BYTE   0               ;;OCTAL DIGIT COUNTER
 3002  016677     000                 $OFILL:  .BYTE   0               ;;ZERO FILL SWITCH
 3003  016700  000000                 $OMODE:  .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  TRAP DECODER

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

016702  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
016704  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
016710  005740                          TST     -(R0)           ;;BACKUP BY 2
016712  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
016714  006300                          ASL     R0              ;;POSITION FOR INDEXING
016716  016000  016724                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
016722  000200                          RTS     R0              ;;GO TO ROUTINE

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
016724                          $TRPAD:
016724  015724                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
016726  016500                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
016730  016454                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
016732  016514                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)

016734  014522                          $RDCHR  ;;CALL=RDCHR    TRAP+4(104404)  TTY TYPEIN CHARACTER ROUTINE
016736  014642                          $RDLIN  ;;CALL=RDLIN    TRAP+5(104405)  TTY TYPEIN STRING ROUTINE
016740  015014                          $RDOCT  ;;CALL=RDOCT    TRAP+6(104406)  READ AN OCTAL NUMBER FROM TTY
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
016742  012737  017106  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
016750  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
016756  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016760  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016762  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016764  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016766  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
016770  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016772  017746  162142                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
016776  010637  017112                  MOV     SP,$SAVR6       ;;SAVE SP
017002  012737  017014  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
017010  000000                          HALT
017012  000776                          BR      .-2             ;;HANG UP

                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
017014  012737  017106  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
017022  013706  017112                  MOV     $SAVR6,SP       ;;GET SP
017026  005037  017112                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
017032  005237  017112          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
017036  001375                          BNE     1$              ;;OF  WORD
017040  012677  162074                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
017044  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
017046  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
017050  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
017052  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
017054  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
017056  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
017060  012737  016742  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
017066  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
017074  104400                          TYPE                    ;REPORT THE POWER FAILURE
017076  017114                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
017100  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEG2
017102  002256                  $PWRAD: .WORD   BEG2            ;;RESTART ADDRESS
017104  000002                          RTI
017106  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
017110  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
017112  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
017114  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
017122  000122
                                        .EVEN
                                .EVEN
017124  000310                  DIST:   .BLKW   200.            ;STATE-WIDTH DISTRIBUTION
017744  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*   997   1020   1074   1097   1115   1132   1173*  1179   1247*  1258*  1403   1461*
                         1470*  1472*  1546*  1597*  1610   1650*  1659   1665   1711   2039
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
ADST1   001320            711*  1460*  1469*  1471*  1655*
AENV  = 000000            619    630
AENVM = 000000            619    631
AFATAL= 000000            619    622
AGAIN   007040           1702*  1731
AGATST  011674           2194*  2229
AGTST   011706           1427*  1444*  1454*  2195   2196*
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    012127            767   2249*
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           2821   2926*
APTENV= 000001           2722   2814   2882   2924*
APTSIZ= 000200            832   2923*
APTSPO= 000100           2816   2884   2925*
AROUND  007162           1727   1730*
ASYNCH  013522           1501   2399*
ASYREG= 000000            619    632
```

```
ATESTN= 000000            619     623
ATMSG   012254           1543    2270*
AUNIT = 000000            619     626
AUSWR = 000000            619     633
AVECT1= 140340            523*    619     658     713     715
AVECT2= 000000            619     659
BASEBR  001326            714*    1056    1067    1475*   1476*
BASECH  001336            718*    1317    1318    1545    1576    1577    1696    1993    2027    2150
BEGIN   001620            536      789*   2196
BEGINA  005260            877     1431*
BEGINC  005036            880     1382*
BEGINL  002556            925*    1424    1434
BEGINN  005646            886     1499*
BEGINS  005716            889     1509*
BEGINW  005346            892     1448*
BEGIN2  001626            538      791*
BEGL    005222            868      883    1422*
BEG2    002256            537      764     865*   3074
BITPNT  001416            742*    1646*   1649    1672    1673*
BIT0  = 000001            495*    1017    1035    1059    1068    1088    1104    1108    1122    1123
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    2658
BIT09 = 001000            476*     486    2666    2732
BIT1  = 000002            494*
BIT10 = 002000            475*    2710
BIT11 = 004000            474*    2673
BIT12 = 010000            473*    1928
BIT13 = 020000            472*    1388    1404    2717
BIT14 = 040000            471*     935    2644
BIT15 = 100000            470*     966     994    1107    1120    2177
BIT2  = 000004            493*
BIT3  = 000010            492*
BIT4  = 000020            491*     959    1170    1172
BIT5  = 000040            490*     951
BIT6  = 000100            489*     943    1035    1039    1059    1068    1086    1088    1398
BIT7  = 000200            488*     991    1039    1107    1172
BIT8  = 000400            487*     931
BIT9  = 001000            486*     989    1109    1171
BK3     005750           1516*    1527
BPTVEC= 000014            502*
BUFFER  017744           1679     1716*   1718*   1733    1761    1830    1842    1844    1872    1883    1884    1885    1907
                         1910     3084*
BUFF1   013643           1813     2417*
BUFF2   013671           1906     2425*
BUMPAD  005410           1425     1442    1452    1457*
BYPASS  005602           1466     1482*
B10     011324           2102*    2103
B11     011360           2112*    2113
CH      012207           1390     2263*
```

```
CHAN    012271          1961    2273*
CHANL   001366           730*   1317*   1517*   1598    1660    1934*   1935    1962    1992*   1993*   1994    2028*   2506
CHANNL  007004          1695*
CH1     001354           725*   1502*   1503*   1506    1512*   1538    1544    1568    1574*   1576*
CH2     001356           726*   1515*   1517    1530    1564    1575*   1577*
CLEAR1  006732          1681*   1683
CLEAR2  006776          1692*   1694
CMSG    012134           761    2251*
COMPAR  011104          1157    1190    1201    1213    1226    1238    1252    1274    1286    1297    1308    1334    2049*
CONV    006376          1608*   1612
CONVR   007066          1706    1709*
CONVRT  010772          1155    1188    1199    1211    1224    1236    1249    1260    1284    1295    1306    1547    2026*
CR    = 000015           410*   2860    2870
CRLF  = 000200           411*   2831    2870
CO      013561           841    2405*
C1      013564          1629    2406*
C2      013566          1822    1912    2407*
C3      013571          1909    2408*
DAC     001410           739*   1322    1325    1567    1571    1647*   1649*   1650    1672*   1980    1983    1986    1989
DASH    012163          1768    2259*
DAWAIT  005026          1248    1259    1283    1377*
DDISP = 177570           417*    600     820
DECPNT  014453          1773*   1802*   2499*
DECTYP  011374          1332    1528    1769    1775    1783    1787    1790    1801    1851    1955    1958    2120*
DECS    003606          1124*   1125
DELAY   001412           740*   1651*   1652*
DELAY1  007130          1712    1720*
DELAY2  007136          1714    1721    1722*
DELAY3  007062          1707*   1708
DELCLR  010242          1811    1823    1913    1922*
DF1     014520           687     694     700     706    2512*
DH1     014274           685    2479*
DH2     014332           704    2484*
DH3     014415           692     698    2493*
DIFLIN  006712          1374    1677*
DISPLA  001142           600*    820*    828*   1406*   2688*   2709*
DISPRE  000174           533*    828
DIST    017124          1684    1744*   1814    3083*
DONE    012306          2084    2276*
DSWR  = 177570           416*    599     819
DT1     014460           686    2504*
DT2     014472           705    2506*
DT3     014510           693     699    2509*
DUMC    011224          1041    1044    1075    2080*
DUMMY   001364           729*   1318*   1564*   1568*   1655    1935*   1994*
DUMW    011152          1032    1055    2062*
EDGE    001414           741*   1319*   1582*   1585*   1605*   1610*   1613*   1614*   1615*   1616*   1620    1665    2000*
                        2003*
EDGFLG  006502          1525*   1590*   1626    1636*   1938*   2008*
EMTVEC= 000030           505*    803*    804*
EM1     014134           684    2461*
EM2     014162           691    2465*
EM3     014212           697    2469*
EM4     014243           703    2474*
ER      010464          1968    1970    1973*
ERMSG   012375          1339    1558    1779    1794    1806    1867    1973    2287*
```

```
ERR     006150          1555    1558#
ERRVEC= 000004           498#    817     818#    829#    896     897#    912#    2649    2650#   2652#   2655#
FIRST   001346           722#   1690#   1756    1758#
FIXADR  005466          1458    1467#
FIXONE  005472           857    1468#
FLAG    001404           737#    836#    853#   1809    1870
GETDAT  010020          1874#   1882
GETEDG  006324          1518    1578    1597#   1936    1995
GMSG    012141           775    2253#
GNS   = ****** U         532    3028    3029    3030    3031    3034    3035    3036
HAFMSG  012565          1803    2310#
HALF    007500          1795    1797#
HEAD1   013724           855    2435#
HEADS   013503          1384    2396#
HT    = 000011           408#   2829    2870
HUNS    014452          2129#   2141#   2143#   2146    2498#
INRNGE  007224          1740    1743#
IOTVEC= 000020           503#    801#    802#
ISERV   001454           756#    859
KBVECT  001340           719#    858    1484
LAST    007342          1754    1771#
LEND    005024          1373    1375#
LESS    011312          2097    2099#
LF    = 000012           409#   2864    2870
LINEA   013462          1852    2393#
LOAD    010220          1908    1911    1915#
LOADY   011272          1817    1819    1918    2093#
LOADO   010224          1916#   1920
LO2     010216          1871    1914#
LSB     012217          1529    2265#
LSBMSG  012154          1770    2257#
MAT     013614          1535    2412#
MAX     001424           745#   1256#   1262    1266#   1630#   1895    1897#   1901
MAXTST  010116          1893    1895#
MEND    012665          1435    2322#
MESP    012647          1959    2319#
MESR    012634          1956    2317#
METST   013621          2066    2413#
MIN     001420           743#   1267#   1327    1889#   1892    1894#   1900
MINUS   012123          2122    2247#
MLSB    013606          1333    2411#
MOFSET  013573          1330    2409#
MSG16   013042          1812    2345#
MSG18   012750          1905    2332#
MSG20   013010          1677    2340#
MSG21   013435          1829    2389#
MSG50   012727           909    2329#
MSG71   013367           869    2382#
NAR     007410          1780    1782#
NARMSG  012434          1784    2293#
NARROW  001344           721#   1689#   1748#   1782    1797
NBEXT   001360           727#    898#    900#    904     910#    911     1457    1465#   1480#
NEXT    007044          1703#   1726
NMBEXT  001362           728#    911#   1480
NOI     012261          1947    2271#
NOIA    010472          1937    1976#
```

```
NOIMSG  012016            1500    2235*
NOITST  010300            1505    1934*
NOI1    010502            1978*   1999
NOI8    010564            1348    1350    1352    1992*
NONE    001E12             772     774     786*
NOTNAR  007260            1747    1753*
NOTNEW  007710            1839    1844*
NOTOK   007154            1724    1728*
NOVTSS  002202             844     847     852     855*    921
NXTCMP  010102            1691*   1899
NXTCVT  006600            1655*   1669
NXTSTA  007652            1831*   1845
NXTY1   007606            1816*   1821
NXT8    010064            1887*   1904
OFFERR  004672            1337    1339*
OFFOK   004700            1338    1341*
OKAY    007146            1717    1719    1725*   1729
OKAYD   011420            2125    2127*
OKMSG   012364            1341    1556    1781    1796    1808    1869    1971    2285*
ONAD    012711            2072    2326*
ONES    014455            2127*   2133*   2134    2136*   2145*   2501*
OUT     001430             747*   1678*   1687*   1741*   1786    1789    1792    1799
OUTMSG  012532            1791    2305*
PC     =%000007            429*    763*    769*    842*    845*    848*    857*    864*    913*    924*    932*    944*    952*
                           960*    967*    995*   1009*   1021*   1032*   1041*   1044*   1055*   1075*   1095*   1112*   1135*
                          1144*   1177*   1248*   1259*   1283*   1332*   1374*   1375*   1380*   1423*   1424*   1425*   1433*
                          1434*   1441*   1442*   1450*   1451*   1452*   1492*   1518*   1520*   1521*   1526*   1528*   1536*
                          1557*   1559*   1572*   1578*   1581*   1591*   1617*   1635*   1703*   1769*   1775*   1783*   1787*
                          1790*   1801*   1811*   1817*   1819*   1823*   1851*   1908*   1911*   1913*   1914*   1918*   1921*
                          1932*   1936*   1937*   1939*   1955*   1958*   1960*   1972*   1974*   1990*   1995*   1999*   2009*
                          2024*   2078*   2085*   2090*   2115*   2147*   2162*   2213*   2216*   2223*   2228    2719*   2725*
                          2781*   2819*   2838*   2845*   2852*   2866*   2868*   2901*   2918*   3073
PEAK    001402             736*   1951    1953*   1957    1969    1977*   1986*   1989*   1997*   2006*   2007*
PERCNT  001426             746*   1641*   1642*   1643*   1644*   1645*   1670
PIRQ   = 177772            415*
PIRQVE= 000240             509*
PLUS    007672            1836    1838*
PLUSR2  011300            2094    2096*
POPRO   001606             784*    787
POS     011406            2121    2124*
POSPEA  010366            1952    1954*
POSRMS  010354            1949    1951*
POSR2   006006            1522    1524*
PR0    = 000000            432*
PR1    = 000040            433*
PR2    = 000100            434*
PR3    = 000140            435*
PR4    = 000200            436*
PR5    = 000240            437*
PR6    = 000300            438*
PR7    = 000340            439*
PS     = 177776            412*    413
PSW    = 177776            413*    872*   1033*   1056*   1057*   1067*   1073*   1385*   2089*
PWRVEC= 000024             50-*    807*    808*   3041*   3042*   3051*   3057*   3069*   3070*
QJEST   012125             786     893    2248*
RANDY   010710            1703    2015*
```

```
RBEG    001634          790    792*
RDCHR = 104404         2565   3034*
RDLIN = 104405          870   2606   3035*
RDOCT = 104406          782   1511   1514   3036*
READ    007175         1734*  1772
RELACC  007636         1810   1827*
RESTI   002206          838    856*
RESVEC= 000010          499*
RET     006500         1627   1635*
RETERR  003704         1142   1144*
RETURN  001610          785*  1396   1607   1656   1701   2035
RMS     001400          735*  1948   1950*  1954   1967   1976*  1980*  1983*  1996*  2004*  2005*
RNA     001372          732*   861*  1704   2015*  2016*  2017*  2018   2021
RNB     001374          733*   862*  2015   2018*  2019*  2020*  2022
RNC     001376          734*   863*  2016   2019   2021*  2022*  2023*
RST     011254          763    769   2087*
RO    =%000000          420*   756    757*   758*   759    765    771    784*   843    846    849    851    858*
                        859*   860*   873*   874*   875    878    881    884    887    890    915*   918*   922*
                        923*   989*   992*   997*  1020*  1031*  1054*  1074*  1109*  1110*  1115*  1121*  1124*
                       1132*  1171*  1175*  1179*  1327*  1328*  1329   1377*  1378*  1386*  1387*  1391   1397*
                       1398*  1399   1403*  1406   1409   1482*  1486   1488*  1489   1490   1493   1494   1598*
                       1599*  1600*  1601   1602*  1603*  1606*  1611*  1648*  1667*  1670   1679*  1681*  1684*
                       1692*  1695*  1696*  1697*  1698*  1699   1700*  1730*  1732*  1771*  1800*  1804   1814*
                       1816   1830*  1831   1834*  1841   1844   1872*  1874   1880*  1883*  1888   1891   1907*
                       1910*  1916   1917   1922*  1924*  2026*  2027*  2028   2029*  2031   2032*  2033*  2036*
                       2040*  2053*  2054*  2056*  2057   2067   2220*  2223   2603   2607*  2610   2626*  2753
                       2754*  2755*  2762*  2763*  2764*  2765*  2766*  2767   2772   2777*  2779*  2783   2785
                       2812   2813*  2818   2823   2826*  2878   2886*  2890   2891   2893*  2894*  2895   2917*
                       3012   3013*  3014   3015*  3016*  3017*  3018*  3043   3068*
R1      =%000001        421*  1414*  1415*  1483*  1484   1486*  1487*  1488   1493   1494   1654*  1668*  1680*
                       1682*  1685*  1693*  1702*  1722*  1725*  1733*  1734   1760   1815*  1820*  1827*  1833*
                       1834   1835   1873*  1881*  1886*  1903*  1915*  1919*  1923*  1926*  2052*  2054   2149*
                       2159*  2160   2604   2608*  2612*  2614*  2616*  2619*  2622   2625*  2879   2916*  3044
                       3067*
R2      =%000002        422*  1257*  1258   1267   1268   1278*  1279   1331*  1400*  1417*  1519*  1523*  1524
                       1567*  1571*  1579*  1586   1588*  1589   1704*  1705*  1707*  1711*  1713   1715*  1716*
                       1718*  1720   1734*  1735*  1736*  1737*  1738*  1739   1743*  1744*  1745*  1746   1749
                       1753   1774*  1782*  1785*  1786*  1789*  1797*  1798*  1799*  1800   1816*  1818*  1831*
                       1832*  1833   1850*  1874*  1875*  1876*  1877*  1878*  1879*  1880   1887*  1898*  1916*
                       1954*  1957*  2093   2095*  2096   2098*  2099   2100*  2101*  2104   2120   2123*  2124
                       2126*  2130   2132*  2152*  2156*  2158*  2159   2605   2609*  2613*  2615*  2617*  2623
                       2624*  3045   3066*
R2POS   006306         1587   1589*
R3      =%000003        423*  1271*  1273   1580*  1583*  1620*  1621   1629*  1630   1686*  1723*  1760*  1761*
                       1762*  1763   1828*  1838   1840*  1846*  1847*  1848*  1849*  1850   1865   1888*  1889
                       1890   1891*  1892   1894   1895   1897   2039*  2105*  2106*  2107*  2108*  2109*  2110*
                       2111*  2114   2561   2562*  2563   2566*  2567   2571   2573   2575*  2577*  2960   2969*
                       2975*  2976*  2979*  2984*  2985*  2986   2995*  3046   3065*
R4      =%000004        424*  1251*  1272   1273*  1524*  1554   1589*  1659*  1660*  1661*  1662*  1663   1835*
                       1837*  1838   1640   1884*  1900*  1902   2961   2963*  2964*  2965*  2966   2967*  2981
                       2983*  2991*  2994*  3047   3064*
R5      =%000005        425*  1155*  1157*  1188*  1190*  1199*  1201*  1211*  1213*  1224*  1226*  1236*  1238*
                       1249*  1252*  1260*  1274*  1284*  1286*  1295*  1297*  1306*  1308*  1320*  1323*  1334*
                       1348*  1350*  1352*  1360*  1364*  1505*  1547*  1565*  1569*  1574   1575   1592*  1641
                       1675*  1841*  1842*  1843*  1853   1859*  1860   1865*  1901*  1902   1934   1940*  1978*
                       1981*  1984*  1987*  1992   2010*  2026   2046*  2049   2050   2059   2060*  2962   2968*
```

```
R6      =%000006   2970* 2972* 2973* 2974* 2975  2993* 3048  3063*
R7      =%000007    426*  428   795*  796*  797
SARSUB   006504     427*  429
SAR1     006530    1320  1323  1565  1569  1641* 1978 1981 1984 1987
SETAA    006256    1646*
SETCH    012232    1581* 1584
SETMSG   012036    1537  2267*
SET1A    006156    1510  2238*
SETB     006220    1520  1521  1564* 1581
SHIFT    006702    1360  1364  1574*
SKIPST   001350    1671  1673*
SKPMSG   012412     723* 1691* 1751* 1774 1777
SLASH    012321    1776  2290*
SP      =%000006    781  1858  2278*
                    428*  756*  762*  768*  776*  783   784   739*  817*  825*  829   873   896*
                    903   904*  912   920   1043  1065  1077  1094  1143* 1391* 1409* 1436* 1467*
                   1512  1515  1530* 1538* 1549* 1621* 1630* 1763* 1853* 1860* 1962* 2067* 2073*
                   2533* 2534* 2537* 2538* 2539  2543* 2544* 2545  2548  2550  2552* 2561* 2566
                   2577  2578* 2579* 2580* 2601* 2602* 2603* 2604* 2605* 2607  2610* 2618* 2619
                   2621  2622* 2624  2625  2626  2649* 2652  2654  2655  2684  2685  2689* 2714
                   2734* 2737* 2753* 2758* 2779  2783* 2812* 2813  2823* 2825  2826  2827* 2829
                   2831  2833  2839  2841* 2843* 2851* 2855  2859  2860  2864  2878* 2879* 2886
                   2887* 2898  2899* 2900* 2910  2911* 2916  2917  2952* 2953  2954  2955* 2960*
                   2961* 2962* 2968  2993  2994  2995  2996* 2997* 3012* 3013  3043* 3044* 3045*
                   3046* 3047* 3048* 3049* 3050  3058* 3062  3063  3064  3065  3066  3067  3068
                   3073*
SPACE    012212    1408  2264*
SPREAD   001406     738* 2050* 2057  2506
STACK =  001100     403*  762   768   799
STATE    012167    1759  2260*
STKLMT=  177774     414*
STREG    001316     710*  975*  981   990*  994*  1005* 1006  1017* 1018  1035* 1036  1038  1059*
                   1060  1062  1068* 1069  1072* 1088* 1089  1091  1104* 1105  1108* 1122* 1123*
                   1126  1133* 1139* 1140  1170* 1180* 1383* 1399* 1401* 1436  1459* 1468* 1601*
                   1608* 1657* 1663* 1699* 1709* 2031* 2037* 2073  2504  2506  2509
SWDIST   007554    1807  1809*
SWR      001140     599*  773   776   783*  797   819*  821   827*  834*  1386  1388  1404  1502
                   1928  2644  2658  2660  2666  2673  2710  2717  2729  2732  3049  3062*
SWREG    000176     534*  827
SW0     = 000001    467*
SW00    = 000001    457*  467
SW01    = 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    1382#   1422#   1431#   1448#   1499#   1509#
TBITVE= 000014          500#
TEMP    001352          724#    973#    977#   1097#   1251    1264    1270    1271    1272#   1322#   1325#   1326#   1329#
                       1331    1347#   1363#   1504#   1516#   1549    1597    1723#   1728#   2030#   2039#   2042#   2043#
                       2044#   2045#   2051
TENS    014454         2128#   2137#   2138    2140#   2144#   2500#
TEST    003662          995    1009    1021    1095    1112    1140#   1177
TESTAD  002422          895#   1423    1433    1450
TESTIT  003654          932     944     952     960     967    1139#
TESTR2  011434         2130#   2135    2139    2142
TKVEC = 000060          507#
TOMSG   012323         1513    2279#
TPVEC = 000064          508#
TRAPVE= 000034          506#    805#    806#
TRTVEC= 000014          501#
TRY     006542         1648#   1674
TRYAG   002276          870#    894
TST1    002556          929#    930
TST10   003044         1002#
TST11   003110         1016#
TST12   003140         1026#
TST13   003232         1042    1049#
TST14   003404         1076    1084#
TST15   003502         1093    1103#
TST16   003556         1119#
TST17   003706         1150#   1153    1154
TST2    002622          941#
TST20   003760         1168#
TST21   004040         1186#
TST22   004070         1197#
TST23   004120         1209#
TST24   004150         1222#
TST25   004200         1234#
TST26   004230         1245#
TST27   004454         1293#
TST3    002646          950#
TST30   004504         1304#
TST31   004534         1315#
TST32   004704         1340    1345#
TST33   004744         1358#
TST34   005002         1370#
TST4    002664          958#
TST5    002702          965#
TST6    002720          972#
TST7    003000          982     988#
TST8    010130         1896    1898#
TYPBAD  007272         1742    1752    1756#
```

```
TYPE  = 104400          761     767     775     781     786     841     855     869     893     909    1330    1333    1339
                       1341    1384    1390    1408    1419    1435    1440    1500    1501    1510    1513    1529    1535
                       1537    1543    1556    1558    1629    1677    1759    1768    1770    1776    1779    1781    1784
                       1788    1791    1794    1796    1803    1806    1808    1812    1813    1822    1829    1852    1858
                       1867    1869    1905    1906    1909    1912    1931    1947    1956    1959    1961    1971    1973
                       2066    2072    2084    2122    2146    2219    2569    2572    2576    2712    2720    2752    2769
                       2771    2774    2776    2780    2787    2834    2987    3028#   3071
TYPEDG  006440          1536    1620#   1960
TYPOC = 104401          2760    2784    3029#
TYPON = 104403          3031#
TYPOS = 104402           778     906    1393    1411    1437    1532    1540    1551    1623    1632    1765    1855    1862
                        1964    2069    2075    3030#
TYPOUT  011510          2131    2143#
TYPRP   010336          1939    1947#   2009
TYPSET  006024          1526    1528#   1591
UNEXP   001432           749#    942    1071    1098
VADR    001332           716#    901    1459    1460    1461
VARLT1  011644          2156    2179#
VARLT2  011654          2158    2184#
VARLT3  011664          2152    2189#
VECTOR  001324           713#    942*   1034*   1058*   1066*   1071*   1087*   1098*   1396*   1462*   1473*   1474*   1477
                        1607*   1656*   1701*   2035*
VECTR1  001330           715#   1463*   1464*   1477*   1478*   1479*
VLIN    011640          1865    2176#
VMP     011634          1969    2174#
VMR     011632          1967    2149    2173#
VSET    011636          1554    2175#
VTFLG   002524           842     845     848     915#
VTINIT  013711          1931    2431#
VTSS    002176           850     853#
VVCT    001334           717#   1462    1463
V1      011610          1192    2163#
V10     011614          2165#
V115    011622          1254    2168#
V144    011620          1299    2167#
V2      011612          1203    1288    2164#
V240    011624          1228    1240    1310    2169#
V5      011626          1276    2170#
V50     011616          1159    1215    2166#
V500    011630          1336    2171#
WFADJ   011540           864    2149#
WFTEST  001422           744#    789*    791*   2154
WIDE    001342           720#   1688*   1755*   1785    1798
WIDMSG  012473          1788    2299#
WPAP    003706          1146#   1441    1451
$APTHD  001000           559     565#
$ASTAT= ****** U        2904    2919
$ATYC   016232          2875    2877#
$ATY1   016206          2873#
$ATY3   016214          2819    2874#
$ATY4   016224          2725    2876#
$AUTOB  001134           596#
$BASE   001250           660#    895    1468    1469    1470
$BDADR  001122           591#
$BDDAT  001126           593#    895*    899     901*    981*   1038*   1062*   1091*   1126*   1127*   1128    1140*   1141
                        2051*   2052    2504    2506    2509
```

```
$BELL   001164            611#  2712  2744
$CDWI   001254            662#
$CHARC  016202           2836*  2846*  2853  2862*  2867#
$CKSWR= ****** U         3034
$CMTAG  001100            579#   794   795   803   809   810   811
$CM3  = 000000            609#
$CNTLG  014765           2586#
$CNTLU  014760           2585#
$CPUOP  001222            634#
$CRLF   001171            613#  1419  1440  2585  2720  2744  2752  2771  2776  2780  2835  2870
$DEVCT  001204            625#
$DEVM   001252            661#
$DOAGN  011774           2215   2221  2227#
$ENDAD  011764            545   2223#  2739
$ENDCT  011744            809   2217#
$ENDMG  012003           2219   2231#
$ENULL  012000           2230#
$ENV    001214            630#  2722  2814  2882  2906
$ENVM   001215            631#   832  2816  2821  2884
$EOP    011710           1428   1445  1455  2207#
$EOPCT  011736            809*  2214#  2218
$ERFLG  001103            582#   751*  2634  2662  2664  2670*  2692  2707*  2744
$ERMAX  001115            588#   812*  2664  2687*  2692
$ERROR  015374            803   2706#
$ERRPC  001116            589#  2504   2506  2509  2714*  2715*  2716  2744  2758
$ERRTB  001256            679#  2766
$ERRTY  015570           2719   2751#
$ERTTL  001112            586#  2713*  2744
$ESCAP  001162            610#   750*   753*   811*  2686*  2735  2737  2744
$ETABL  001214            629#
$ETEND  001256            571    663#
$FATAL  001176            622#  2910*
$FFLG   016452           2873*  2876*  2904  2913*  2921#
$FILLC  001156            607#  2839   2870
$FILLS  001155            606#  2870
$GDADR  001120            590#
$GDDAT  001124            592#   931*   934*   935   943*   951*   959*   966*   974*   991*  1004*  1039*  1086*
                         1107*  1120*  1128  1139  1141  1172*  2049*  2053  2504  2506
$GET42  011754           2220#
$GTSWR= ****** U         3033
$HD   = 000000            400
$HIBTS  001000            566#
$HIOCT  015114           2623*  2628#
$ICNT   001104            583#  2677*  2678  2680*  2691
$ILLUP  017106           3041   3057  3076#
$INTAG  001135            597#
$ITEMB  001114            587#  2716*  2724  2744  2755
$LF     001172            614#  2576   2585  2744  2870
$LFLG   016451           2914*  2920#
$LPADR  001106            584#   813*   929*  1154*  2065*  2083*  2668*  2684*  2689  2691
$LPERR  001110            585#   814*   930*  1153*  2064*  2082*  2668   2685*  2691  2734
$MADR1  001226            647#
$MADR2  001232            651#
$MADR3  001236            654#
$MADR4  001242            657#
$MAIL   001174            567    571   620#   831  2683  2722  2814
```

```
$MAMS1  001224            641#
$MAMS2  001230            649#
$MAMS3  001234            652#
$MAMS4  001240            655#
$MBADR  001002            567#
$MFLG   016450           2874#   2880    2915#   2919#
$MNEW   015003           2589#
$MSGAD  001210            627#   2890#   2893
$MSGLG  001212            628#   2895#
$MSGTY  001174            621#   2888    2896#   2908    2912#
$MSWR   014772           2587#
$MTYP1  001225            642#
$MTYP2  001231            650#
$MTYP3  001235            653#
$MTYP4  001241            656#
$MXCNT  015372           2681    2691#
$NULL   001154            605#   2841    2870
$NWTST= 000001            926#    938#    947#    955#    962#    969#    985#    999#   1013#   1023#   1046#   1081#   1100#
                         1116#   1147#   1165#   1183#   1194#   1206#   1219#   1231#   1242#   1290#   1301#   1312#   1342#
                         1355#   1367#
$OCNT   016676           2959#   2988#   3001#
$OMODE  016700           2954#   2958#   2963    2966#   2977#   3003#
$OVER   015356           2645    2661    2669    2679    2688#
$PASS   001202            624#    831#   1372    1432#   1449#   2062    2080    2211#   2212#   2230    2675    2692
$PASTM  001006            569#
$POWER  017114           3072    3079#
$PWRAD  017102           3074#
$PWRON  016742            807    3041#   3069
$PWRMG  017076           3072#
$PWRUP  017014           3051    3057#
$QUES   001170            612#   2569    2585    2744    2870
$RDCHR  014522           2533#   3034
$RDDEC= ****** U         3037
$RDLIN  014642           2561#   3035
$RDOCT  015014           2601#   3036
$RDSZ = 000010           2554#
$RTNAD  011776           2229#
$R2A  = ****** U         3037
$SAVRE= ****** U         3037
$SAVR6  017112           3050#   3058    3059#   3060#   3078#
$SCOPE  015116            801    2643#
$SETUP= 000037            680#    800     803     805     807     809     810     811     813    2209    2522    2591
                         2644    2707    2732    2739
$STUP = 177777            680#
$SVLAD  015322           2653    2682#
$SVPC = 000214            543#    548
$SWR  = 167400            390#    400     514     515     516     517     518     519     520     521     609     610     611
                          810     811     813     814     930     942     951     959     966     973     989    1003    1017
                         1027    1050    1085    1104    1120    1151    1169    1187    1198    1210    1223    1235    1246
                         1294    1305    1316    1346    1359    1371    2202    2210    2222    2228    2230    2635    2636
                         2637    2638    2639    2644    2656    2658    2659    2662    2663    2664    2671    2672    2673
                         2685    2688    2691    2698    2699    2700    2701    2702    2710    2717    2729    2732    2744
                         3075
$SWREG  001216            632#    834
$SWRMK= 000000            521     522    2639    2640    2660
$TESTN  001200            623#   2683#
```

```
$TIMES  001160          609*   810*   973*  1003*  1085*  1151*  1169*  1187*  1198*  1210*  1223*  1235*  1246*
                       1294*  1305*  1316*  1346*  1359*  1371*  2210*  2671*  2678   2681*  2691
$TKB    001146          602*   757    922   2520   2537   2543
$TKS    001144          601*   840*   871*   916    980*  1011*  1282*  2088*  2194*  2520   2535   2541
$TN   = 000035          390*   400    926    930*   938    942*   947    951*   955    959*   962    966*   969
                        973*   982    985    989*   999   1003*  1013   1017*  1023   1027*  1042   1046   1050*
                       1076   1081   1085*  1093   1100   1104*  1116   1120*  1147   1151*  1152   1165   1169*
                       1183   1187*  1194   1198*  1206   1210*  1219   1223*  1231   1235*  1242   1246*  1290
                       1294*  1301   1305*  1312   1316*  1340   1342   1346*  1355   1359*  1367   1371*
$TPB    001152          604*  2104*  2114*  2859*  2870
$TPF..G 001157          608*  2808   2870
$TPS    001150          603*  2102   2112   2857   2870
$TRAP   016702          805   3012*
$TRP  = 000007         3020*  3029*  3030*  3031*  3032*  3034   3035*  3036*  3037*
$TRPAD  016724         3017   3027*
$TSTM   001004          568*
$TSTNM  001102          581*  1152*  2209*  2634   2660   2682*  2683   2688   2692   2709   2744
$TTYIN  014750         2562   2563   2580   2584*
$TYPBN= ****** U       3032
$TYPDS= ****** U       3032
$TYPE   015724         2808*  2901   3020   3029
$TYPEC  016136         2838   2845   2852   2857*  2858
$TYPEX  016204         2863   2865   2868*
$TYPOC  016500         2957*  3029
$TYPON  015514         2956   2959*  3031
$TYPOS  016454         2952*  3030
$UNIT   001206          626*
$UNITM  001010          570*
$USWR   001220          633*
$VECT1  001244          658*  1473   1475
$VECT2  001246          659*
$XTSTR  015126         2647*
$$GET4= 000000         2222*
$OFILL  016677         2953*  2957*  2967   3002*
$40CAT= ****** U       2644   2719
.     = 037744          528*   532*   543    544*   546*   548*   549*   555    556*   558*   560*   578*   615
                        798    813    814   2230   2233   2502*  2520   2584*  2585   2591   2691   2692   2744
                       2790*  2870   2922*  3053   3077   3083*  3084*
.$ASTA= ****** U       2874   2877
.$X   = 001000          555*   560
```

```
COMMEN     1#    510#
DUMWRN   683#   1027    1050
ENDCOM     1#    510#
ERROR    404#    752     933     945     953     961     968     983     996    1010    1022    1040    1063    1078    1092
        1096    1113    1130    1160    1178    1193    1204    1216    1229    1241    1255    1277    1289    1300    1311
ESCAPE     1#    510#    749
GETPRI     1#    510#
GETSWR     1#    510#
MULT       1#    510#
NEWTST     1#    510#    926     938     947     955     962     969     985     999    1013    1023    1046    1081    1100
        1116    1147    1165    1183    1194    1206    1219    1231    1242    1290    1301    1312    1342    1355    1367
POP        1#    510#   2624    2916    2917    3062    3063
PUSH       1#    510#   2603    2877    2879    2900    3043    3049
REPORT     1#    510#
SCOPE    405#    941     950     958     965     972     988    1002    1016    1026    1049    1084    1103    1119    1134
        1168    1186    1197    1209    1222    1234    1245    1293    1304    1315    1345    1358    1370
SETPRI     1#    510#
SETTRA  3020#   3029    3030    3031    3034    3035    3036
SETUP      1#    510#    793
SKIP       1#    510#    876     879     882     885     888     891     902     917     919     921     936     982    1042
        1076    1093    1129    1142    1340    1389    1929    2151    2153
SLASH      1#    510#
SPACE    510#
STARS      1#    510#    541     552     554     561     574     615     618     926     928     938     940     947     949
         955     957     962     964     969     971     985     987     993    1001    1013    1015    1023    1025    1046
        1048    1081    1083    1100    1102    1116    1118    1147    1149    1165    1167    1183    1185    1194    1196
        1206    1208    1219    1221    1231    1233    1242    1244    1290    1292    1301    1303    1312    1314    1342
        1344    1355    1357    1367    1369    2199    2519    2525    2554    2593    2631    2694    2746    2793    2872
        2929    3006    3039    3055
SWRSU      1#    510#    815#
TRMTRP  3020#
TYPBIN     1#    510#
TYPDEC     1#    510#
TYPNAM     1#    510#
TYPNUM     1#    510#
TYPOCS     1#    510#    776     904    1391    1409    1530    1538    1549    1621    1630    1763    1853    1860    1962
        2067    2073
TYPOCT     1#    510#   2758    2782
TYPTXT     1#    510#
$$CMRE   572#
$$CMTM   572#
$$ESCA     1#    510#
$$NEWT     1#    510#    926     938     947     955     962     969     985     999    1013    1023    1046    1081    1100
        1116    1147    1165    1183    1194    1206    1219    1231    1242    1290    1301    1312    1342    1355    1367
$$SET   3020#   3029    3030    3031    3034    3035    3036
$$SETM   831#
$$SKIP     1#    510#    982    1042    1076    1093    1340
.EQUAT     1#    390#    400
.HEADE     1#    390#
.KT11      1#
.SETUP     1#    390#    680
.SWRHI     1#    390#    510
.SWRLO   522#
.$ACTI     1#    390#    539
.$APTB     1#    390#    616#
.$APTH     1#    390#    550
```

```
.$APTY      1#    390#   2870
.$ASTA      1#
.$CATC      1#    390#    526
.$CMTA      1#    390#    572
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#    390#   2197
.$ERRO      1#    390#   2692
.$ERRT      1#    390#   2744
.$MULT      1#
.$PARM    390#
.$POWE      1#    390#   3037
.$RAND      1#    390#
.$RDDE      1#
.$RDOC      1#    390#   2591
.$READ      1#    390#   2517
.$R2AZ      1#
.$SAVE      1#    390#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    390#   2629
.$SIZE      1#
.$SPAC    390#
.$SUPR      1#
.$SWDO    390#
.$TRAP      1#    390#   3004
.$TYPB      1#
.$TYPD      1#    390#
.$TYPE      1#    390#   2791
.$TYPO      1#    390#   2927
.$40CA      1#
.1170       1#
```

```
ADC     1616  1738  1849  1878  2005  2007  2017  2020  2023  2045
ADD      901  1143  1325  1459  1460  1461  1462  1463  1467  1472  1478  1567  1576  1577  1610
        1628  1649  1696  1786  1798  1799  1833  1879  1980  1986  1993  2015  2016  2018  2019
        2021  2022  2027  2039  2619  2766  2827  2887  2899  2911  2955  2965
ASL     1328  1642  1643  1644  1645  1715  1743  2612  2614  2616  2763  2764  2765  3016
ASR     1613  1614  1615  1673  1735  1736  1737  1745  1762  1843  1846  1647  1848  1875  1876
        1877  2004  2006  2042  2043  2044  2105  2106  2107  2108  2109  2894
BEQ      774   833   850   867   982  1129  1373  1405  1458  1485  1627  1706  1712  1714  1721
        1778  1793  1810  1871  1929  2131  2151  2221  2611  2659  2661  2663  2667  2676  2708
        2711  2733  2736  2768  2773  2786  2817  2830  2865  2881  2885  2905  2907  2982
BGE     1666  1747  1893  2679
BGT     1555  1671  1968  1970  2058  2215  2551  2989
BHI     2665
BIC      758   840   923   994  1127  1387  1474  1503  1705  2100  2110  2212  2538  2544  2552
        2618  2979
BICB     874
BIS      871   980  1011  1282  1398  1600  1657  1662  1698  2088  2101  2111  2194  2984  2985
BISB    2037  2143  2144  2145  2755
BIT     1388  1404  1928  2644  2658  2666  2673  2710  2717  2732
BITB     832  2816  2821  2853  2884
BLE     1754  1805  1839  1866  1896  2125
BLOS    2564
BLT     1265  1740  2097  2549  2844  2990
BMI      917  1724
BNE      760   766   772   798   822   838   844   847   852   876   879   882   885   888   891
         919   936   978   993  1111  1125  1142  1176  1263  1269  1280  1379  1389  1416  1418
        1491  1584  1604  1612  1653  1669  1674  1683  1694  1708  1726  1731  1750  1757  1772
        1821  1845  1882  1899  1904  1920  1925  1927  2002  2034  2041  2063  2081  2135  2139
        2155  2540  2546  2568  2574  2645  2674  2718  2723  2740  2756  2778  2815  2822  2824
        2832  2840  2854  2861  2883  2889  2892  2909  2980  3061
BPL     1007  1019  1037  1061  1070  1090  1106  1522  1587  1717  1836  1949  1952  2055  2094
        2103  2113  2121  2161  2536  2542  2730  2809  2858  2978
BR       787   790   824   894   902   921  1042  1064  1076  1079  1093  1337  1338  1340  1407
        1420  1426  1443  1453  1466  1495  1507  1527  1719  1727  1729  1742  1752  1780  1795
        1807  1868  2142  2153  2157  2547  2570  2620  2647  2653  2656  2669  2672  2728  2761
        2788  2811  2837  2847  2856  2863  2875  2897  2956  2971  2992  3053  3077
CLR      753   789   796   810   811   831   836   872   898   915   974  1004  1033  1072  1073
        1133  1180  1247  1256  1377  1383  1385  1432  1449  1464  1479  1487  1519  1579  1605
        1647  1648  1678  1681  1686  1687  1688  1689  1690  1691  1692  1728  1818  1827  1828
        1922  1950  1953  1976  1977  1996  1997  2030  2089  2095  2209  2210  2608  2609  2671
        2586  2754  2969  3059
CLRB    1476  2127  2128  2129  2136  2140  2575  2670  2836  2862  2913  2914  2915
CMP      773   797   821   843   846   849   851   903   935  1043  1065  1077  1094  1128  1141
        1264  1268  1279  1484  1490  1493  1494  1554  1626  1665  1670  1713  1720  1739  1746
        1753  1804  1838  1844  1865  1892  1895  1902  1967  1969  2057  2096  2124  2539  2545
        2548  2550  2563  2654  2678  2739
CMPB     759   765   771   875   878   881   884   887   890  2134  2138  2567  2573  2660  2664
        2722  2814  2829  2831  2839  2860  2864  2882
DEC      910   918   977   992  1110  1124  1175  1415  1417  1465  1583  1603  1611  1652  1668
        1682  1693  1707  1725  1730  1771  1820  1881  1898  1903  1919  1924  1926  2001  2023
        2040  2132  2213  2762
DECB    1378  2843  2846  2977  2988
EMT      404
HALT     532  1930  2731  2741  2810  3052  3076
INC      751   853   900   990  1005  1266  1278  1401  1471  1582  1608  1667  1709  1716  1722
        1723  1741  1744  1748  1751  1755  1758  1859  2000  2211  2677  2713  2912  2983  2991
```

```
        3060
INCB    2133    2137    2141    2682    2707    2866
IOT      405
JMP      536     537     538     764     770     868     877     880     883     886     889     892    1428    1445    1455
        2195    2228
JSR      763     769     842     845     848     857     864     932     944     952     960     967     995    1009    1021
        1032    1041    1044    1055    1075    1095    1112    1155    1157    1177    1188    1190    1199    1201    1211
        1213    1224    1226    1236    1238    1248    1249    1252    1259    1260    1274    1283    1284    1286    1295
        1297    1306    1308    1320    1323    1332    1334    1348    1350    1352    1360    1364    1374    1423    1424
        1425    1433    1434    1441    1442    1450    1451    1452    1505    1518    1520    1521    1526    1528    1536
        1547    1565    1569    1578    1581    1591    1703    1769    1775    1783    1787    1790    1801    1811    1817
        1819    1823    1851    1908    1911    1913    1918    1936    1937    1939    1955    1958    1960    1978    1981
        1984    1987    1995    1999    2009    2223    2719    2725    2819    2838    2845    2852    2901
MOV      750     756     757     762     768     776     783     784     791     795     799     801     802     803     804
         805     806     807     808     809     813     814     817     818     819     820     825     827     828     829
         834     858     859     860     861     862     863     873     895     896     897     904     911     912     922
         929     930     931     942     943     951     959     966     973     975     976     981     989     991     997
        1003    1017    1020    1031    1034    1035    1038    1039    1054    1056    1058    1059    1062    1066    1067
        1068    1071    1074    1085    1086    1087    1088    1091    1097    1093    1104    1107    1108    1109    1115
        1120    1121    1122    1126    1132    1139    1140    1151    1152    1153    1154    1169    1170    1171    1172
        1173    1179    1187    1198    1210    1223    1235    1246    1251    1257    1258    1267    1270    1271    1273
        1294    1305    1316    1317    1318    1319    1322    1327    1331    1346    1347    1359    1363    1371    1382
        1386    1391    1396    1399    1400    1403    1406    1409    1414    1422    1427    1431    1436    1444    1448
        1454    1468    1469    1470    1473    1477    1480    1482    1483    1486    1488    1499    1502    1504    1509
        1512    1515    1516    1517    1524    1525    1530    1538    1544    1546    1549    1564    1568    1574    1575
        1580    1589    1590    1597    1601    1602    1606    1607    1620    1621    1630    1641    1646    1650    1651
        1654    1656    1659    1660    1663    1679    1680    1684    1685    1695    1699    1700    1701    1702    1704
        1711    1718    1732    1733    1734    1760    1763    1774    1782    1785    1789    1797    1800    1814    1815
        1816    1830    1831    1834    1835    1840    1841    1850    1853    1860    1872    1873    1874    1880    1883
        1884    1895    1886    1887    1838    1889    1890    1891    1894    1897    1900    1901    1907    1910    1915
        1916    1923    1934    1935    1938    1954    1957    1962    1992    1994    1998    2006    2026    2028    2031
        2032    2035    2036    2049    2050    2051    2052    2053    2064    2065    2067    2073    2082    2083    2098
        2099    2126    2149    2152    2156    2158    2159    2216    2220    2533    2534    2561    2562    2577    2578
        2579    2580    2601    2602    2603    2604    2605    2607    2622    2623    2624    2625    2626    2649    2650
        2652    2655    2668    2670    2681    2684    2685    2688    2689    2709    2714    2734    2737    2753    2758
        2767    2772    2777    2779    2783    2812    2813    2818    2826    2841    2878    2879    2886    2890    2895
        2896    2898    2900    2910    2916    2917    2952    2960    2961    2962    2968    2973    2993    2994    2995
        2996    2997    3012    3013    3017    3041    3042    3043    3044    3045    3046    3047    3048    3049    3050
        3051    3057    3058    3062    3063    3064    3065    3066    3067    3068    3069    3070    3073
MOVB     812    1123    1475    1548    1655    1773    1802    2104    2114    2537    2543    2565    2571    2610    2683
        2687    2716    2724    2823    2851    2859    2873    2874    2876    2953    2954    2957    2958    2959    2963
        2966    2967    2986    3015
NEG     1523    1588    1837    2056    2123    2964
NOP     1150    2208    2224    2225    2226
RESET    792     856     865     979    1008    1281    2087    2222
ROL      934    2613    2615    2617    2970    2972    2973    2974    2976
RTI      754     785     826    2553    2581    2627    2690    2743    2828    2998    3075
RTS      913     924    1135    1144    1375    1380    1492    1557    1559    1572    1592    1617    1635    1675    1914
        1921    1932    1940    1972    1974    1990    2010    2024    2046    2060    2078    2085    2090    2115    2147
        2162    2781    2868    2918    3018
SUB     1057    1272    1326    1329    1545    1571    1585    1672    1761    1832    1842    1983    1989    2003    2054
        2715    2893
SWAB    1397    1599    1661    1697    2029
TRAP    3020    3029    3030    3031    3034    3035    3036
TST      837     866     899     920    1262    1372    1457    1489    1586    1749    1756    1777    1792    1809    1870
        1917    1948    1951    2059    2062    2080    2093    2120    2130    2150    2154    2160    2621    2651    2675
```

```
        2729  2735  2785  2825  2833  2855  2888  2906  2908  2981  3014
TSTB     916  1006  1018  1036  1060  1069  1089  1105  2102  2112  2535  2541  2662  2808  2857
        2880  2891  2904
WAIT    1402  1609  1658  1664  1710  2038
.ASCII   612   613  2322  2332  2345  2353  2359  2396  2435  2442  2445  2448  2451  2454
.ASCIZ   611   614  2231  2235  2238  2257  2259  2260  2263  2264  2265  2267  2270  2271  2273
        2276  2278  2279  2285  2287  2290  2293  2299  2305  2310  2317  2319  2326  2329  2336
        2340  2371  2382  2389  2393  2399  2405  2406  2407  2408  2409  2411  2412  2413  2457
        2461  2465  2469  2474  2479  2484  2493  2505  2586  2587  2589  2789  3079
.BLKB   2584
.BLKW   3083  3084
.BYTE    581   582   587   588   596   597   605   606   607   608   630   631   641   642   649
         650   652   653   655   656   779   780   907   908  1394  1395  1412  1413  1438  1439
        1533  1534  1541  1542  1552  1553  1624  1625  1633  1634  1766  1767  1856  1857  1863
        1864  1965  1966  2070  2071  2076  2077  2230  2247  2248  2249  2251  2253  2417  2425
        2431  2498  2499  2500  2501  2582  2583  2726  2727  2919  2920  2921  2999  3000  3001
        3002
.DSABL  2522
.ENABL     1   390  2520
.END    3086
.ENDC    395   404   496   510   516   520   521   522   537   542   546   548   553   555   562
         575   579   581   609   610   611   612   616   619   641   649   652   655   658   659
         660   661   662   665   680   751   780   781   799   800   803   805   807   809   810
         811   813   815   836   877   880   883   886   889   892   903   908   909   918   920
         922   927   928   929   930   937   939   940   941   942   948   949   950   951   956
         957   958   959   963   964   965   966   970   971   972   973   974   983   986   987
         988   989  1000  1001  1002  1003  1004  1014  1015  1016  1017  1024  1025  1026  1027
        1043  1047  1048  1049  1050  1077  1082  1083  1084  1085  1086  1094  1101  1102  1103
        1104  1117  1118  1119  1120  1130  1143  1148  1149  1150  1151  1152  1166  1167  1168
        1169  1170  1184  1185  1186  1187  1188  1195  1196  1197  1198  1199  1207  1208  1209
        1210  1211  1220  1221  1222  1223  1224  1232  1233  1234  1235  1236  1243  1244  1245
        1246  1247  1291  1292  1293  1294  1295  1302  1303  1304  1305  1306  1313  1314  1315
        1316  1317  1341  1343  1344  1345  1346  1347  1356  1357  1358  1359  1360  1368  1369
        1370  1371  1372  1390  1395  1396  1413  1414  1534  1535  1542  1543  1553  1554  1625
        1626  1634  1635  1767  1768  1857  1858  1864  1865  1930  1966  1967  2071  2072  2077
        2078  2152  2154  2200  2201  2202  2204  2206  2209  2215  2218  2219  2220  2222  2228
        2230  2233  2520  2521  2522  2526  2554  2555  2562  2564  2567  2569  2585  2591  2594
        2596  2629  2632  2635  2640  2644  2646  2657  2660  2661  2662  2664  2666  2673  2677
        2682  2684  2688  2691  2692  2695  2698  2707  2714  2719  2720  2721  2729  2739  2743
        2744  2747  2762  2791  2794  2823  2873  2874  2877  2904  2919  2930  3007  3013  3016
        3028  3029  3030  3031  3032  3033  3034  3035  3036  3037  3040  3049  3050  3056  3062
        3063  3073  3075  3082
.EQUIV   404   405   413   428   429   458   459   460   461   462   463   464   465   466   467
         486   487   488   489   490   491   492   493   494   495
.EVEN    619  2339  2502  2790  2922  3081  3082
.IF      391   402   468   496   517   519   520   521   522   535   541   544   546   552   554
         561   574   578   580   609   610   611   615   616   618   641   649   652   655   658
         659   660   661   662   663   665   680   750   779   780   794   799   801   803   805
         807   809   810   811   813   831   876   879   882   885   888   891   902   907   908
         917   919   921   926   928   930   936   938   940   942   947   949   951   955   957
         959   962   964   966   969   971   973   974   982   985   987   989   999  1001  1003
        1004  1013  1015  1017  1023  1025  1027  1042  1046  1048  1050  1076  1081  1083  1085
        1086  1093  1100  1102  1104  1116  1118  1120  1129  1142  1147  1149  1151  1152  1165
        1167  1169  1170  1183  1185  1187  1188  1194  1196  1198  1199  1206  1208  1210  1211
        1219  1221  1223  1224  1231  1233  1235  1236  1242  1244  1246  1247  1290  1292  1294
        1295  1301  1303  1305  1306  1312  1314  1316  1317  1340  1342  1344  1346  1347  1355
```

```
                1357  1359  1360  1367  1369  1371  1372  1389  1394  1395  1412  1413  1533  1534  1541
                1542  1552  1553  1624  1625  1633  1634  1766  1767  1856  1857  1863  1864  1929  1965
                1966  2070  2071  2076  2077  2151  2153  2199  2200  2201  2202  2203  2204  2208  2214
                2217  2219  2220  2222  2228  2230  2231  2233  2519  2521  2522  2525  2526  2554  2562
                2563  2567  2568  2584  2585  2591  2593  2596  2608  2631  2634  2639  2644  2656  2658
                2659  2660  2662  2663  2664  2673  2675  2683  2685  2690  2691  2692  2694  2697  2707
                2710  2717  2719  2720  2722  2729  2732  2739  2743  2744  2746  2761  2777  2793  2814
                2872  2874  2877  2904  2919  2929  3006  3012  3016  3020  3029  3030  3031  3032  3033
                3034  3035  3036  3037  3039  3049  3050  3055  3062  3063  3071  3073  3075  3079
.IFF             402   517   520   521   522   542   546   548   553   555   562   575   578   581   609
                 616   619   750   779   781   799   876   879   882   885   888   891   902   908   917
                 919   921   926   927   928   929   930   936   939   940   941   942   948   949   950
                 951   956   957   958   959   963   964   965   966   970   971   972   973   983   986
                 987   988   989  1000  1001  1002  1003  1014  1015  1016  1017  1024  1025  1026  1027
                1043  1047  1048  1049  1050  1077  1082  1083  1084  1085  1094  1101  1102  1103  1104
                1117  1118  1119  1120  1129  1142  1147  1148  1149  1150  1151  1166  1167  1168  1169
                1184  1185  1186  1187  1195  1196  1197  1198  1207  1208  1209  1210  1220  1221  1222
                1223  1232  1233  1234  1235  1243  1244  1245  1246  1247  1291  1292  1293  1294  1302
                1303  1304  1305  1313  1314  1315  1316  1317  1341  1343  1344  1345  1346  1347  1356
                1357  1358  1359  1360  1368  1369  1370  1371  1372  1389  1395  1413  1414  1534  1542
                1553  1554  1625  1626  1634  1635  1767  1768  1857  1858  1864  1865  1929  1966  2071
                2076  2078  2151  2153  2200  2203  2208  2215  2218  2230  2520  2522  2526  2528  2533
                2554  2555  2564  2568  2585  2594  2632  2657  2660  2661  2664  2691  2692  2695  2697
                2710  2739  2744  2747  2762  2791  2794  2873  2930  3007  3013  3040  3056  3073
.IFT            2528  2533  2612  2628  2629  2672  2720
.IFTF           2522  2526  2529  2608  2612  2628  2670  2719
.IIF             390   395   400   514   515   516   518   521   522   532   615   619   777   800   803
                 809   810   811   813   814   905  1392  1410  1531  1539  1550  1622  1631  1764  1854
                1861  1963  2068  2074  2201  2202  2209  2210  2230  2233  2520  2577  2585  2591  2635
                2636  2637  2638  2639  2640  2644  2671  2672  2688  2691  2692  2698  2699  2700  2701
                2702  2707  2732  2739  2744  2759  2784  2870  3028  3029  3030  3031  3034  3035  3036
.IRP             680   926   938   947   955   962   969   985   999  1013  1023  1046  1081  1100  1116
                1147  1165  1183  1194  1206  1219  1231  1242  1290  1301  1312  1342  1355  1367  2208
                2603  2624  2878  2879  2900  2916  2917  3043  3049  3062  3063
.LIST              1   390   510   521   532   609   616   619   680   815   926   930   938   942   947
                 951   955   959   962   966   969   973   985   989   999  1003  1013  1017  1023  1027
                1046  1050  1081  1085  1100  1104  1116  1120  1147  1151  1165  1169  1183  1187  1194
                1198  1206  1210  1219  1223  1231  1235  1242  1246  1290  1294  1301  1305  1312  1316
                1342  1346  1355  1359  1367  1371  2209  2222  2554  2639  2739  3020  3028  3029  3030
                3031  3032  3034  3035  3036  3037
.MACRO             1   522   572   683   831  3020
.MCALL           390   510   616   815
.MEXIT           664
.NLIST             1   390   510   521   532   609   616   619   680   815   926   930   938   942   947
                 951   955   959   962   966   969   973   985   989   999  1003  1013  1017  1023  1027
                1046  1050  1081  1085  1100  1104  1116  1120  1147  1151  1165  1169  1183  1187  1194
                1198  1206  1210  1219  1223  1231  1235  1242  1246  1290  1294  1301  1305  1312  1316
                1342  1346  1355  1359  1367  1371  2209  2222  2554  2639  2739  3020  3028  3029  3030
                3031  3032  3034  3035  3036  3037
.PAGE            572   665
.REM               1
.REPT            532
.SBTTL           400   510   526   535   539   550   572   616   665   709   755   788   793   839   854
                 926   938   947   955   962   969   985   999  1013  1023  1046  1081  1100  1116  1145
                1147  1165  1183  1194  1206  1219  1231  1242  1290  1301  1312  1342  1355  1367  1381
                1421  1430  1447  1456  1498  1508  2197  2234  2517  2591  2629  2692  2744  2791  2870
```

```
        2927    3004    3020    3037
.TITLE   390
.WORD    532     533     534     547     566     567     568     569     570     571     580     583     584     585     586
         589     590     591     592     593     594     595     596     599     600     621     622     623     624     625
         626     627     628     632     633     634     647     651     654     657     658     659     660     661     662
        2214    2217    2229    2628    2770    2775    2820    2867    2902    3003    3072    3074

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZADLA.SEQ/SOL/CRF/PAGNUM/NL:TOC=DZADLA.SML,DZADLA.P11
RUN-TIME:  39 51 7 SECONDS
RUN-TIME RATIO:  159/98=1.6
CORE USED:  34K  (67 PAGES)
```

```
  10                        ...B1    2243  012074  023440    ...B5
  63                        ...C1    2296  012456  020051    ...C5
 116                        ...D1    2348  013064  020040    ...D5
 172                        ...E1    2404  013560     000    ...E5
 213                        ...F1                             ...F5
 241                        ...G1    2463  014150  020056    ...G5
 297                        ...H1    2507  014500  001124    ...H5
 352                        ...I1    2526                    ...I5
 408             000011     ...J1    2582  014746     000    ...J5
 464                        ...K1    2600                    ...K5
 520                        ...L1    2638                    ...L5
 548             000214     ...M1    2694                    ...M5
 581  001102        000     ...N1    2750                    ...N5

 637                        ...B2    2800                    ...B6
 674                        ...C2    2856  016134  000724    ...C6
 718  001336   000000       ...D2    2912  016422  005237    ...D6
 764  001510   000137       ...E2    2936                    ...E6
 797  001644   022706       ...F2    2992  016652  000744    ...F6
 845  002144   004737       ...G2    3013  016704  016600    ...G6
 863  002244   012737       ...H2    3046  016764  010346    ...H6
 912  002516   012637       ...I2  ADDW11=  000000           ...I6
 934  002606   006137       ...J2  BEGINA   005260           ...J6
 978  002754   001375       ...K2  CONV     006376           ...K6
1032  003146   004737       ...L2  GNS    = ****** U         ...L6
1088  003430   012777       ...M2  NXTCMP   010102           ...M6
1125  003610   001376       ...N2  RETERR   003704           ...N6

1154  003732   012737       ...B3  SET8     006220           ...B7
1203  004114   011612       ...C3  TADDR    001370           ...C7
1251  004256   013704       ...D3  TYPON  = 104403           ...D7
1299  004500   011620       ...E3  $CRLF    001171           ...E7
1321  004572   000062       ...F3  $MSGTY   001174           ...F7
1351  004734   000007       ...G3  $TPB     001152           ...G7
1390  005100   104400       ...H3  NEWTST      1#     510#   ...H7
1430                        ...I3  .SERRT      1#     390#   ...I7
1465  005460   005337       ...J3             2711    2733   ...J7
1517  005756   013737       ...K3             1425    1433   ...K7
1567  006172   063702       ...L3  .BLKB      2584    2465   ...L7
1602  006350   012700       ...M3  .IFF         402     517   ...M7
1646  006530   012737       ...N3   RUN-TIME:  39 51 7 SECON ...N7

1685  006744   012701       ...B4   **END** USER DAVIES,TOM  ...B8
1741  007216   005237       ...C4
1791  007454   104400       ...D4
1834  007662   010120       ...E4
1883  010044   012700       ...F4
1931  010272   104400       ...G4
1984  010532   004537       ...H4
2034  011024   001376       ...I4
2070  011204      002       ...J4
2095  011276   005002       ...K4
2127  011420   105037       ...L4
2172  011630   000062       ...M4
2206                        ...N4
```